DOKUZ EYLÜL UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

NONLINEAR DYNAMICAL STATE FEEDBACK DESIGN FOR TRACKING AND CHAOTIFICATION

by

Savaş ŞAHİN

May, 2010

İZMİR

NONLINEAR DYNAMICAL STATE FEEDBACK DESIGN FOR TRACKING AND CHAOTIFICATION

A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of Dokuz Eylül University
In Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy in Electrical and Electronics Engineering, Department of Electrical and Electronics Engineering

by
Savaş ŞAHİN

May, 2010
İZMİR

Ph.D. THESIS EXAMINATION RESULT FORM

We have read the thesis entitled "**NONLINEAR DYNAMICAL STATE FEEDBACK DESIGN FOR TRACKING AND CHAOTIFICATION**" completed by **SAVAŞ ŞAHİN** under supervision of **PROF. DR. CÜNEYT GÜZELİŞ** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Doctor of Philosophy.

Prof. Dr. Cüneyt GÜZELİŞ

Supervisor

Prof. Dr. Ferit Acar SAVACI

Thesis Committee Member

Prof. Dr. Saide SARIGÜL

Thesis Committee Member

Assoc. Prof. Dr. Müştak Erhan YALÇIN

Examining Committee Member

Asst. Prof. Dr. Güleser KALAYCI DEMİR

Examining Committee Member

Prof. Dr. Mustafa SABUNCU
Director
Graduate School of Natural and Applied Sciences

ACKNOWLEDGMENTS

I especially want to thank my advisor, Prof. Dr. Cüneyt Güzeliş, for excellent guidance, support, confidence, encouragement, and valuable suggestions during my doctoral study.

I would like to give thanks to my thesis committee members Prof. Dr. F. Acar Savacı and Prof. Dr. Saide Sarıgül for the benefit of their collective wisdom and helpful comments. And also, I would like to thank Prof. Dr. Ömer Morgül for his valuable contributions and directions.

I would like to give thanks to my all colleagues, Mustafa Berkant Selek, Yakup Kutlu, Mehmet Emre Çek, Mehmet Ölmez, Aykut Kocaoğlu, and Ömer Karal for their assistance, encouragement and tolerance. In particular, I would like to thank my close friend Yalçın İşler for his friendship and supports. And also, I would like to thank to Funda Gayretoğlu, Fatih Baba and Dr. Kemal Kemahlıoğlu, who are instructors at Ege University Ege Vocational School, for their technical support in the liquid mixing experiment.

My deepest gratitude goes to my wife Kübra and my son Cem Oğuz for their love, trust, tolerance and support during the thesis. And also, I would like to thank my big family members, my parents Yaşar and Ünal Şahin, my sister Burcu Erbil and her husband Mustafa Cenk Erbil, my mother-in-law Aynur Evren and her daughter Burcu Evren for their constant love, patient and support.

Savaş ŞAHİN

NONLINEAR DYNAMICAL STATE FEEDBACK DESIGN FOR TRACKING AND CHAOTIFICATION

ABSTRACT

The thesis develops methods and tools; i) for designing controllers which work well under real-time and real environmental conditions, ii) for designing and updating controllers online directly from the plant's input-output measurement data for tracking problems and iii) for designing controllers modifying the asymptotical behaviors of the plants in order to provide the ability of tracking the chaotic trajectories. One of the main contributions of the thesis is the implementation of a microcontroller based low-cost real-time simulation-emulation platform managed by a graphical user interface for controller design-test-and-redesign. The platform provides a set of novel real-time operating modes as well as the well-known real-time simulation modes. The second main contribution is the design of a novel nonlinear dynamical adaptive control scheme based on the introduced error minimization learning algorithm using input-output measurement data. The third contribution is the development of a new chaotification method based on dynamical state feedback which is valid for any input feedback linearizable nonlinear control system including linear controllable ones as special cases. The proposed chaotification method is demonstrated by experimentation to be very efficient in terms of the consumed energy in liquid mixing actuated by a chaotified DC motor.

Keywords: Real-time simulation, Controller design, Nonlinear control, Adaptive control, Dynamical state feedback control, Chaotification.

YÖRÜNGE İZLEME VE KAOTİKLEŞTİRME İÇİN DOĞRUSAL OLMAYAN DİNAMİK DURUM GERİBESLEME TASARIMI

ÖZ

Tez; i) gerçek zaman ve gerçek çevre şartları altında iyi çalışan denetleyici tasarımı, ii) yörünge izleme problemleri için doğrudan sistem giriş-çıkış ölçüm verilerine dayalı denetleyici tasarımı ve çevrim-içi güncellenmesi ve iii) kaotik yörüngeleri izleme yeteneği kazandırmak için sistemin asimptotik davranışlarını değiştiren denetleyicilerin tasarımı için yöntemler ve araçlar geliştirmiştir. Tezin ana katkılarından birincisi, denetleyici tasarım-test-yeniden-tasarım işlemi için görsel bir kullanıcı ara yüzü tarafından yönetilen mikrodenetleyici tabanlı düşük-maliyetli bir gerçek-zaman benzetim platformu gerçekleştirilmesidir. Platform, iyi bilinen gerçek zamanlı benzetim çalışma biçimlerinin yanında, yeni gerçek zamanlı benzetim çalışma biçimleri de sağlamaktadır. İkinci ana katkı, önerilen hata enazlama öğrenme algoritmasına dayalı olarak giriş-çıkış ölçüm verilerini kullanarak çalışan yeni bir doğrusal olmayan dinamik uyarlamalı denetleyici tasarım yönteminin geliştirilmesidir. Üçüncü ana katkı, doğrusal denetlenebilir olanlar özel durum olmak üzere giriş geri-beslemesi ile doğrusallaştırılabilen herhangi bir doğrusal olmayan sistem için geçerli olan dinamik durum geri-besleme tabanlı yeni bir kaotikleştirme yöntemi geliştirilmesidir. Önerilen kaotikleştirme yönteminin, kaotikleştirilen bir DC motor tarafından sürülen sıvı karıştırma işleminde tüketilen enerji açısından çok verimli olduğu deneysel olarak gösterilmiştir.

Anahtar sözcükler: Gerçek zaman benzetimi, Denetleyici tasarımı, Doğrusal olmayan denetleme, Uyarlanır denetleme, Dinamik durum geri-besleme denetimi, Kaotikleştirme.

CONTENTS

Page

CHAPTER ONE

INTRODUCTION

Control is a problem of finding a suitable control input deriving a given system, say plant, to have a desired behavior (Bolton, 2004; Dorf & Bishop, 2008; Doyle et al., 1992; Goodwin et al., 2001; Mandal, 2006). A general control system briefly consists of a plant, a controller, sensors and actuators as depicted in Figure 1.1. The feedback control system realizations require designing and implementing a controller which produces the necessary control in terms of the feed-backed actual (plant) output and a reference signal representing the desired (plant) output.

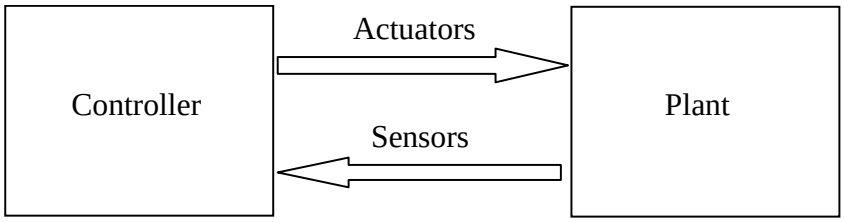


Figure 1.1 General Control System

Regulation, tracking, stabilization and identification are four fundamental problems of control systems. (Chen & Narendra, 2004; Fradkov, 1994; Isidori, 1995; Levin & Narendra, 1993; Narendra & Mukhopadhyay, 1997). Regulation, which is known historically the oldest control problem, is defined as keeping the actual output of the system as constant at a set-point (Chen & Narendra, 2004; Isidori, 1995; Landau & Zito 2006). Tracking is a problem of maintaining the actual output of the control system close to a (reference) trajectory (Boyd & Barratt, 1991; Fradkov & Pogromsky, 1999; Isidori, 1995; Nagrath & Gopal, 2006; Sanner & Slotine, 1992). Stabilization is applied to obtain a stable closed loop dynamics so forcing the system trajectory to an attractor, e.g. to an equilibrium point as in the regulation and to a non-constant trajectory as in the tracking (Isidori, 1995; Isidori & Byrnes, 1990; Khalil, 1996; Levin & Narendra, 1993). Designing a controller to meet the mentioned control goals requires having a reasonable model for the plant which is obtained by system identification. Identification is, indeed, a system representation

problem which might be tackled in many ways such as finding a state model or an input-output relationship fitting the measured input-output data or a model obeying physical laws governing the dynamics (Landau & Zito, 2006; Ljung, 1999).

To tackle the outlined control problems which may be defined in the linear or in the nonlinear settings and also may be subject to the plant parameter variations and noisy environmental conditions, a vast amount of control design methods including adaptive and robust ones is developed in the literature (Abdallah et al., 1991; Ackermann & Blue, 2002; Astrom & Hagglund, 1995; Astrom & Wittenmark, 1994; Doyle, 1983; Dullerud & Paganini, 2000; Hedrick & Girard, 2005; Khalil, 1996; Lewis et al., 2004; Lin, 2007; Sastry & Bodson, 1989; Slotine, 1988; Slotine & Li, 1991; Zhou et al., 1996). However, there is a growing need in developing control design methods which perform, in a way better than the conventional ones, for highly nonlinear dynamical systems under the model and parameter uncertainties about the plant and its environment. One of the common approaches to deal with these modeling difficulties due to changes in environment or aging is to use input-output data based adaptive controller design methods. On the other hand, nonlinear dynamics, in particular, chaotic ones are observed to be useful in some real control engineering applications (Fradkov & Evans, 2005; Ott, 1993; Ottino & Wiggins, 2004; Pecora & Carroll, 1990). So, modifying the asymptotical behavior of a control system to a desired chaotic oscillation, that is the chaotification of the control system, as a preferred solution for certain control applications in contrast to driving systems towards stable equilibrium or, in some cases, limit cycle dynamics provides new potentials to control systems area (Chen, 1999; Chen & Dong, 1993; Ditto et al., 1990; Fradkov & Evans, 2005; Fradkov & Pogromsky, 1999; Morgül & Solak, 1996; Vanecek & Celikovsky, 1994; Wang & Chen, 2000; Wang et al., 2000; Zhang et al., 2004).

In the above control systems perspective, this thesis focuses on the following three issues: i) designing controllers which work well under real-time and real environmental conditions, ii) designing and updating controllers online directly from the plant's input-output measurement data for tracking problems and iii) designing

controllers modifying the asymptotical behaviors of the plants in order to provide the ability of tracking the chaotic trajectories. To address these issues, the thesis develops i) a new real-time simulation/emulation design, test and redesign platform, ii) a new input-output data based nonlinear dynamical adaptive controller design algorithm, and iii) a new chaotification method based on dynamical state feedback which is valid for any input feedback linearizable nonlinear control system including linear controllable ones as special cases. The three main contributions of the thesis are described in the sequel.

Real time design, test and redesign platform: The design of a controller working well for a given real plant in a real environment, which is indeed the ultimate goal of control system design, needs the consideration of the real behavior of the plant under the real operating conditions (Betin et al., 2007; Güvenç & Güvenç, 2002; Keel et al., 2003; Lin, 1997; Mehta & Chiasson, 1998; Pellegrinetti, & Bentsman, 1996; Rodriguez & Emadi, 2007; Yamamoto et al., 2009). Such a controller design problem can be attempted to be solved by examining the simulated controller on the simulated plant under the simulated operating conditions (Boyd, & Barratt 1991; Goodwin et al., 2001) in one extreme case or by testing and tuning the controller hardware on the real plant under the real environment (Astrom and Hagglund, 1995; Ogata, 1994, 1997; Ziegler & Nichols, 1942) in the other extreme case. Testing the proposed controllers' performance on the simulated or real plant is followed by a redesigning or parameter tuning process performed offline or online. Both approaches have its own advantages/disadvantages and also difficulties for experimentation. For most of the cases, examining the controller candidates directly on the real plant may not be possible in the laboratory environment or might be dangerous due to possible damages caused (Bishop, 2008; Zeigler & Kim, 1993). On the other hand, mimicking the real plant in real-time and in real environmental conditions especially together with its analog/digital interface units is not only complicated in software simulations but also it, with a great possibility, yields unreliable simulators which are highly sensitive on the unavoidable modeling errors occurred at each simulated unit (Bacic, 2005; Maclay, 1997). The developed Controller-Design-Test-Redesign-Platform (CDTRP) which consists of a simulator

together with a software manager in PC, i.e. Graphical User Interface (GUI), microcontroller based emulator and a hardware peripheral unit is intended to have the advantages of using simulated or emulated plants in the controller design and also of testing the candidate (simulated, emulated or real) controllers on the almost real operating conditions. The CDTRP platform is developed according to a controller design-test-redesign methodology which can be stated as to provide a high level of flexibility of choosing and testing controller and plant models from a wide variety by the simulator unit in the early stages of the controller design and then to create environmental conditions as close as possible to the real world by the emulator and peripheral units in the final stage of the controller design process.

Simulation of control systems is preferred, in general, i) for understanding the behavior of the plant together with its actuator and sensory devices based on their identified models obtained beforehand, i.e. for the analysis, and ii) for testing the designed controllers if the design specification are met, i.e. for the synthesis. In the controller synthesis case which is the main concern in this chapter, testing the controllers is followed by a redesign and/or tuning procedure. Depending on the implementation of the controller, plant and peripheral unit as simulation, emulation or real hardware, the developed platform CDTRP can be operated in 24 different real time operation modes (See Tables 3.1-3.) which are also called as real time simulation modes meaning the same thing throughout this thesis. The simulator and emulator in all of the 24 modes of CDTRP are designed for performing real-time simulations, however they can be run faster or slower for different purposes; e.g. fast running plant emulator or simulator can be used for model reference adaptive control. So, the simulation modes that can be realized in the platform are not restricted to the mentioned 24 real time modes. Some of the real time simulation modes correspond to the well-known “hardware-in-the-loop simulation” (Dufour et al., 2007; Facchinetti & Mauri, 2009; Hanselmann, 1996; Isermann et al., 1999; Li et al., 2006; Lu et al., 2007; Steurer et al., 2009) where the controller is realized as hardware and the plant is implemented in the PC or in the emulator (See Tables 3.1-3.). On the other hand, some of the other modes correspond to the well-known “control prototyping” and “software-in-the-loop simulation” (Isermann et al., 1999)

both simulate the controller in the PC but differ from each other in the plant part; the first one is the real and the second is implemented in the PC (See Tables 3.1-3.).

CDTRP can be used i) for the design-test-redesign of the controllers under the framework of a chosen specific controller design method, such as an adaptive or robust method or else, ii) for comparison of the performances of the different controller design methods, techniques and algorithms on the simulated, emulated or real plants with the emphasis of the controller design in real-time and real environment, so supporting the selection of the best controller for a specific applications, on the other hand serving as a test-bed for researchers in examining their immature controller design method in its development phase, iii) for verification and validation of a plant-model (Balcı, 2003; Özer et al., 2004; Sargent, 2004; Smith & Doyle, 1992) based on the simulated and emulated plants with the emphasis running in the real-time and in the real environment, iv) for controller design requiring a parameter training procedure based on the measurements and also calculations on an emulated (identified) plant model as in done artificial neural networks based controller design methods (Fukuda & Shibata, 1992; Li et al., 2006; Narendra, 1996; Spooner et al., 2002; Suykens et al, 1996), and v) for low cost real time implementation of control systems based on the benchmark plants which are of educational value but hard or impossible to be realized in an educational laboratory (Şahin et al., 2009).

Similar real-time simulation platforms have been realized in the literature (Betin et al., 2007; Facchinetti & Mauri, 2009; Mehta & Chiasson, 1998; Rodriguez & Emadi, 2007; Tarte et al., 2006; Wang et al., 2000), however; i) they are not dedicated to be a general purpose design-test-redesign controller platform, ii) they are restricted either to a specific control application, e.g. robot, specific electrical motors, pantograph or dynamometer or to a certain type simulation mode such as software-in-the-loop or hardware-in-the-loop, iii) none of them possesses a (real world) hardware peripheral unit which comprises all of the (real) analog and digital actuators, sensory devices, the external disturbance, parameter perturbation signal derivers and analog/digital controller hardware components, so they have the ability

to recreate a real environment in a more restricted sense than CDTRP, iv) the frequency limits for the real time simulation modes realized by these platforms have not been reported in contrast to the CDTRP, and v) they do not have such a GUI unit capable of monitoring and controlling of the overall simulation platform that the CDTRP possesses.

Learning algorithms for adaptive nonlinear dynamical controller design: Adaptive control is an attractive area due to its capability of producing efficient solutions to nonlinear control problems. Flight control, motor control, and process control are among the numerous applications to be mentioned (Astolfi et al., 2008; Aseltine et al., 1958; Astrom & Wittenmark, 1994, 1997; Blanchini et al., 2009; Ioannou & Sun, 1996; Khalil, 1996; Kokotovic, 1992; Krener, 2003; Krstic et al., 1992; Krstic et al., 1994; Lavretsky, 2009; Lewis et al., 2004; Narendra & Annaswamy, 1989; Narendra & Venkataraman, 1995; Pan & Başar, 1998; Salomonsson et al, 2008; Sastry, 1999; Sastry & Bodson, 1989; Sastry & Isidori, 1989; Seto et al., 1994; Tang et al., 2009; Wu et al., 2007).

The adaptive control is a control method where the controller parameters are changed in an online fashion according to the changes in the plant and/or environment. Adaptive control methods can be categorized into two groups: i) direct methods and ii) indirect methods (Astrom, 1987; Astrom & Wittenmark, 1994; Data & Ioannou, 1994; Guo & Chen, 1991; Krstic et al., 1994; Middleton et al., 1988; Narendra & Kudva, 1974; Narendra & Venkataraman, 1995; Sastry & Bodson, 1989; Slotine & Li, 1991; Wittenmark, 1995). Controller parameters in the direct methods are changed as a function of the output of the plant. Gain scheduling and Model Reference Adaptive Control (MRAS) are examples for direct control. In MRAS, there is also a reference model so that the controller parameters are changed as a function of its output which is, in fact, desired to be tracked by the control system in an adaptive way. On the other hand, the controller parameters in the indirect methods are changed in accordance with the plant parameters which are estimated by certain techniques. Self tuning regulator and dual control are examples for indirect methods.

In any kind of adaptive control, the controller becomes nonlinear and/or time-varying even if the plant and the chosen model for controller are linear and time-invariant. An adaptive control system has two loops: i) the feedback loop ii) the controller parameter calculation loop. In the latter loop, the controller parameters are updated in accordance with plant output, controller output and reference input (Astrom, 1987; Astrom & Wittenmark, 1980, 1994, 1997; Karacan et al., 1997; Kosmatopoulos, 2008; Krstic et al., 1994; Lee et al., 2001; Middleton et al., 1988; Narendra & Balakrishnan, 1994; Narendra & Kudva, 1974; Narendra & Venkataraman, 1995; Sastry & Bodson, 1989; Sen & Pena, 1997; Slotine & Li, 1991; Pan & Başar, 1998; Tang et al., 2009; Wang & Lee, 1988; Widrow et al., 1993; Widrow & Plett, 1996; Wu et al., 2007; Yang & Huang, 1992).

In this thesis, a new input-output data based nonlinear dynamical adaptive controller design method is developed. The developed adaptive control algorithm, which employs ARMA and NARMA input-output models both for plant and the closed-loop system consisting of plant and controller, is suitable to run online based on measurement data. In the linear case, it can be viewed as an algorithm solving Diophantine equation in real-time using data measured from the plant not a model of the plant (Astrom, 1987; Astrom & Wittenmark, 1994). The proposed learning algorithm for adaptive control has the possibility of implementing it as an Artificial Neural Network (ANN) choosing appropriate basis functions in NARMA models (Chen & Narendra, 2001, 2003; Ge et al., 1999; Ge & Wang, 2002; Huang et al., 2007; Levin & Narendra, 1996; Lin & Shen, 2006; Sanner & Slotine, 1992; Zhang et al., 1999). As opposed to the inverse system based ANN controllers, it attempts to find a closed loop system to possess a desired behavior rather than attempting to find an inverse of the plant yielding a unity closed loop system.

The developed adaptive control scheme defines a kind of Model Reference Adaptive Control (MRAC) when the desired output of the plant is provided by a stable reference model and the controller parameters are updated directly based on the measured plant outputs in real-time without taking into account previous measurements. On the other hand, it defines a self-tuning adaptive control when the

measured plant input-outputs within a chosen time window are first used for identifying a plant model and then for updating the controller parameters not at each time but at the sampled times with sampling period not less than the window length used for identification.

Considering wide interest in Proportional Integral Derivative (PID) controllers, a special version of the introduced adaptive control method is also developed for determining and/or tuning PID parameters. The developed adaptive nonlinear dynamical controllers, in particular PID controller, are designed and tested in the CDTRP and also is applied for controlling a real DC motor. In the literature, finding PID parameters (Astrom & Hagglund, 2004; Baek & Kuc, 1997; Benaskeur & Desbiens, 2002; Cao et al., 2008; Galotto et al., 2007; Liu & Daley, 2000; Omatu & Yoshioka, 1997; Skoczowski et al., 2005; Toscano, 2005; Yamamoto et al., 2009) and controlling real DC motor (Baek & Kuc, 1997; Cao et al., 2008; Mehta & Chiasson, 1998; Salomonsson et al, 2008) issues are still attractive areas.

The developed adaptive control scheme is suitable to be improved by introducing robustifying mechanism into the controller parameter learning process and also into the plant parameter identification subroutine, so handling nonlinearities and model/parameter uncertainties in specific control problems.

Dynamical state feedback chaotification of input state linearizable systems: Most of the researches on deterministic chaotic systems are devoted to the analysis and implementation of a set of well-known chaotic systems and their variants (Chua et al., 1993; Lorenz, 1963; Lü & Chen, 2002; Ott et al., 1990; Sprott, 2000). On the other hand, there is a growing interest on the real world applications of chaotic systems as seeking an answer to the question how to get benefit of random like yet deterministic complex behavior of chaotic dynamical systems which can be produced even within a very simple system structure. Secure communication, encryption, pseudorandom generation numbers (Monte Carlo method), information and signal processing, chaotic liquid mixing are among the potential applications in this direction of chaos research (Burghelea et al., 2004; Fradkov & Evans, 2005;

Kavaslar & Güzeliş, 1995; Leung & Qin, 2001; Ott, 1993; Ottino & Wiggins, 2004; Paul et al., 2004; Pecora & Carroll, 1990; Savaci et al., 2003; Yaowen et al., 2000). In such applications, in contrast to the usual chaos control methods driving the chaotic system to a limit cycle or to an asymptotically stable equilibrium (Ott et al., 1990), chaos is a desired property such that the chaotic behavior is tried to be sustained for originally chaotic systems or created by a chaotifying control (called also as anti-control or chaotization) applied to originally non-chaotic systems. Synchronization of the chaotic systems is one of the main issues investigated largely in the first class of chaos applications where the well known chaotic systems are employed for obtaining necessary chaotic signals (Cuomo et al, 1993; Kavaslar & Güzeliş, 1995; Kocarev & Parlitz, 1995; Morgül, 2003; Morgül & Solak, 1996; Pecora & Carroll, 1990; Rosenblum et al., 1996; Savaci et al., 2003). For the second class of chaos applications, the chaotification is achieved either by periodic excitation of the systems in a feed-forward way or by feedback control exploiting time-delay or static nonlinearity in the feedback path (Chen, 1999; Chen & Dong, 1993; Ditto et al., 1990; Fradkov & Evans, 2005; Fradkov & Pogromsky, 1999; Morgül & Solak, 1996; Sinha et al., 2000; Soong & Huang, 2007; Vanecek & Celikovsky, 1994; Wang & Chen, 2000; Wang et al., 2000; Zhang et al., 2004).

This thesis introduces a novel chaotification method which employs a suitable dynamical state feedback to the system under consideration to match a part of its dynamics, in fact the last row of its state equation system in the Brunovsky canonical form, to a part of a chaotic reference model. The developed method is indeed the extension of the model based static feedback chaotification method proposed in (Morgül & Solak, 1996) to the dynamical feedback case. The developed dynamical state feedback chaotification method has the following features distinguishing it from the other chaotification methods: i) It can chaotify any input-state linearizable and observable system, ii) Any chaotic system of arbitrary dimension can be used as the reference model with no need to transform it into a special form, so providing the advantage of exploiting the vast amount of information on chaotic systems and their implementations available in the literature, iii) It is a state feedback control scheme which requiring, in general, that all states are available either by direct measurement

or by estimation, iv) It is a dynamical feedback scheme that augments the degree of the original system at most by one below the degree of the reference chaotic system, i.e. by two for a third order reference chaotic system and v) It admits the matching between the last state equation of the system to be chaotified and any of the state equations of the reference chaotic system as a valid matching, so possessing a large set of implementation alternatives from which the most efficient one can be chosen.

To demonstrate its potential in real world applications, the dynamical state feedback chaotification is applied to a permanent magnet DC motor as matching the closed loop dynamics to the well known Chua's chaotic circuit. The chaotified DC motor, on which an impeller is mounted, is then used as a tool for mixing liquids to reach homogenous mixture under less energy consumption and/or within less time consumption as compared to the conventional constant or periodical speed mixing. Corn syrup added acid-base reaction (Ascanio et al., 2002) is considered as a benchmark test for liquid mixing in order to compare the mixing performances of different DC motor speed modes, i.e. constant, periodical and chaotic. It is observed in a non-intrusive way that the chaotic mode for DC motor speed provides much more efficient mixing as compare to the constant and also periodical motor speed modes in terms of the neutralization time under the same power consumption.

The liquid mixing problem is considered in the thesis as the application area for the developed chaotification method because of its importance in a diverse application area including chemical, petroleum, food and pharmaceutical industries (Paul et al., 2004). Energy and time efficient mixing is a very attractive issue in these industrial sectors since the annual cost which might be saved by efficient industrial mixing is estimated for US as much as ten billion dollars (Harnby et al., 1992). In the last decade, several chaotic mixing methods for obtaining efficient mixing are proposed and their superiority to constant and periodical mixing is reported (Alvarez-Hernández et al., 2002; Takigawa et al., 2000; Ye & Chau, 2007; Zhang & Chen, 2005). In a part of these methods, a chaotic signal produced, for instance, by a Chua's circuit or Lorenz system is used for driving a DC motor which actuates either an impeller mounted on its shaft or the tank containing the liquid. This kind of

chaotic mixing methods is of open loop chaotic mixing where the control input is chaotic but there is no guarantee that the mixing system, herein the impeller or tank actuated by DC motor, has chaotic motions (Zhang & Chen, 2005). In the closed loop chaotic mixing methods including the one proposed in this thesis, the DC motor speed is feed backed, so there is a possibility of applying a suitable control algorithm for the mixing system to track a chaotic motion. It is observed in the realized experiments that the developed closed loop chaotic mixing method employing a nonlinear dynamical state feedback provides more efficient mixing as compare to the available methods including the one proposed in (Ye & Chau, 2007) which exploits the time-delay feedback type chaotification of (Wang et al., 2000).

The organization of the chapters of this thesis is as follows. Chapter 2 gives a background on control systems modeling; input-output and state space representations; stability, controllability and observability; feedback linearization; artificial neural networks based controller design and adaptive control. In Chapter 3, the developed real-time simulation/emulation design, test and redesign platform is explained in details together with some benchmark implementations. The operating modes of the real-time simulations are stated in a comparative way in an introduced taxonomy. The proposed design, test and redesign procedure for controller system design is also given in this section. Chapter 4 presents the developed adaptive control scheme which is based on the input-output data in the linear and nonlinear settings. Chapter 4 also presents an adaptive PID control algorithm as a special case of the general nonlinear dynamical adaptive control algorithm. Chapter 5 explains the introduced dynamical state feedback chaotification method as an extension of the reference chaotic system based static state feedback chaotification method available in the literature. Chapter 5 gives also the experimental results on chaotic liquid mixing actuated by a chaotified DC motor. Conclusions and future directions in the context of the research realized in the thesis studies are outlined in Chapter 6.

CHAPTER TWO

BACKGROUND ON CONTROL SYSTEMS MODELING, ANALYSIS AND DESIGN

In this chapter, a brief background on control systems modeling, analysis and design concepts are introduced. These concepts, including black box and state space representations, qualitative analysis of control systems and controller design methods, will be used in the subsequent chapters.

2.1 Control Systems Modeling

Control systems modeling is a problem of finding an appropriate mathematical representation for a given plant desired to be controlled (Astrom & Murray, 2008; Cassandras & Lafortune, 2008; Dorf & Bishop, 2008; Hellerstein et al., 2004). This representation, say model, is a key element for designing a controller working well for a real plant under real environmental conditions. So, modeling constitutes one of the most important theoretical and practical issues in control area (Rojas et al, 2007; Chen & Narendra, 2004; Loh & Lu 2002; Narendra & Lewis, 2001).

Modeling can be done mainly in two different ways: The first approach employs physical laws, such as Newton's laws and Kirchoff's laws, to derive an internal representation, usually constituting a set of differential equations, for the plant under consideration. The second one uses the input-output measurement data to identify a plant model and then to identify its parameters. These two stages are called as model and, respectively, parameter identification; both as system identification. In both of the approaches, modeling aims to find a simple yet enough representation to describe the qualitative and quantitative properties of plants in an efficient way.

As will be explained in Subsection 2.1.2, system identification can be based on an internal representation, for instance, a state model which consists of n first order differential equations defining derivatives of the states in terms of the states and

inputs and algebraic equations defining outputs in terms of states and inputs. In this case, the identification turns out first finding suitable expressions for nonlinearities intrinsic to the relations among the state variables, inputs and outputs of the system and then determining parameters defining these relations. The resulting state space representation not only provides a basis for designing controller but also for qualitative analyses including Lyapunov stability, controllability and observability of the considered system.

As explained in Subsection 2.1.1, system identification can also be realized based on a black box representation which neither needs knowledge about the inside of the plant nor tries to build up an internal representation for the plant. In contrast to state models, black box models do not allow certain qualitative analyses such as Lyapunov stability but they are easier to be obtained in an efficient way with high accuracies from the input-output measurement data, they usually constitute efficient models and, in some cases, they provide a unique choice in modeling a system under several uncertainties.

Although stochastic models can also be used in control systems modeling, only deterministic models are considered throughout this thesis work.

2.1.1 Black Box Representation

A black box model shown in Figure 2.1 can be constructed first by considering a general purpose input-output model which is supposed to fit the input-output data and then determining model parameters from measured input-output data using a suitable algorithm (Ljung, 1999; Pintelon & Schoukens, 2001, Sjoberg et al., 1995).

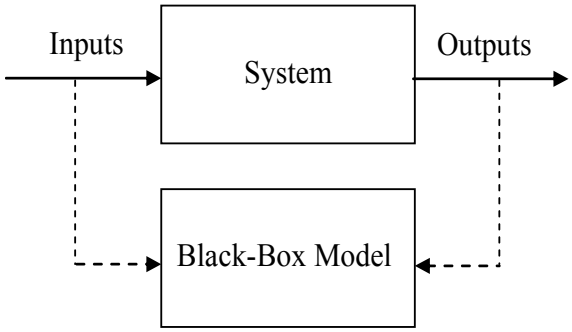


Figure 2.1 Black-box modeling

Any black box model provides a relation among the input and outputs of the system. The relation can be, in the most general case, an implicit one where the outputs can not be determined as functions of inputs. The relation is rarely algebraic but usually a dynamical relation defining outputs at a specific time instant as explicit functions of the past outputs as well as the current and past inputs. Such relations are usually called as Auto Regressive Moving Average (ARMA) or Nonlinear Auto Regressive Moving Average (NARMA) depending on the existence of nonlinear dependence among the input and/or output variables. As will be cleared by their definitions in Equation 2.1 and Equation 2.2 of Subsection 2.1.1.1, the auto-regressive term recalls the dependence of the current output on the past outputs and the moving average recalls that the current output is determined also by a weighted average of the current and finite number of past inputs taken place in a finite duration moving window.

2.1.1.1 ARMA-NARMA Models

Although continuous-time and time-varying versions are also possible to be defined, discrete-time time-invariant versions will be considered in this thesis due to their convenience to handle input-output measurement data and due to the emphasis of the thesis focused on time-invariant systems. In a discrete-time (time-invariant) NARMA model, the current output is given, in the most general case, as a nonlinear function of N past outputs and the current and M past inputs as in Equation 2.1.

$$y(k) = \mathrm{H}[y(k-1), y(k-2), \cdots, y(k-N); u(k), u(k-1), \cdots, u(k-M)] \tag{2.1}$$

Where, $\mathrm{H}(\circ) : R^{N+M+1} \to R$ is a nonlinear algebraic function. If the system has more than one input, the current and past values of these inputs also would be added into the inside of the bracket at the right hand side of the relation in Equation 2.1. If the system has more than one output, then for each output there should have a relation as in Equation 2.1.

In the linear case, NARMA model in Equation 2.1 takes the following (ARMA) form where the first sum corresponds to AR part while the second to MA part (Cabrera & Narendra, 1999; Levin & Narendra, 1996; Yegnanarayana, 1981).

$$y(k) = \sum_{i=1}^{N} \alpha_i y(k-i) + \sum_{j=0}^{M} \beta_j u(k-j) \tag{2.2}$$

Where, $\alpha_i \in R$ and $\beta_j \in R$ with $i = 1, \cdots, N$ and $j = 0,1, \cdots, M$ are the linear weights, so called AR and MA parameters, respectively.

NARMA and especially ARMA models are widely used as plant models in control systems identification and as controller models in control system design, and also employed in a diverse area such as prediction models for time series analysis in many fields including weather forecast, for speech coding and recognition in communication and for feature extraction for instance in biomedical signal processing (Brown, 2004; Chen & Billings, 1989; Chen & Narendra, 2003; Hyvärinen & Oja, 1997; Narendra & Mukhopadhyay, 1997; Rank, 2003; Wang et al., 2003; Zhan & Jay-Kuo, 2001).

The NARMA model in Equation 2.1 is quite general covering a large class of nonlinear systems. However, there is a need to introduce special NARMA forms suited to specific applications, i.e. the simpler NARMA models bringing analysis and design efficiency yet having sufficient generality to model the input-output behavior

of the system. One special case is given in Figure 2.3 supposing that the effects of the past outputs and inputs are integrated in an additive way.

$$y(k) = F[y(k-1), y(k-2), \cdots, y(k-N)] + G[u(k), u(k-1), \cdots, u(k-M)] \quad (2.3)$$

Simpler but still powerful NARMA architectures which can be exploited in specific applications are outlined range from Equation 2.4 to Equation 2.11.

$$y(k) = H[\sum_{i=1}^{N} \alpha_i y(k-i) + \sum_{j=0}^{M} \beta_j u(k-j)] \quad (2.4)$$

$$y(k) = F[\sum_{i=1}^{N} \alpha_i y(k-i)] + G[\sum_{j=0}^{M} \beta_j u(k-j)] \quad (2.5)$$

$$y(k) = F[\sum_{i=1}^{N} \alpha_i y(k-i)] + \sum_{j=0}^{M} \beta_j u(k-j) \quad (2.6)$$

$$y(k) = F[\sum_{i=1}^{N} \alpha_i y(k-i)] + \beta u(k) \quad (2.7)$$

$$y(k) = \sum_{i=1}^{N} \alpha_i y(k-i) + G[\sum_{j=0}^{M} \beta_j u(k-j)] \quad (2.8)$$

$$y(k) = G[\sum_{j=0}^{M} \beta_j u(k-j)] \quad (2.9)$$

$$y(k) = F[y(k-1), y(k-2), \cdots, y(k-N)] + G[y(k-1), y(k-2), \cdots, y(k-N)] \cdot u(k) \quad (2.10)$$

$$y(k) = \sum_{i=1}^{I} F_i[y(k-1), y(k-2), \ldots, y(k-N)] + \sum_{j=0}^{J} G_j[y(k-1), y(k-2), \ldots, y(k-N)] \cdot u(k) \quad (2.11)$$

Where, $H(\circ): R \to R$, $F(\circ): R^N \to R$, $G(\circ): R^{M+1} \to R$, $F_i(\circ): R^N \to R$ and $G_j(\circ): R^M \to R$ with $i = 1,2,\cdots,I$ and $j = 0,1,\cdots,J$ are nonlinear (algebraic) functions. And, $\alpha_i \in R$, $\beta_j \in R$ and $\beta \in R$ with $i = 1,2,\cdots,N$ and $j = 0,1,\cdots,M$. The NARMA models in Equation 2.6 and Equation 2.10 are introduced in this thesis as the models for which the nonlinear version of the data dependent controller design algorithm developed in this thesis can be applicable.

Discrete-time ARMA models are very advantageous with respect to design and implementation issues. Their parameters can be found by linear analysis techniques such as using a least square approach. It can be implemented in the structure of Figure 2.2 just by a linear weighted summing unit and by time delay units (z^{-1}'s).

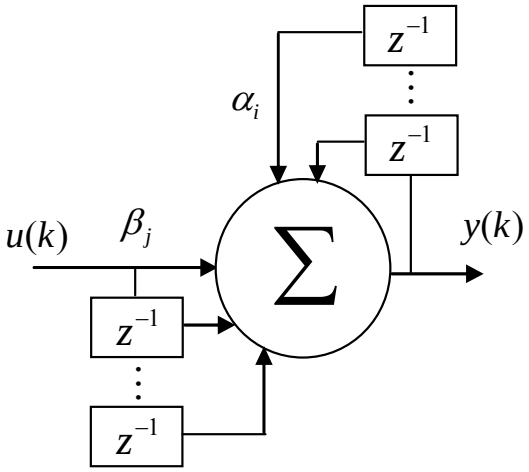


Figure 2.2 Architecture of an ARMA model

On the other hand, for NARMA models, nonlinearities can be handled in many ways. Universal function expansions/approximations based on specific sets of basis functions and uniform architectures are quite common to tackle this nonlinear function representation problem. Artificial Neural Networks serve efficient solutions to such nonlinear function approximation/representation issues.

2.1.1.2 Artificial Neural Networks

Artificial Neural Networks (ANNs) are widely used in control systems area since 1980s. Since ANNs define, in general, a nonlinear algebraic function, they can cope with nonlinearities inherent in control systems possessing complex dynamics. On the other hand, ANNs are universal function approximators which are capable of approximating to any continuous function in a compact set within arbitrary degree of accuracy (Cybenko, 1989). Due to its parallel architecture, they are fault tolerant. There are many efficient ANN architectures and many associated efficient learning algorithms for designing them by a finite set of training data with providing a powerful generalization ability of responding well for the test data not learned before. ANNs can learn in supervised or unsupervised ways depending on the availability of data class labels, in a more general setting, say desired outputs. The information is coded in the connection weights associated to the pairs of neurons which are the functional units of the ANN. In any kind of learning, learning is defined as an optimization problem and is accomplished by a changing rule for connection weights minimizing the cost of the optimization problem which is, for instance, the difference between desired and actual outputs for supervised learning cases.

As in the general ANN literature, the mostly widely used ANN model in identification and control is the Multi Layer Perceptron (MLP) due to its function approximation capability and the existence of an efficient learning algorithm, so called back-propagation, associated to it (Ahmed, 2000; Lightbody & Irwin, 1995; Limanond & Si, 1998; Meireles et al., 2003; Noriega & Wang, 1998; Omidvar & Elliott, 1997). MLP is a multilayer, algebraic neural network of neurons, called as perceptrons, which are multi-input, single-output functional units taking firstly a weighted sum of their inputs and then pass it through a sigmoidal nonlinearity to produce its output (See Figure 2.4.). As shown in Figure 2.3, a multi-input, multi-output MLP with one hidden layer can be used as a NARMA model.

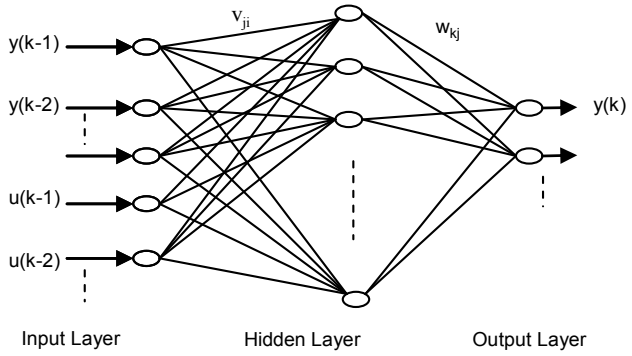


Figure 2.3 MLP implementing NARMA model

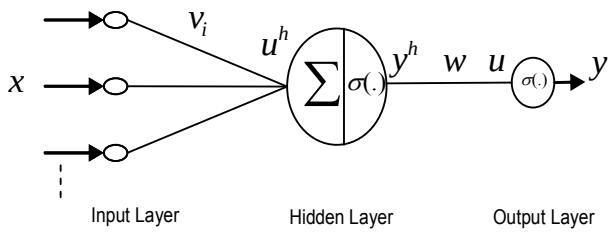


Figure 2.4 Perceptron as a hidden neuron

MLP is usually designed, in a supervised way, by determining connection weights v and w using the celebrated error Back-Propagation (BP) algorithm which is indeed a gradient descent technique used for finding an acceptable local minimum of the squared error in Equation 2.12 between the desired and actual outputs.

$$\varepsilon = \frac{1}{2}(r-y)^2 \tag{2.12}$$

Where, $e = r - y$ represents the error for a unique data sample. BP calculates the partial derivatives of the output error in Equation 2.12 with respect to the connection weights by employing chain rule as shown in Equation 2.13 and 2.14

$$\frac{\partial \varepsilon}{\partial w} = \frac{\partial \varepsilon}{\partial y}\frac{\partial y}{\partial u}\frac{\partial u}{\partial w} = -(r-y)\sigma'(u)y^h \tag{2.13}$$

$$\frac{\partial \varepsilon}{\partial v_i} = \frac{\partial \varepsilon}{\partial y}\frac{\partial y}{\partial u}\frac{\partial u}{\partial y^h}\frac{\partial y^h}{\partial u^h}\frac{\partial u^h}{\partial v_i} = -(r-y)\sigma'(u)w\sigma'(u^h)x \tag{2.14}$$

Where, $\sigma'(\circ)$ denotes the derivative of the sigmoidal nonlinearity which can be calculated without differentiation just in terms of the sigmoidal function itself for the tangent hyperbolic sigmoid case. The partial derivatives calculated are then used for updating the connection weights in the opposite of the gradient direction towards one of the local minima with a sufficiently small step size ζ, called also as learning rate:

$$w(k+1) = w(k) - \zeta\frac{\partial \varepsilon}{\partial w(k)} = w(k) + \zeta(r-y)\sigma'(u)y^h \tag{2.15}$$

$$v_i(k+1) = v_i(k) - \zeta\frac{\partial \varepsilon}{\partial v(k)} = v(k) + \zeta(r-y)\sigma'(u)w\sigma'(u^h)x \tag{2.16}$$

If the updates are implemented according to the above recursions allowing the connection weights to be changed for each sample, then BP is called as pattern mode or data mode BP. Otherwise, it is called as batch mode or group mode allowing an update for the whole set of training samples once at each time instant which requires summing up the individual gradients obtained for each specific sample to take a step. The other ANN models have their own architectures and learning algorithms possessing advantageous in one hand and disadvantageous in the other, so one can choose one of them depending on the nonlinearities and the measurement data properties intrinsic to the considered control system. Gradient based learning algorithms for those ANNs have the same structure with the one given in Equations 2.15 and 2.16 but with other basis functions different than the above sigmoidal ones, for instance Gaussian functions as in Radial Basis Function Networks (RBFNs) (Mao, 2002; Park, et al., 2002; Selver & Güzeliş, 2009; Uykan et al., 2000) and with different connection topologies, requiring some modifications in the learning algorithms. Moreover, some ANNs, for instance RBFNs, may employ hybrid learning algorithms, i.e. they use an unsupervised method, e.g. clustering, for learning the parameters of the first layer neurons and a gradient based supervised

method for learning the parameters of output layer neurons (Mao, 2002; Park, et al., 2002; Selver & Güzeliş, 2009; Uykan et al., 2000).

Especially due to their nonlinear nature, learning and generalization abilities, ANNs are widely preferred in nonlinear, adaptive and robust identification and control (Chen & Narendra, 2004; Lewis et al., 1995; Narendra, 1993; Narendra & Parthasarathy, 1990; Poggio & Girosi, 1990a, 1990b; Yeşildirek & Lewis, 1994).

2.1.1.3 Artificial Neural Network for Identification

In control systems literature, there are two kinds of the ANN based identification model: Parallel and series–parallel. The parallel model is shown in Figure 2.5; the ANN identification model is fed by the model outputs and plant inputs yielding the ARMA model in Equation 2.10 for a linear ANN (Narendra & Parthasarathy, 1990).

$$\hat{y}(k) = \sum_{i=1}^{N} \alpha_i \hat{y}(k-i) + \sum_{j=0}^{M} \beta_j u(k-j) \tag{2.17}$$

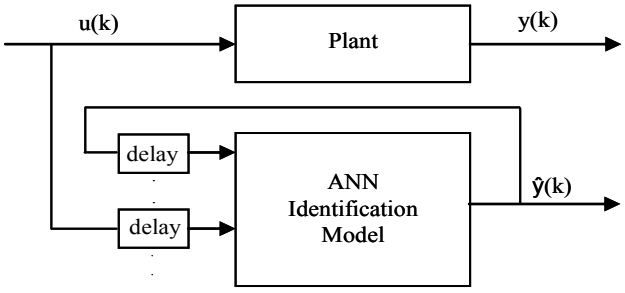


Figure 2.5 ANN based parallel identification model

On the other hand, the ANN based series-parallel model in Figure 2.6 is fed by the inputs and also the outputs of the plant yields the ARMA model for a linear ANN (Narendra & Parthasarathy, 1990).

$$\hat{y}(k) = \sum_{i=1}^{N} \alpha_i y(k-i) + \sum_{j=0}^{M} \beta_j u(k-j) \tag{2.18}$$

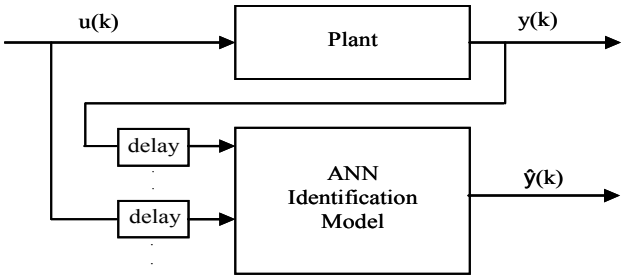


Figure 2.6 ANN based series-parallel identification model

Series-parallel identification model is widely used in the control literature (Barreto & Araujo, 2004; Narendra & Parthasarathy, 1990) because it does need the initial condition of the plant to be identical of the model. Both parallel and series parallel identification approaches define the following NARMA model in the most general case.

$$y(k) = \mathrm{H}[y(k-1), y(k-2), \cdots, y(k-N); u(k), u(k-1), \cdots, u(k-M)] \qquad (2.19)$$

2.1.1.4 Artificial Neural Networks Based Controllers

As a consequence of their capabilities in handling nonlinearities and uncertainties, and also of their adaptation abilities crucial for changing plants and/or environments, ANNs are used as controllers with a growing acceptance since 1980s (Narendra 1991, 1996; Astrom & Hagglund, 1995; Kawato et al., 1987; Kwan et al., 1998; Lewis, 1996; Lewis et al., 1999; Lewis et al., 2004; Narendra & Parthasarathy, 1990; Nguyen & Widrow, 1990; Psaltis et al., 1988; Seong & Widrow, 2001a, 2001b; Widrow & Bilello, 1993; Werbos, 1991).

ANN based controller design methods can be categorized into two groups: i) feed-forward ANNs (can also be said as algebraic ANNs.) and ii) recurrent ANNs (can also be said as dynamical ANNs.). The feed-forward ANNs define static mappings. The inputs in the feed-forward ANN based controllers are fed by the current and/or

past input-output data pairs of the plant. On the other hand, the recurrent ANNs define dynamic mappings. The inputs of the recurrent ANNs are chosen similar to the feed-forward ANNs. However, in addition to these inputs, the outputs of the ANN are also used as extra inputs in recurrent ANNs (Ku & Lee, 1995; Narendra & Parthasarathy, 1990; Sundareshan & Condarcure, 1998; Suykens et al., 2000).

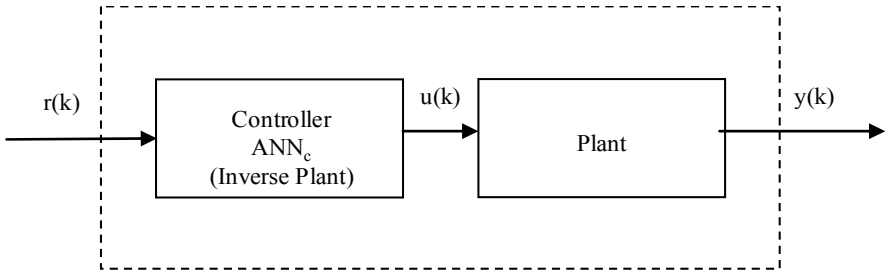


Figure 2.7 Inverse system based ANN controller

The basic step of designing a controller is to find an optimum control signal for the system to be controlled. This objective can be accomplished by choosing the inverse of the plant as the controller such that the overall system becomes a unity system as shown in Figure 2.7.

Psaltis et al. (1988) and also Narendra (1996) classify ANN based control appeared in the literature into four main groups according to their structure as: i) direct inverse control, ii) feed-forward inverse control, iii) specialized learning control and iv) internal model control. The basic issue with all of these inverse system based methods is on the assumption of the existence of the inverse plant which is not always true.

Direct inverse control: Direct inverse control is based on two identical ANN controllers in Figure 2.8. The first one is used to produce the control input $u(k)$ from the desired output (i.e. the reference input.) $r(k)$. The other one is used to produce a signal $\hat{u}(k)$ to match the control input $u(k)$ from the actual output $y(k)$. Each of the identical controllers becomes the inverse of the plant whenever the error $e(k) = u(k) - \hat{u}(k)$ is zero.

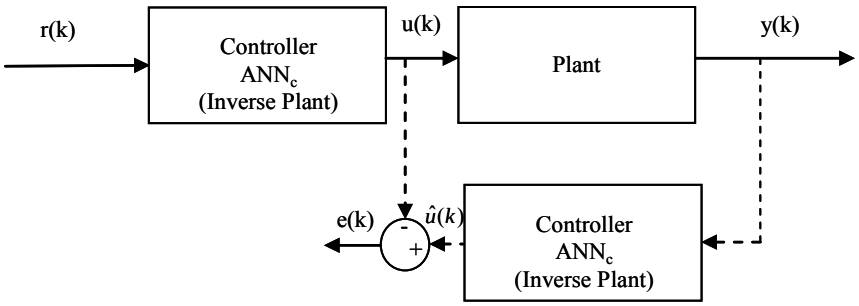


Figure 2.8 Training stage of direct inverse control

In fact, minimization of this error is, in general, not equivalent to the minimization of the output error defined as the difference between the actual $y(k)$ and desired outputs $r(k)$. This may cause erroneous results in practice.

Feed-forward inverse control: Feed-forward inverse control requires, in the training stages, two different ANNs as in the direct inverse control. As can be seen in Figure 2.9, one of the ANNs is used for identification and the other for the inverse of the plant. The ANN used for identification, i.e. ANN_P, is trained by measured input-output data pairs of the plant not in the reverse direction but in the forward direction. Training of the inverse system controller ANN_C is realized based on the minimization of plant's output error, for instance, by BP algorithm as holding fixed the connection weights of ANN_P already determined by the identification procedure.

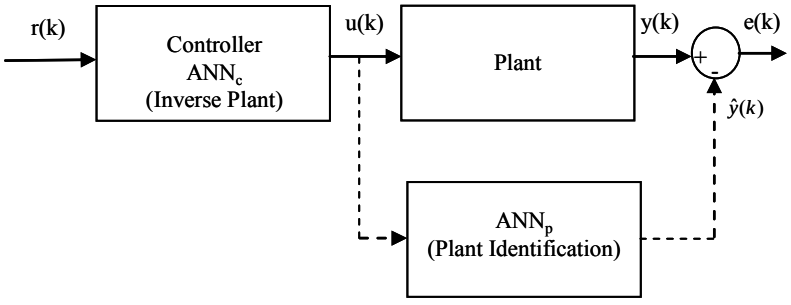


Figure 2.9 Training stage of feed-forward inverse control

Specialized learning control: Specialized learning control was proposed by Psaltis et al. (1988). It requires the knowledge of the Jacobian of the real plant in order to update weights of the ANN controller. This method is similar to the feed-forward

inverse control. The Jacobian of the real plant is used in specialized learning control whereas the Jacobian of the identified plant model ANN_P is used in feed-forward inverse control. Moreover, the connection weights of ANN_P, i.e. the controller parameters are updated online, so providing an adaptive control system depicted in Figure 2.10.

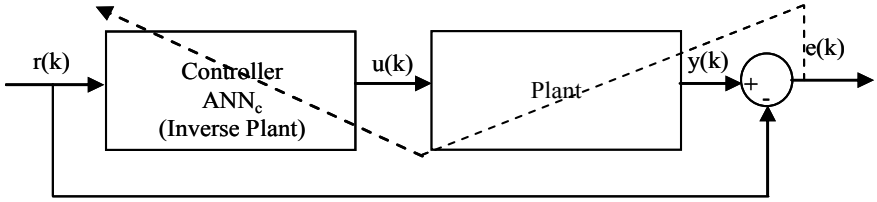


Figure 2.10 Specialized learning control

Internal model control: Internal model controller is widely used for chemical processes. The training of the ANN controller weights are implemented, in an off-line fashion, through the cascade of ANN controller and ANN identification model as in the feed-forward inverse control. However, ANN_P remains in the control system when the training stage is finished, so constituting an internal model. The connection weights of ANN controller ANN_C are hold fixed after the training. It should be observed from the configuration of the internal model control in Figure 2.11 that the output of the plant $y(k)$ becomes equal to the reference $r(k)$ when the identification error (or say model error) $e(k)$ and ANN controller is the exact inverse of the plant (Narendra, 1996). Moreover, feed-backing the model error may improve the performance of the inverse controller.

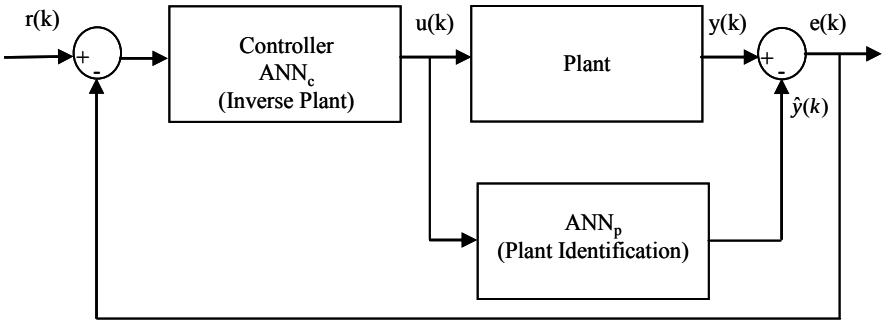


Figure 2.11 Internal model control

2.1.2 State Space Representations

In this subsection, state space representations for linear and/or nonlinear time-invariant cases are introduced in continuous and/or discrete time systems. These cases are used for modeling, analysis and designing of control systems.

2.1.2.1 Linear Time-Invariant Cases

State equations are ordinary differential equations having a special form in which the variables, whose time derivatives appear in the equation, take place at the right hand side together with the inputs and the derivatives of these variables take place at the left hand (See Equation 2.20) for the linear time-invariant case.). System models based on the state equation, called as state space models, was started to be used for control systems at beginning of 1960s by Kalman (Kalman, 1960; 1961). State space models constitute a suitable framework for modeling, analysis and designing of control systems as well as for studying Lyapunov stability, controllability, observability etc.

The state space representation allows a compact expression for linear and also for nonlinear systems. (Chen, 1984; Doyle et al., 1989; Franklin et al., 2000; Goodwin & Sin, 1984; Landau et al., 1997; Landau & Zito, 2006; Phillips & Troy, 1997). For the Linear Time Invariant (LTI) case, the state model of a multi-input, multi-output continuous-time system is defined by the A, B, C and D matrices as follows.

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} \\ \mathbf{y} &= \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u} \end{aligned} \tag{2.20}$$

Where, $\mathbf{A} \in \mathbf{R}^{nxn}$, $\mathbf{B} \in \mathbf{R}^{nxm}$, $\mathbf{C} \in \mathbf{R}^{pxn}$, $\mathbf{D} \in \mathbf{R}^{pxm}$, $\mathbf{x} \in \mathbf{R}^{n}$, $\mathbf{u} \in \mathbf{R}^{m}$ and $\mathbf{y} \in \mathbf{R}^{p}$. The first equations of the state model are called as state equations while the second as the output equations. The states and outputs which are the variables of interest can

be completely determined in terms of **A**, **B**, **C**, **D** matrices and the initial conditions $\mathbf{x}(t_0)$.

Any system defined by Equation 2.20 can be implemented by a set of integrators, pure summers and also linear weighted summers defined by the A, B, C and D matrix-vector multiplications shown in Figure 2.12.

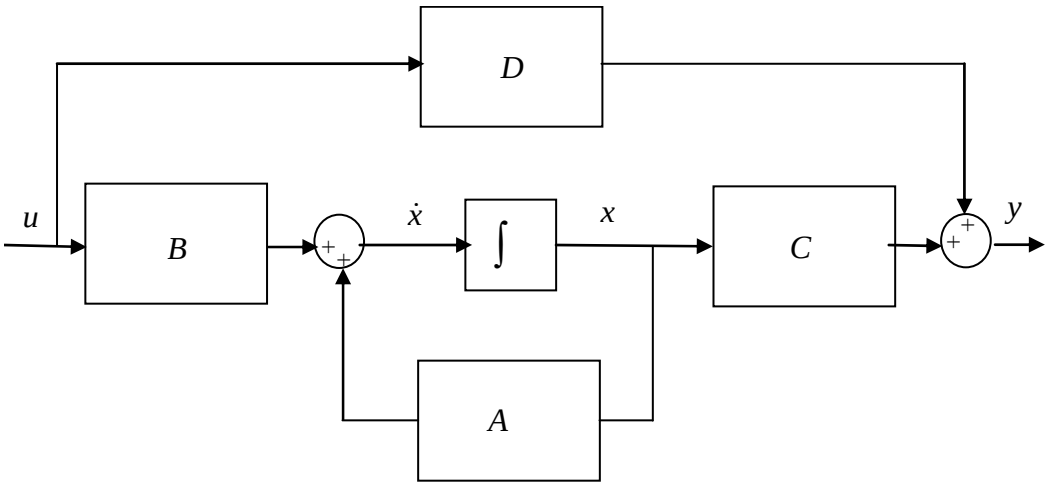


Figure 2.12 State-space block diagram representation of linear, time-invariant continuous-time systems

In order to see the connection between the ARMA model and the state equations, it will be shown that any linear, constant coefficient, nth order differential equation in Equation 2.21 which is, indeed, the continuous-time version of a special ARMA model, where the left hand side defines AR part and the right hand side is the current value of the input, can be recast into a special form of state equations in Equation 2.2.

$$\frac{d^n}{dt^n}y(t)+a_{n-1}\frac{d^{n-1}}{dt^{n-1}}y(t)+\cdots+a_0y(t)=u(t) \qquad (2.21)$$

Where, $u(t)$ is the input, $y(t)$ is the output and $y(t_0), \frac{d}{dt}y(t_0), \cdots, \frac{d^{n-1}}{dt^{n-1}}y(t_0)$ with $t \geq t_0$ are the initial conditions. Let the output and its first *n-1* derivatives be defined as state variables:

$$x_1 \triangleq y, x_2 \triangleq \frac{dy}{dt}, \cdots, x_n \triangleq \frac{d^{n-1}y}{dt^{n-1}} \tag{2.22}$$

Then, the corresponding state equation can be written as follows.

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \vdots \\ \dot{x}_n \end{bmatrix} = \begin{bmatrix} 0 & 1 & \cdots & 0 \\ 0 & 0 & \cdots & \vdots \\ \vdots & \vdots & \ddots & 1 \\ -a_0 & -a_1 & \cdots & -a_{n-1} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix} u$$
$$y = \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \tag{2.23}$$

The state equations defined by $\mathbf{A}$ and $\mathbf{B}$ matrices in Equation 2.23 has a special form, called as controllable canonical state equation form in the control literature. It is known that any (single-input) LTI controllable system in Equation 2.20 can be transformed into the controllable canonical form by a linear change of variables $\bar{\mathbf{x}} =: \mathbf{Tx}$, so the resulting state and input matrices obtained as $\bar{\mathbf{A}} = \mathbf{T}^{-1}\mathbf{AT}$ and $\bar{\mathbf{b}} = \mathbf{T}^{-1}\mathbf{b}$. Where, the controllability of the LTI system in Equation 2.20 can be defined as the existence of a suitable control input $u_{[t_0 \quad t_f]}$ which transfers the system from an arbitrarily given state $x(t_0)$ at time t_0 to any other arbitrarily given state $x(t_f)$ at time t_f (Rugh, 1996).

The necessary and sufficient condition for the controllability of an LTI system in Equation 2.20 is that the controllability matrix in Equation 2.32 has rank *n* (Rugh, 1996).

$$\mathbf{M}_C = [\mathbf{B} \quad \mathbf{AB} \quad \mathbf{A}^2\mathbf{B} \cdots \mathbf{A}^{n-1}\mathbf{B}] \tag{2.24}$$

Checking this condition gives information about the transformability of the state equations into the controllable canonical form Equation 2.23, and identically into the ARMA system in Equation 2.21, in a reversible manner, when the output is chosen as equal to the first state variable. The controllability condition ensuring the existence of the controllability canonical form Equation 2.23 will be used in this thesis as the sufficient condition for the applicability of the developed dynamical state feedback chaotification method.

The discrete-time versions of the state model Equation 2.20 and the corresponding block diagram in Figure 2.12 are given in Equation 2.25 and in Figure 2.13, respectively.

$$\begin{aligned} \mathbf{x}(k+1) &= \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k) \\ \mathbf{y}(k) &= \mathbf{C}\mathbf{x}(k) + \mathbf{D}\mathbf{u}(k) \end{aligned} \tag{2.25}$$

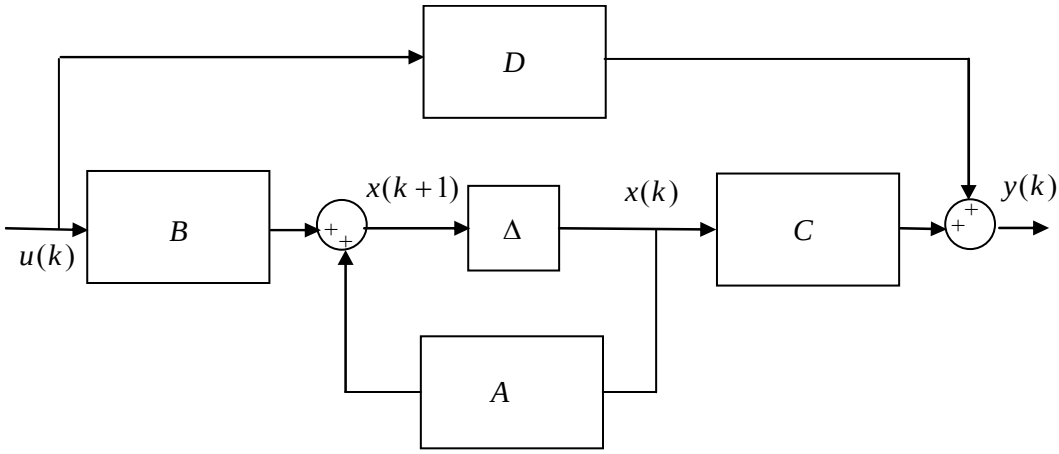


Figure 2.13 State-space block diagram representation of linear, time-invariant discrete-time systems

The definition of the controllability, the controllability matrix, the necessary and sufficient condition for the controllability and the transformability conditions into the

following controllability canonical form are all the same for discrete-time system in Equation 2.25. It should be observed that a single-input, single-output, controllable LTI discrete-time system defined by the state equations Equation 2.26 in the controllable canonical form yields the ARMA model in Equation 2.27.

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \\ \vdots \\ x_n(k+1) \end{bmatrix} = \begin{bmatrix} 0 & 1 & \cdots & 0 \\ 0 & 0 & \cdots & \vdots \\ \vdots & \vdots & \ddots & 1 \\ -a_0 & -a_1 & \cdots & -a_{n-1} \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \\ \vdots \\ x_n(k) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix} u(k)$$
$$y(k) = \begin{bmatrix} 1 0 \cdots 0 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \\ \vdots \\ x_n(k) \end{bmatrix} \tag{2.26}$$

$$y(k+n) = -a_0 y(k) - a_1 y(k+1) \cdots - a_{n-1} y(k+n-1) + u(k) \tag{2.27}$$

Equation 2.27 can be rewritten as in Equation 2.28.

$$y(k) = -a_{n-1} y(k-1) - a_{n-2} y(k-2) \cdots - a_0 y(k-n) + u(k-n) \tag{2.28}$$

Another important property which is actually the dual concept of the controllability and can be studied in a rigorous way by state space representation is the observability. An LTI system in Equation 2.20 is said to be observable if and only if the initial condition $x(t_0)$ can uniquely be determined from the observation of the output $y_{[t_0 \quad t_f]}$ (Rugh, 1996). It is known (Rugh, 1996) that both for the continuous-time in Equation 2.20 and discrete-time in Equation 2.25 models, the necessary and sufficient condition for the observability is that the rank of the following observability matrix is equal to *n*.

$$\mathbf{M_O} = \begin{bmatrix} \mathbf{C} \\ \mathbf{CA} \\ \vdots \\ \mathbf{CA^{n-1}} \end{bmatrix} \tag{2.29}$$

One can identify that the systems in Equation 2.23 and Equation 2.26 are observable.

2.1.2.2 Nonlinear Time-Invariant Case

The most general state space form for nonlinear time-invariant systems is given in Equation 2.30.

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x},\mathbf{u}) \\ \mathbf{y} &= \mathbf{h}(\mathbf{x},\mathbf{u}) \end{aligned} \tag{2.30}$$

Where, $\mathbf{f}(\circ):\mathbf{R^n} x \mathbf{R^m} \rightarrow \mathbf{R^n}$ and $\mathbf{h}(\circ):\mathbf{R^n} x \mathbf{R^m} \rightarrow \mathbf{R^p}$ are nonlinear functions. In this subsection, the bilinear form in Equation 2.31 with no exogenous input in the output equation is assumed for studying feedback linearization concepts for a Multi Input Multi Output (MIMO) system.

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}) + \sum_{i=1}^{m} \mathbf{g}_i(\mathbf{x}) u_i \\ \mathbf{y} &= \mathbf{h}(\mathbf{x}) \end{aligned} \tag{2.31}$$

Where, $\mathbf{f}(\circ):\mathbf{R^n} \rightarrow \mathbf{R^n}$, $\mathbf{g}_i(\circ):\mathbf{R^n} \rightarrow \mathbf{R^n}$, $\mathbf{h}(\circ):\mathbf{R^n} \rightarrow \mathbf{R^p}$ and $u_i(\circ):\mathbf{R} \rightarrow \mathbf{R}$. Such a bilinear system may naturally arise as a state dependent **A**, **B** and **C** matrices by a system modeling/identification procedure or can be obtained by a state transformation from a general state model in Equation 2.31 as depicted in Figure 2.14.

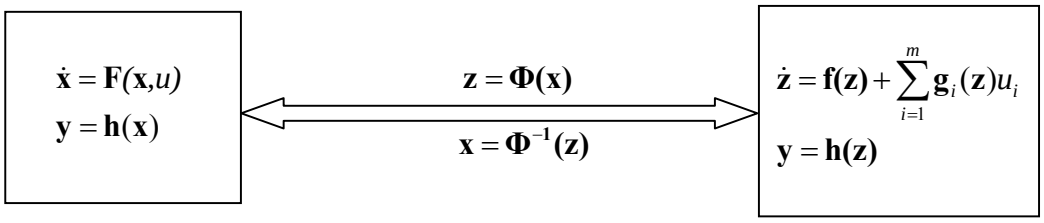


Figure 2.14 Bilinear state model obtained by transformation of a general state model

The controllability, observability and associated canonical forms outlined in the previous section are well established issues in linear systems theory. They are generalized into the nonlinear case by using Lie theory (Brockett, 1973; Cheng et al., 1985; Fliess, 1990; Hermann & Krener, 1977; Isidori, 1985; Jurdjevic & Sussmann, 1972; Kucera, 1967; Sontag, 1990). Nonlinear state space canonical forms including so called Brunovsky form and sufficient conditions on their existence are reviewed in the sequel. It should be noted that Brunovsky canonical form is employed in the thesis for the application of developed dynamical state feedback chaotification method.

The idea behind the nonlinear state space canonical forms is to transform nonlinear systems into linear ones having a suitable state representation as the same with the controllable canonical form presented in the previous section. As will be seen below, this can be achieved by a (nonlinear) state transformation $\mathbf{z} = \Phi(\mathbf{x})$ and by applying a nonlinear static state feedback control $u = \alpha(\mathbf{x}) + \beta(\mathbf{x})v$, i.e. an input transformation, introducing a new control input v. This procedure, which is indeed the nonlinear version of the transformation mapping the state equations into the controllable canonical form, is called as input-state (feedback) linearization (Isidori, 1985; Slotine & Li, 1991). There is another type of feedback linearization, so called input-output linearization which provides a linear dynamical input-output relation rather than a linearized state equation system by an appropriate static state feedback control $u = \alpha(\mathbf{x}) + \beta(\mathbf{x})v$ introducing a new control input v while decoupling the nonlinear effects between the output and this newly defined input.

Input state linearization assumes the bilinear form in Equation 2.31 for state equations. It should be noted that single-input system in Equation 2.32 will be considered for the input-state linearization issues in the thesis for simplicity.

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})u \tag{2.32}$$

Where, $\mathbf{f}(\circ): \mathbf{R}^n \to \mathbf{R}^n$, $\mathbf{g}(\circ): \mathbf{R}^n \to \mathbf{R}^n$ and $u(\circ): \mathbf{R} \to \mathbf{R}$. The input-state feedback linearization is different from the conventional approximate linearization based on the Taylor expansion around an equilibrium point. If there are more than one equilibrium points around which linearization is applied, the first type linearization results in more than one each locally defined linear state models, so a gain-scheduling type control would be used for handling such a control system model (See Subsection 2.3.2.1.). On the other hand, the feedback linearization is an exact linearization and it provides a unique globally defined linear state model which is valid for all equilibrium points of the original nonlinear system (Conte et al., 1999; Grizzle & Kokotovic, 1988; Hunt et al., 1983; Isidori, 1995; Khalil, 1996; Krener, 2003; Sastry, 1999; Sastry & Isidori, 1989; Slotine & Li, 1991; Sontag, 1990; Vidyasagar, 1993). The differences between the abovementioned types of linearization are depicted in Figure 2.15 where A_z and b_z define a controllable canonical form.

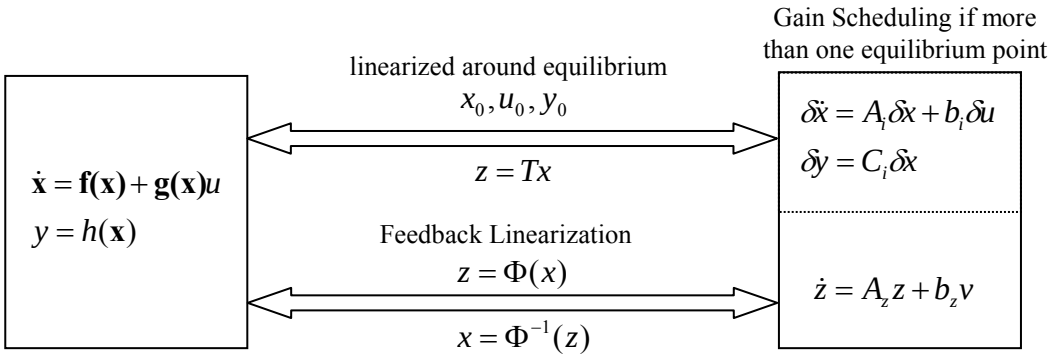


Figure 2.15 Linearization around equilibrium versus input-state feedback linearization

Input – State Linearization: The idea behind the input-state linearization can be explained as follows. If the nonlinear state equations with a scalar input have the

Brunovsky canonical form in Equation 2.33 and $g(\mathbf{x}) \neq 0$, then there exists a state feedback control $u = \alpha(\mathbf{x}) + \beta(\mathbf{x})v =: \frac{1}{g(\mathbf{x})}[-f(\mathbf{x}) + v]$ transforming the state equations in Equation 2.33 into the linear controllable canonical form in Equation 2.23 for the new input v.

$$\begin{bmatrix} \dot{x}_1 \\ \vdots \\ \dot{x}_{n-1} \\ \dot{x}_n \end{bmatrix} = \begin{bmatrix} x_2 \\ \vdots \\ x_n \\ f(\mathbf{x}) + g(\mathbf{x})u \end{bmatrix} \tag{2.33}$$

Where, $f(\circ): \mathbf{R}^n \to \mathbf{R}$ and $g(\circ): \mathbf{R}^n \to \mathbf{R}$. The resulting linear system can then be controlled by a linear control law, for instance, by a static state feedback control as shown in Figure 2.16.

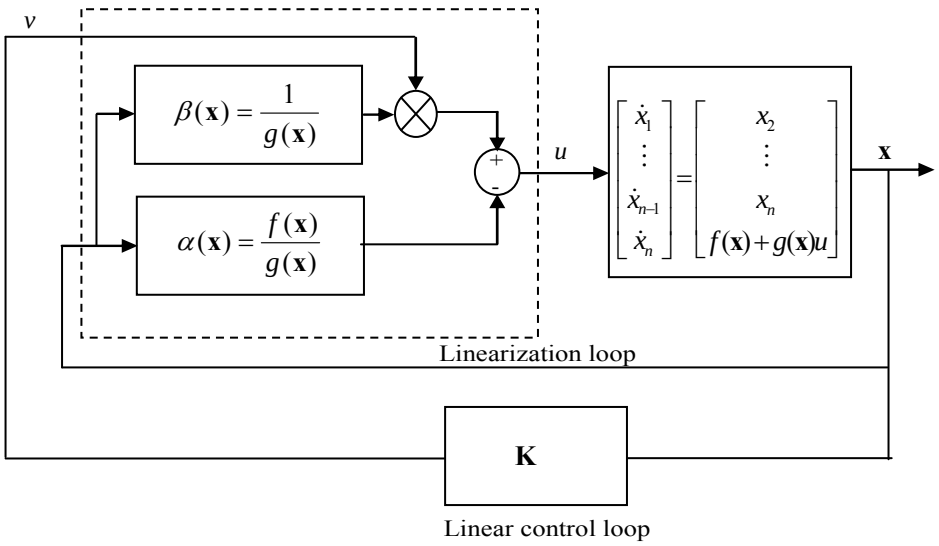


Figure 2.16 Linear static state feedback control of input-state linearized Brunovsky

Now, three questions arise as: i) What are the conditions for a given SISO system in Equation 2.32 to be transformed into the system in Equation 2.33? ii) What are the conditions for $g(\mathbf{x})$ to be nonzero? iii) Do the resulting state equations have the same qualitative properties? Theorem 2.1 gives the answer as the input-state linearizability, which is rigorously defined below, to all these questions.

Definition 2.1 (Slotine & Li, 1991)**:** A single-input system $\dot{\mathbf{x}} = \mathbf{f(x)} + \mathbf{g(x)}u$ where $\mathbf{f(x)} \in \mathbf{R^n}$ and $\mathbf{g(x)} \in \mathbf{R^n}$ with $\mathbf{x} \in \mathbf{R^n}$ are smooth vector fields is said to be input-state linearizable if there exists a region $\mathbf{\Omega} \subseteq \mathbf{R^n}$, a C^k diffeomorphic state transformation $\mathbf{\Phi}(\circ) : \mathbf{R^n} \to \mathbf{R^n}$ and a nonlinear static state feedback control $u = \alpha(\mathbf{\Phi(x)}) + \beta(\mathbf{\Phi(x)})v$ such that the transformed state equations with the new state variables $\mathbf{z} = \Phi(\mathbf{x})$ and the new input $v = f(\mathbf{\Phi(x)}) + g(\mathbf{\Phi(x)})u = \dfrac{1}{\beta(\mathbf{\Phi(x)})}(u - \alpha(\mathbf{\Phi(x)}))$ has the following linear time-invariant state equation form:

$$\dot{\mathbf{z}} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 \end{bmatrix} \mathbf{z} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} v = \mathbf{A}_z \mathbf{z} + \mathbf{B}_z v \tag{2.34}$$

□

Theorem 2.1 (Slotine & Li, 1991): Let $\dot{\mathbf{x}} = \mathbf{f(x)} + \mathbf{g(x)}u$ be a single-input nonlinear system where $\mathbf{f(x)} \in \mathbf{R^n}$ and $\mathbf{g(x)} \in \mathbf{R^n}$ with $\mathbf{x} \in \mathbf{R^n}$ are smooth vector fields. Then, the system is input–state linearizable if and only if there exists a region $\mathbf{\Omega} \subseteq \mathbf{R^n}$ on which the following conditions are satisfied:

i) The set of vector fields $\{\mathbf{g}, ad_{\mathbf{f}}\mathbf{g}, \ldots, ad_{\mathbf{f}}^{n-1}\mathbf{g}\}$ is linearly independent in $\mathbf{\Omega} \subseteq \mathbf{R^n}$.

ii) The set of (linearly independent) vector fields $\{\mathbf{g}, ad_{\mathbf{f}}\mathbf{g}, \ldots, ad_{\mathbf{f}}^{n-2}\mathbf{g}\}$ is involutive, i.e. the Lie bracket of any pair of vector fields in the set is a linear combination of the vector fields in this set. Herein, $ad_{\mathbf{f}}\mathbf{g}$ denotes the Lie bracket $[\mathbf{f}, \mathbf{g}] = (\nabla \mathbf{g})\mathbf{f} - (\nabla \mathbf{f})\mathbf{g}$ with ∇ being the gradient operator with respect to $\mathbf{x}$. □

Linear static state feedback control $v = \mathbf{Kz}$ with $\mathbf{K} = \begin{bmatrix} k_1 & k_2 & \cdots & k_n \end{bmatrix}$ can be applied after input-state linearization as in Figure 2.17.

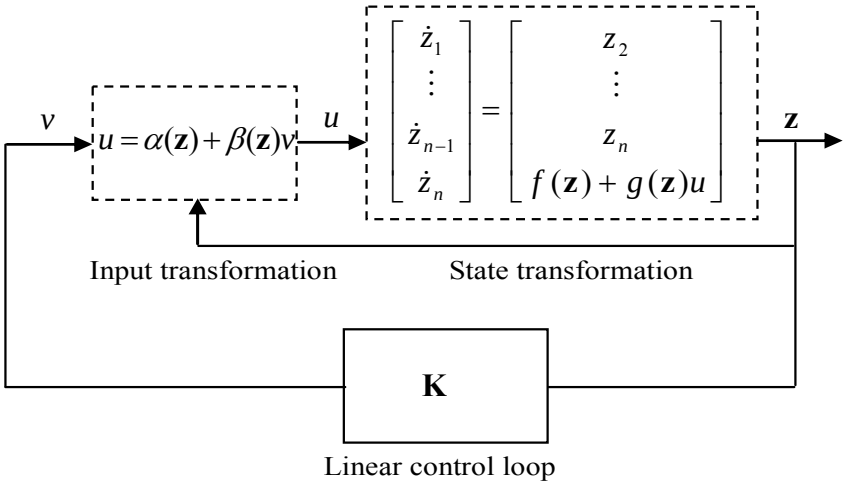


Figure 2.17 Linear static state feedback control of input-state linearized system

For the original system states $\mathbf{x}$, the linear static state feedback control scheme for the input-state linearized system is depicted in Figure 2.18.

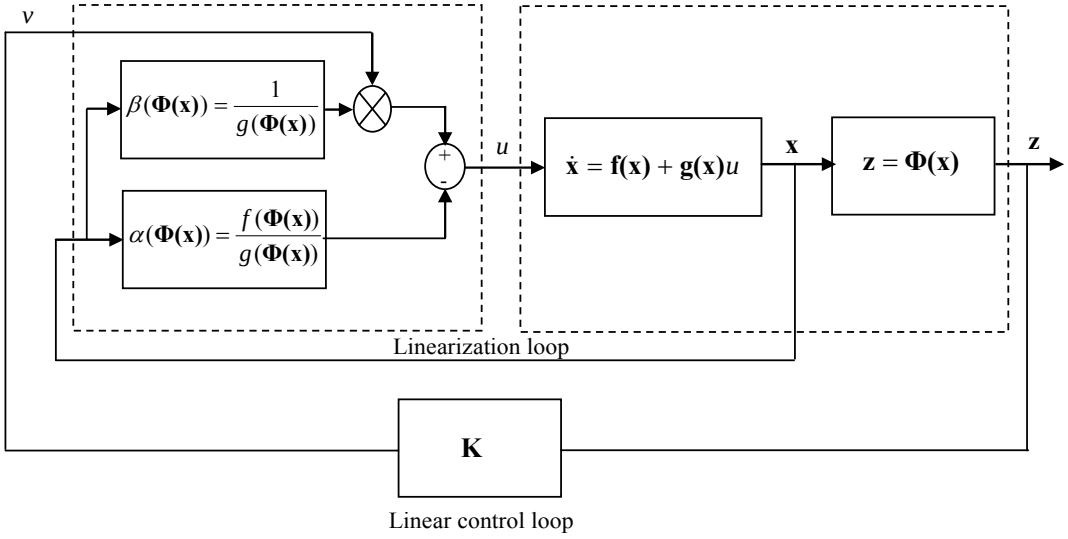


Figure 2.18 Input-state linearization in detail diagram

Input – Output Linearization: Despite the input-state linearization providing a linear state equation system for a new state vector and a new input, input-output linearization constructs a linear, dynamical input-output relation for the same system output with a newly defined input v. As will be explained below, input-output linearization follows a procedure of differentiating outputs with respect to time until

the inputs appear and then apply an appropriate static state feedback control input introducing the new input v while decoupling the nonlinearities. Herein, the aim is to obtain a dynamical input-output relation whose degree is as much as close to the dimension of the state space.

Let the Single-Input, Single-Output (SISO) bilinear, time-invariant systems in Equation 2.35 be considered for simplicity.

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})u \\ y &= h(\mathbf{x}) \end{aligned} \tag{2.35}$$

Where . $\mathbf{f}(\circ) : \mathbf{R}^{\mathbf{n}} \to \mathbf{R}^{\mathbf{n}}$, $\mathbf{g}(\circ) : \mathbf{R}^{\mathbf{n}} \to \mathbf{R}^{\mathbf{n}}$, $u(\circ) : \mathbf{R} \to \mathbf{R}$ and $h(\circ) : \mathbf{R}^{\mathbf{n}} \to \mathbf{R}$. The output function $h(.) \in R^n \to R$ is assumed to be smooth nonlinear function having continuous partial derivatives of any required degree (Hauser et al., 1992; Isidori et al., 1981; Marino & Tomei, 1993; Slotine & Li, 1991).

It can be seen from Equation 2.35 that a relation between the output y and the current input u such that no states $\mathbf{x}$ appearing in the relation can be constructed by taking the derivatives of the output y with respect to time. So, let the first time derivative of y be taken in an open connected region $\mathbf{\Omega} \subseteq \mathbf{R}^{\mathbf{n}}$ as in Equation 2.36.

$$\begin{aligned} \dot{y} &= \frac{\partial y}{\partial t} = \frac{\partial h(\mathbf{x})}{\partial t} = \frac{\partial h}{\partial \mathbf{x}} \frac{\partial \mathbf{x}}{\partial t} = \frac{\partial h}{\partial \mathbf{x}} \dot{\mathbf{x}} = \frac{\partial h}{\partial \mathbf{x}} [\mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})u] \\ &= \frac{\partial h}{\partial \mathbf{x}} \mathbf{f}(\mathbf{x}) + \frac{\partial h}{\partial \mathbf{x}} \mathbf{g}(\mathbf{x})u = (\nabla h)\mathbf{f} + (\nabla h)\mathbf{g}u = L_{\mathbf{f}} h + (L_{\mathbf{g}} h)u \end{aligned} \tag{2.36}$$

Where, $L_{\mathbf{f}} h =: (\nabla h)\mathbf{f}$ scalar function denotes the Lie derivative of the scalar function $h(\mathbf{x})$ with respect to the vector field $\mathbf{f(x)}$ (Slotine & Li, 1991). If $L_{\mathbf{g}} h \neq 0$ for some $\mathbf{x}_Q \in \Omega$, then the input-output relation $\dot{y} = v$ valid in a finite neighborhood

of $\mathbf{x}_Q \in \Omega$ is constructed by the locally defined input transformation $u = -\frac{L_\mathbf{f} h}{L_\mathbf{g} h} + \frac{1}{L_\mathbf{g} h} v = \alpha(\mathbf{x}) + \beta(\mathbf{x}) v$.

If $L_\mathbf{g} h = 0$ for all $\mathbf{x} \in \Omega$, then $\dot{y}$ is differentiated one more time as in Equation 2.37.

$$\begin{aligned} \ddot{y} &= \frac{\partial L_\mathbf{f} h}{\partial t} = \frac{\partial L_\mathbf{f} h}{\partial \mathbf{x}} \frac{\partial \mathbf{x}}{\partial t} = \frac{\partial L_\mathbf{f} h}{\partial \mathbf{x}} \dot{\mathbf{x}} = \frac{\partial L_\mathbf{f} h}{\partial \mathbf{x}} [\mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x}) u] \\ &= \frac{\partial L_\mathbf{f} h}{\partial \mathbf{x}} \mathbf{f}(\mathbf{x}) + \frac{\partial L_\mathbf{f} h}{\partial \mathbf{x}} \mathbf{g}(\mathbf{x}) u = (\nabla L_\mathbf{f} h)\mathbf{f} + ((\nabla L_\mathbf{f} h)\mathbf{g}) u \\ &= L_\mathbf{f}(L_\mathbf{f} h) + (L_\mathbf{g}(L_\mathbf{f} h)) u = L_\mathbf{f}^2 h + (L_\mathbf{g} L_\mathbf{f} h) u \end{aligned} \tag{2.37}$$

Where, higher order Lie derivatives are defined in a recursive way as in Equation 2.38.

$$\begin{aligned} L_\mathbf{f}^{\,0} h &= h \\ L_\mathbf{f}^{\,i} h &= L_\mathbf{f}(L_\mathbf{f}^{\,i-1} h) = (\nabla L_\mathbf{f}^{\,i-1} h)\mathbf{f} \end{aligned} \tag{2.38}$$

If $L_\mathbf{g} L_\mathbf{f} h = 0$ for all $\mathbf{x} \in \Omega$, then the procedure is repeated until $L_\mathbf{g} L_\mathbf{f}^{\,\gamma-1} h \neq 0$ for some $\mathbf{x}_Q \in \Omega$. Then, the input transformation in Equation 2.39 provides a simple dynamical input-output relation $y^{(\gamma)} =: v$ which is defined locally in a finite neighborhood of $\mathbf{x}_Q \in \Omega$. Herein, γ is called as the well-defined relative degree of the system (Slotine & Li, 1991).

$$u = -\frac{L_\mathbf{f}^{\,\gamma} h}{L_\mathbf{g} L_\mathbf{f}^{\,\gamma-1} h} + \frac{1}{L_\mathbf{g} L_\mathbf{f}^{\,\gamma-1} h} v = \alpha(\mathbf{x}) + \beta(\mathbf{x}) v \tag{2.39}$$

If the relative degree γ is equal to n which is the dimension of the state space, then the input-output linearization yields the input-state linearization. As seen from

Equation 2.40, for $\gamma = n$, the state transformation $\mathbf{z} =: \mathbf{\Phi(x)}$ with $\dot{\mathbf{z}} = [z_1, z_2, \ldots, z_n]^T$ defined in Equation 2.40 maps the bilinear form in Equation 2.35 into the Brunovsky canonical form Equation 2.33.

$$
\begin{aligned}
z_1 &=: y = h(\mathbf{x}) \\
\dot{z}_1 &= z_2 =: \dot{y} = L_{\mathbf{f}} h(\mathbf{x}) \\
\dot{z}_2 &= z_3 =: \ddot{y} = L_{\mathbf{f}}{}^2 h(\mathbf{x}) \\
&\vdots \\
\dot{z}_{n-1} &= z_n =: y^{(n-1)} = L_{\mathbf{f}}{}^{\gamma-1} h(\mathbf{x}) + L_{\mathbf{g}} L_{\mathbf{f}}{}^{\gamma-2} h(\mathbf{x}) u \\
\dot{z}_n &= y^{(n)} = L_{\mathbf{f}}{}^{\gamma} h(\mathbf{x}) + L_{\mathbf{g}} L_{\mathbf{f}}{}^{\gamma-1} h(\mathbf{x}) u =: v
\end{aligned}
\tag{2.40}
$$

Then, the obtained Brunovsky form system Equation 2.33 can be transformed into the controllable canonical form Equation 2.34 by the input transformation in Equation 2.41.

$$
u = -\frac{L_{\mathbf{f}}{}^{n} h}{L_{\mathbf{g}} L_{\mathbf{f}}{}^{n-1} h} + \frac{1}{L_{\mathbf{g}} L_{\mathbf{f}}{}^{n-1} h} v = \alpha(\mathbf{x}) + \beta(\mathbf{x}) v \tag{2.41}
$$

The relative degree cited above is indeed an extension of the definition of relative degree in linear systems, i.e. $\gamma = \#(poles) - \#(zeros)$. Using the above procedure, one can derive the input-output relation in Equation 2.42 for the system given by the controllable canonical form in Equation 2.23 whose relative degree $\gamma = n$.

$$
y^{(n)}(t) = -a_{n-1} y^{(n-1)}(t) - \cdots - a_0 y(t) + u(t) \tag{2.42}
$$

Once the linear dynamical input-output relation is obtained for relative degree $\gamma = n$, one can employ a linear (output) feedback control as shown in Figure 2.19.

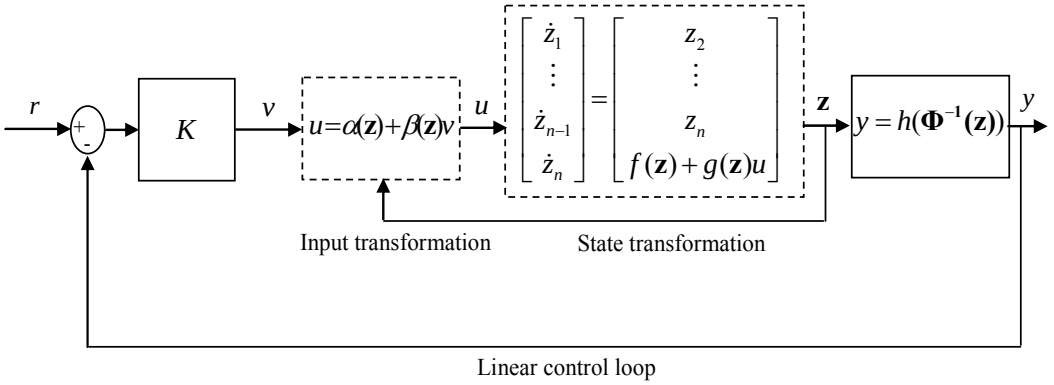


Figure 2.19 Linear output feedback control for systems in Brunovsky form

The input-output realization depicted in terms of the transformed $\mathbf{z}$ variables in Figure 2.19 can be also in terms of the original system variables $\mathbf{x}$ as shown in Figure 2.20.

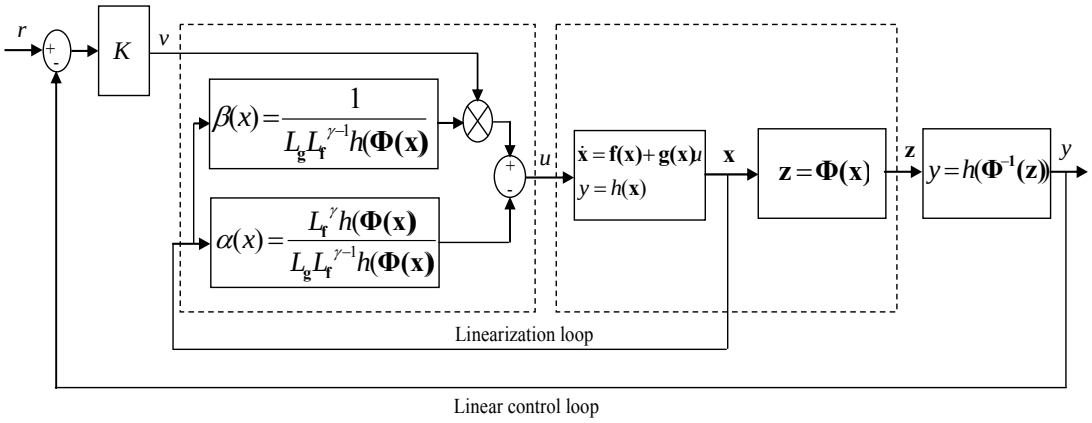


Figure 2.20 Linear output feedback control of input-output linearized systems of relative degree n

If the relative degree of a nonlinear system is less than the dimension of its state space, i.e. $\gamma < n$, a given nonlinear system in Equation 2.35 can be transform into its normal form defined with Equation 2.43 and 2.44 by a state transformation $\mathbf{z} =: \mathbf{\Phi}(\mathbf{x}) =: \begin{bmatrix} \mu_1 & \cdots & \mu_\gamma & \psi_1 & \cdots & \psi_{n-\gamma} \end{bmatrix}^T$ where $\boldsymbol{\mu} =: \begin{bmatrix} \mu_1 \cdots \mu_\gamma \end{bmatrix}^T =: \begin{bmatrix} y \dot{y} \cdots y^{\gamma-1} \end{bmatrix}^T$ (Doyle et al., 1996; Khalil, 1996; Sastry, 1999; Slotine & Li, 1991).

$$\begin{aligned} \dot{\boldsymbol{\mu}} &= \mathbf{A}\boldsymbol{\mu} + \mathbf{B}u \\ \dot{\boldsymbol{\psi}} &= \mathbf{w}(\boldsymbol{\mu}, \boldsymbol{\psi}) \end{aligned} \tag{2.43}$$

$$\begin{bmatrix} \dot{\mu}_1 \\ \vdots \\ \dot{\mu}_{\gamma-1} \\ \dot{\mu}_\gamma \end{bmatrix} = \begin{bmatrix} \mu_2 \\ \vdots \\ \mu_\gamma \\ a(\boldsymbol{\mu},\boldsymbol{\psi}) \end{bmatrix} + \begin{bmatrix} 0 \\ \vdots \\ 0 \\ b(\boldsymbol{\mu},\boldsymbol{\psi}) \end{bmatrix} u \qquad (2.44)$$
$$\dot{\boldsymbol{\psi}} = \mathbf{w}(\boldsymbol{\mu},\boldsymbol{\psi})$$

Where $a(\boldsymbol{\mu},\boldsymbol{\psi}) = L_{\mathbf{f}}^{\gamma} h(\mathbf{x})$ and $b(\boldsymbol{\mu},\boldsymbol{\psi}) = L_{\mathbf{g}} L_{\mathbf{f}}^{\gamma-1} h(\mathbf{x})$. The input can not be seen in the equation $\dot{\boldsymbol{\psi}} = \mathbf{w}(\boldsymbol{\mu},\boldsymbol{\psi})$ since $L_{\mathbf{g}} \boldsymbol{\psi}_i(\mathbf{x}) = 0$ for $i = 1,\ldots,n-\gamma$. On order to control the overall system, the nonlinear system is required to be asymptotically minimum phase, i.e. the following zero dynamics should be asymptotically stable (Slotine & Li, 1991).

$$\begin{aligned} \dot{\boldsymbol{\mu}} &= \mathbf{A}\boldsymbol{\mu} + \mathbf{B}u \\ \dot{\boldsymbol{\psi}} &= \mathbf{w}(\boldsymbol{\mu},\boldsymbol{\psi}) \end{aligned} \qquad (2.45)$$

2.2 Qualitative Analysis of Control Systems

The controllability and observability which are two crucial qualitative properties in a control system design are explained briefly within a state space context in Section 1. The equilibrium dynamics, stability, periodical and chaotic oscillations are among the other issues involved with modeling, identification and control of dynamical systems. The fundamental concepts related with these issues are described below in connection with the main concerns of the thesis which are real-time data dependent controller design for tracking problems and modification of asymptotic behaviors of the systems, namely form the equilibrium dynamics towards chaotic oscillations.

A control system defined by the state equation $\dot{\mathbf{x}} = \hat{\mathbf{f}}(\mathbf{x},\mathbf{u}(t))$ for some control law $\mathbf{u}(t) = \alpha(\mathbf{x},t)$ can define a time-varying state equation $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x},t)$ with $\mathbf{f}(\mathbf{x},t) =: \hat{\mathbf{f}}(\mathbf{x},\alpha(\mathbf{x},t))$. Throughout this section, the systems are assumed defined as

$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x},t)$ with $\mathbf{f}(\circ): \mathbf{R}^n x \mathbf{R} \to \mathbf{R}^n$ being sufficient smoothness implying the existence and uniqueness of the solutions.

2.2.1 Equilibrium Dynamics

The equilibrium is the simplest qualitative behavior of dynamical control systems. On the other hand, one of the most common control problems is to find a control input regulating the system output around a desired position, in other words around an equilibrium point.

Definition 2.2 Equilibrium Point: $\mathbf{x_0} \in \mathbf{R}^n$ is an equilibrium point of $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x},t)$ at t_0 if and only if $\mathbf{f}(\mathbf{x_0},t) = \mathbf{0}\ \forall t \in [t_0, \infty)$. □

2.2.1.1 Stability in the Sense of Lyapunov

Any solution $\mathbf{x}^*(\circ) \in \mathbf{R} \to \mathbf{R}^n$ to the state equation $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x},t)$ can be translated into the zero equilibrium solution of another state equation $\dot{\bar{\mathbf{x}}} = \bar{\mathbf{f}}(\bar{\mathbf{x}},t)$ by a simple coordinate transformation $\bar{\mathbf{x}} =: \mathbf{x} - \mathbf{x}^*$. Where, $\bar{\mathbf{f}}(\bar{\mathbf{x}},t) =: \mathbf{f}(\mathbf{x},t) - \mathbf{f}(\mathbf{x}^*,t) = \mathbf{f}(\bar{\mathbf{x}} + \mathbf{x}^*,t) - \mathbf{f}(\mathbf{x}^*,t)$. So, it is sufficient to define Lyapunov stability of a zero solution.

Definition 2.3 Stability of an Equilibrium Point: The zero solution of $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x},t)$ is said to be stable at t_0 in the sense of Lyapunov if and only if for each $\varepsilon > 0$ there exists $\delta(\varepsilon, t_0) > 0$ such that $\|\mathbf{x}(t, t_0, \mathbf{x_0})\| < \varepsilon$ for all $\|\mathbf{x_0}\| < \delta(\varepsilon, t_0)$ and for all $t \geq t_0$. Where, t_0 is the initial time, $\mathbf{x_0} = \mathbf{x}(t_0)$ is the initial state and $\mathbf{x}(t, t_0, \mathbf{x_0})$ is the solution due to the initial state $\mathbf{x_0} = \mathbf{x}(t_0)$ at the initial time t_0 □

As illustrated by Figure 2.21, Lyapunov stability of the zero solution can be interpreted as that restricting the initial state x_0 to a sufficiently small $\delta(\varepsilon, t_0) > 0$

ball centered at the origin results in that trajectory remains forever in a prescribed ball of radius ε.

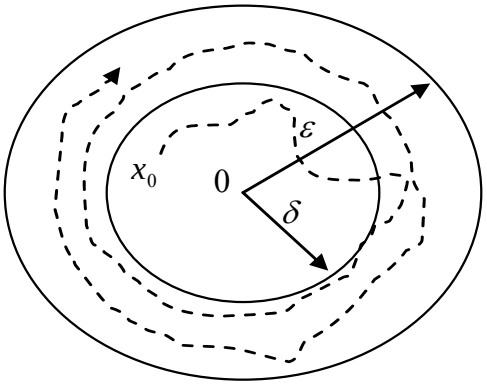


Figure 2.21 Geometrical interpretation of the stability of an equilibrium point in the sense of Lyapunov

Definition 2.4 Asymptotic Stability of an Equilibrium Point: The zero solution of $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x},t)$ is said to be asymptotically stable at t_0 in the sense of Lyapunov if and only if the zero solution of $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x},t)$ is stable in the sense of Lyapunov and if there exists $r(t_0) > 0$ such that $\|\mathbf{x_0}\| < r(t_0)$ implies $\|\mathbf{x}(t,t_0,\mathbf{x_0})\|$ goes to zero as $t \to \infty$. □

If the analytical solution of a considered state equation is known, then one can try to apply Definitions 2.2 and 2.3. However, this is not the case for almost all nonlinear systems. Theorem 2.1 based on Lyapunov stability theory provides a solution for determining stability of nonlinear systems with no need to know the solution of the state equations.

Theorem 2.2 Local Lyapunov Stability (Slotine & Li, 1991): Let $V(\mathbf{x},t)$ be a scalar function with continuous partial derivatives, which is defined in a B_{R_0} local neighborhood of the origin with $R_0 > 0$. Then, if $V(\mathbf{x},t)$ is positive definite in B_{R_0}, i.e. $V(\mathbf{x},t) > 0$ for all $\mathbf{x}$ other than origin in B_{R_0} and its time derivative along

trajectory is negative semi-definite B_{R_0}, i.e. $\dot{V}(\mathbf{x},t) \le 0$ for all $\mathbf{x}$ in B_{R_0}, then the equilibrium point is zero in the sense of Lyapunov. □

If the negative semi-definiteness condition for $\dot{V}(\mathbf{x},t)$ in Theorem 2.1 is replaced with the negative definiteness, i.e. $\dot{V}(\mathbf{x},t) < 0$ for all $\mathbf{x}$ other than the origin in B_{R_0}, then Theorem 2.1 provides the uniform asymptotic stability of the zero equilibrium point.

2.2.1.2 Input-Output Stability

Stability is studied based on the state space model in the previous section. Stability can also be studied in an input-output context as briefly explained below. Let L_p^n denote the norm defined in the n-valued function space as

$$\left\|\mathbf{u}(\circ)\right\|_p =: \left(\int_0^\infty \left\|\mathbf{u}(t)\right\|^p dt \right)^{\frac{1}{p}} \tag{2.46}$$

Where, $\|(\circ)\|$ is the usual vector norm. Let L_{pe}^n denote the extended L_p^n norm obtained by truncating the functions up to time T.

Definition 2.5 Finite Gain $\mathbf{L_p}$ Stability (Sastry, 1999): A casual nonlinear operator $\mathbf{A(\circ) : L_{pe}^n \to L_{pe}^n}$ is said to be finite gain $\mathbf{L_p}$ stable if and only if i) for any given input $\mathbf{u(\circ) \in L_p^n}$, the output $\mathbf{A(u(\circ)) \in L_p^n}$ and ii) there exist $k, \beta > 0$ such that, for any given $\mathbf{u(\circ) \in L_{pe}^n}$, the output $\mathbf{A(u(\circ)) \in L_{pe}^n}$ satisfies $\left\|\mathbf{A(u(\circ))}\right\|_T \le k\left\|\mathbf{u(\circ)}\right\|_\mathbf{T} + \beta$ for all $T > 0$. □

The small gain theorem provides a tool for analyzing input-output stability of a closed-loop control system.

Theorem 2.3 Small Gain Theorem (Sastry, 1999): Consider the system in Figure 2.22 where the individual systems $\mathbf{G}_1(\circ):\mathbf{L}_{\mathbf{pe}}^{\mathbf{n}i} \to \mathbf{L}_{\mathbf{pe}}^{\mathbf{n}o}$, $\mathbf{G}_2(\circ):\mathbf{L}_{\mathbf{pe}}^{\mathbf{n}o} \to \mathbf{L}_{\mathbf{pe}}^{\mathbf{n}i}$ with inputs $\mathbf{r}_1$, $\mathbf{r}_2$ and outputs $\mathbf{y}_1$, $\mathbf{y}_2$ are both causal and finite gain stable, i.e. there exist $k_1, k_2, \beta_1, \beta_2$ such that

$$\begin{aligned} \left\|\mathbf{G}_1(\mathbf{e}_1(t))\right\|_{\mathbf{T}} &\le k_1 \left\|\mathbf{e}_1(t)\right\|_T + \beta_1 \\ \left\|\mathbf{G}_2(\mathbf{e}_2(t))\right\|_{\mathbf{T}} &\le k_2 \left\|\mathbf{e}_2(t)\right\|_{\mathbf{T}} + \beta_2 \end{aligned} \tag{2.47}$$

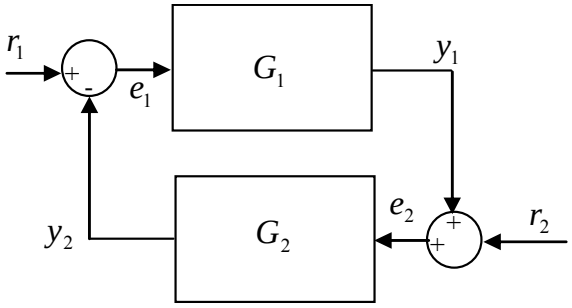


Figure 2.22 Interconnected control systems

Assume that the loop is well posed in the sense that for any given $\mathbf{r}_1(\circ) \in \mathbf{L}_{\mathbf{pe}}^{\mathbf{n}i}$, $\mathbf{r}_2(\circ) \in \mathbf{L}_{\mathbf{pe}}^{\mathbf{n}o}$ there are unique $\mathbf{e}_1(\circ), \mathbf{y}_2(\circ) \in \mathbf{L}_{\mathbf{pe}}^{\mathbf{n}i}$ and $\mathbf{e}_2(\circ), \mathbf{y}_1(\circ) \in \mathbf{L}_{\mathbf{pe}}^{\mathbf{n}o}$ satisfying the loop equations below

$$\begin{aligned} \mathbf{e}_1 &= \mathbf{r}_1 - \mathbf{G}_2(\mathbf{e}_2), \quad \mathbf{y}_1 = \mathbf{G}_1(\mathbf{e}_1) \\ \mathbf{e}_2 &= \mathbf{r}_2 - \mathbf{G}_1(\mathbf{e}_1), \quad \mathbf{y}_2 = \mathbf{G}_2(\mathbf{e}_2) \end{aligned} \tag{2.48}$$

Then, the closed loop system is finite gain stable from $\mathbf{r}_1$, $\mathbf{r}_2$ to $\mathbf{y}_1, \mathbf{y}_2$ if $k_1 k_2 < 1$.

□

2.2.2 Periodical and Chaotic Oscillations

Periodical and chaotic steady-state behaviors arise in dynamical control systems due to the feedback and intrinsic nonlinearities usually unintentionally (Alvarez-Hernández et al., 2002; Takigawa et al., 2000; Ye & Chau, 2007; Zhang & Chen, 2005). These complex dynamics are also desired to be created for some engineering applications such as chaotification of DC motor (Ye & Chau, 2007; Zhang & Chen, 2005).

Definition 2.6 Periodical Solution (Parker & Chua, 1987a, 1987b). A solution $\mathbf{x}(t, t_0, x_0)$ to $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x},t)$ is said to be periodical if and only if there exists a $T > 0$ such that $\mathbf{x}(t,t_0,x_0) = \mathbf{x}(t+T,t_0,x_0)$ for all t. Where, the smallest $T > 0$ is called as the period. □

A periodic solution has a discrete Fourier spectrum consisting of $f = \frac{1}{T}$ as the main harmonic and $\frac{k}{T}$ as sub-harmonics. In control literature, asymptotically stable periodical solutions are called as limit cycle. Limit cycle and stable periodical behaviors of a control system $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x},t)$ can be studied based on the results given in the previous subsection.

A more complex dynamical behavior is the quasi periodicity defined below.

Definition 2.7 Quasi-Periodical Solution: A solution $\mathbf{x}(t, t_0, x_0)$ to $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x},t)$ is said to be quasi-periodic if and only if there exist a set of periodic functions $p_i(t)$ each of which has period T_i such that $\mathbf{x}(t; t_0, x_0) = \sum_i p_i(t)$. Where, the periodic functions set may not be finite but $f_i = \frac{1}{T_i}$ can be written as an integer-

weighted combination of a finite set $\{\bar{f}_1, \cdots, \bar{f}_P\}$ of base frequencies, i.e. $f_i = \sum_j^P k_j \bar{f}_j$ with $k_j \in Z$. □

Quasi periodic solutions may arise in periodically excited systems originally possessing a limit cycle, for instance the non-autonomous version of van der Pol system (Parker & Chua, 1987a, 1987b). The quasi periodicity of a system can be identified best from its k-torus limit set in the state space.

The most complicated dynamics for deterministic systems is currently known as chaos. The limit set of a chaotic dynamics is of fractal dimensional as opposed to the equilibrium dynamics having zero dimensional limit set, periodical and quasi-periodical dynamics having one or higher integer dimensional limit sets. They have broad band frequency spectra. Chaotic dynamics, which has bounded trajectories as stable equilibrium and limit cycle, has three distinguishing features; i) sensitive dependency on initial conditions, ii) topological transitivity, also called as mixing property, and iii) denseness of periodic orbits.

Definition 2.8 Chaotic Invariant Set (Wiggins, 1992)**:** Let $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$ be an autonomous system with a continuously differentiable function $\mathbf{f}(\circ): \mathbf{R^n} \to \mathbf{R^n}$. Assume its flow $\Phi(\mathbf{x},t)$ exists for all $t \geq 0$. Let $\mathbf{\Lambda} \in \mathbf{R^n}$ be a compact set which is invariant under the flow $\Phi(\mathbf{x},t)$. Then, the invariant set $\mathbf{\Lambda} \in \mathbf{R^n}$ is said to be chaotic if

i. $\Phi(\mathbf{x},t)$ is sensitive dependent on the initial conditions in $\mathbf{\Lambda} \in \mathbf{R^n}$, i.e. $\exists \varepsilon \geq 0$ such that $\forall \mathbf{x} \in \Lambda$ and for some U neighborhood of $\mathbf{x}$, $\exists \hat{\mathbf{x}} \in U$ and $\exists t > 0$ such that $\|\mathbf{\Phi(x,t)} - \mathbf{\Phi(\hat{x},t)}\| > \varepsilon$.

ii. $\Phi(\mathbf{x},t)$ is topologically transitive on Λ, i.e. for each pair of $U, V \subset \Lambda$ open sets $\exists t \in R$ such that $\Phi(U,t) \cap V \neq \phi$. Where, ϕ stands for empty set.

iii. Periodical trajectories of the flow $\Phi(\mathbf{x},t)$ are dense in Λ, i.e. in each open neighborhood of a periodical trajectory there must exist another periodical trajectory. □

Sensitive dependence on initial conditions implies a local expansion, in other words divergence of two neighboring trajectories. Lyapunov exponents which can be calculated using trajectories give information about the chaotic behavior: For a chaotic behavior, the largest Lyapunov exponent λ defined by Equation 2.49 is positive (Peitgen et al., 1992; Sastry, 1999; Wolf et al., 1985).

$$\lambda = \limsup_{t\to\infty}\left(\frac{1}{t}\log\left(\frac{\left\|\mathbf{x}(t)-\hat{\mathbf{x}}(t)\right\|}{\left\|\mathbf{x}_0-\hat{\mathbf{x}}_0\right\|}\right)\right) \tag{2.49}$$

Where, $\mathbf{x}$, $\hat{\mathbf{x}}$ denote two trajectories due to the initial states $\mathbf{x}_0$, $\hat{\mathbf{x}}_0$. If the largest Lyapunov exponent λ is zero, then the associated (bounded solution) system is a limit cycle. If $\lambda < 0$, it is a stable equilibrium.

2.3 Controller Design

Stages of a controller design can be given as follows (Doyle et al., 1992):

- Set up the real system to be controlled with sensors and actuators
- Model the system and identify its properties.
- Design controller to meet performance specifications
- Analyze the system by simulation
- Implement controller hardware and/or control algorithm software
- Tune the controller parameters in a real-time application.

One of the most important issues in controller design is to find a suitable controller to meet given performance specifications for a control system under consideration. This issue can be solved by linear controllers such as Proportional Integral Derivative (PID), high gain, state and output feedbacks (Goodwin et al., 2001; Dorf, 1989; Franklin et al., 1991; Kuo, 1995; Ziegler & Nichols, 1942). PID is widely used in industrial applications sharing ninety percent of overall controllers due to its high efficiency compared to its simplicity (Astrom & Hagglund, 1995). The main control design methods are described in this subsection.

2.3.1 Feedback Control

Feedback control methods can be grouped into static and dynamical feedbacks (Goodwin et al., 2001). The dynamical feedback control structure is shown in Figure 2.23 for linear, time-invariant, discrete-time case.

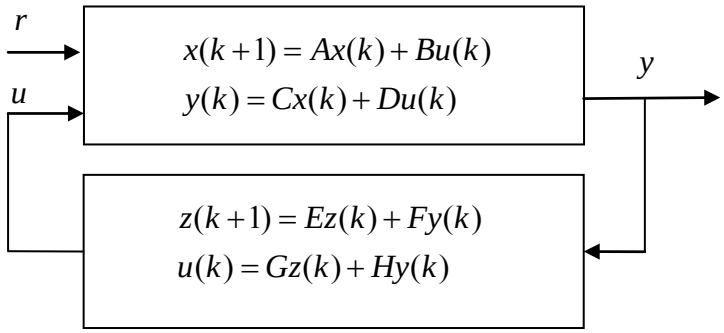


Figure 2.23 Dynamical feedback control scheme

The feedback control in Figure 2.23 becomes static feedback when $u(k) = Hy(k)$.

2.3.1.1 State and Output Feedbacks

Both static and dynamical feedback methods can also be classified into state and output feedback subgroups (Cao et al., 1998; Ding, 1999; Ding, 2002; Ezal et al., 1999; Pimpalkhare & Bandyopadhyay, 1994; Trofino-Neto & Kucera 1993; Syrmos et al., 1994). If the feedback is in terms of the outputs, then it is called as output feedback. Otherwise, if it is in terms of states then it is called as state feedback.

2.3.1.2 Proportional Integral Derivative (PID) Controllers

A PID controller is defined by Equation 2.50 in the time-domain or by Equation 2.51 in the s-domain (Astrom & Hagglund, 1995; Goodwin et al., 2001; Katsuhiko, 2002; Zhang et al, 2004).

$$u(t) = K_p e(t) + K_i \int_0^t e(t)dt + K_d \frac{de(t)}{dt} \tag{2.50}$$

$$\frac{U(s)}{E(s)} = K_p + \frac{K_i}{s} + K_d s \tag{2.51}$$

Where, u is the controller output, e is the controller input, K_p, K_i and K_d stands for proportional, integral and derivative gain, respectively. The above given descriptions provide input-output representations. A state space model for PID controller can be obtained as in Equation 2.52 and 2.53 by assuming $\dot{x}_1 =: x_2$ and $x_2 =: e$ which imply $\dot{x}_1 = e$ and $\dot{x}_2 = \dot{e}$.

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \dot{e} \tag{2.52}$$

$$u = \begin{bmatrix} K_i & K_p \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} K_d \end{bmatrix} \dot{e} \tag{2.53}$$

As seen more clearly from Figure 2.24, a PID is cascade of a pure differentiator with a 2-dimensional state model whose input is the derivative $\dot{e}$ of the error.

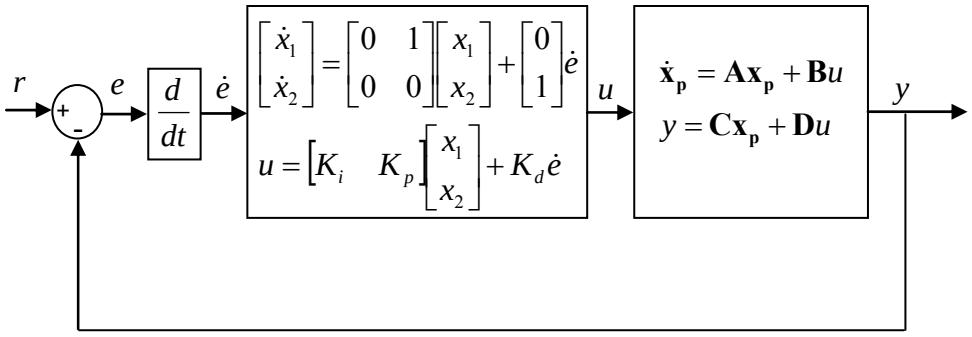


Figure 2.24 Block diagram for state representation of PID control

It is interesting to note that PID is a dynamical output feedback controller and it is controllable for all choices of PID gains and observable whenever $K_i \neq 0$.

2.3.2 Nonlinear Control

In real world applications, linear controllers are not enough when dealing with nonlinear systems having nonlinearities in plant, sensor and/or actuator etc. Feedback linearization which is already described in Subsection 2.1.2.1 and adaptive control are two essential methods widely used in nonlinear control area (Khalil, 1996; Kokotovic, 1992; Krener, 2003; Krstic M.et al., 1992; Sastry, 1999; Sepulchre et al., 1997; Vidyasagar, 1993; Xu & Ioannou, 2003). The idea of feedback linearization is to transform a given nonlinear system into a linear one, so applying linear control methods. Adaptive control considers nonlinearities as changes in the plant and/or in the environments, so employs an adaptive control for handling these changes.

2.3.2.1 Adaptive Control

The adaptive control is a control method where the controller parameters are changed in an online fashion according to the changes in the plant and/or environment. Adaptive control methods can be categorized into two groups: i) direct methods and ii) indirect methods.

Controller parameters in the direct methods are changed as a function of the output of the plant. Gain scheduling and Model Reference Adaptive Control (MRAS) are examples for direct control. In MRAS, there is also a reference model so that the controller parameters are changed as a function of its output which is, in fact, desired to be tracked by the control system in an adaptive way.

On the other hand, the controller parameters in the indirect methods are changed in accordance with the plant parameters which are estimated by certain techniques. Self tuning regulator and dual control are examples for indirect methods.

In any kind of adaptive control, the controller becomes nonlinear and/or time-varying even if the plant and the chosen model for controller are linear and time-

invariant (Astrom, 1987; Astrom & Wittenmark, 1994, 1997; Middleton et al., 1988; Narendra & Kudva, 1974; Sastry & Bodson, 1989; Slotine & Li, 1991; Pan & Başar, 1998; Wu et al., 2007). As shown in Figure 2.25, an adaptive control system has two loops: i) the feedback loop ii) the controller parameter calculation loop. In the latter loop, the controller parameters are updated in accordance with plant output, controller output and reference input.

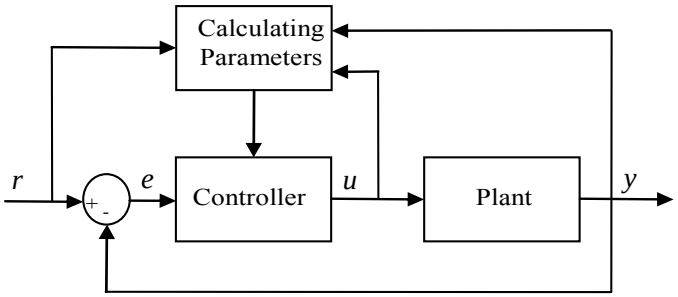


Figure 2.25 General adaptive control schemes

Gain Scheduling Method: Gain scheduling is a switching technique for controlling nonlinear systems via a set of linear controllers each of which works for a different operating point of the system. Controller's parameters are chosen according to measured variables as observing the plant. This measured variable is called gain and its changing is called schedule, so this method is called as gain scheduling. Gain scheduling does not need parameter estimation from the plant, so it responses very quickly to plant changes (Astrom, 1987; Astrom & Wittenmark, 1994; Blanchini, 2000; Leith & Leithead, 2000).

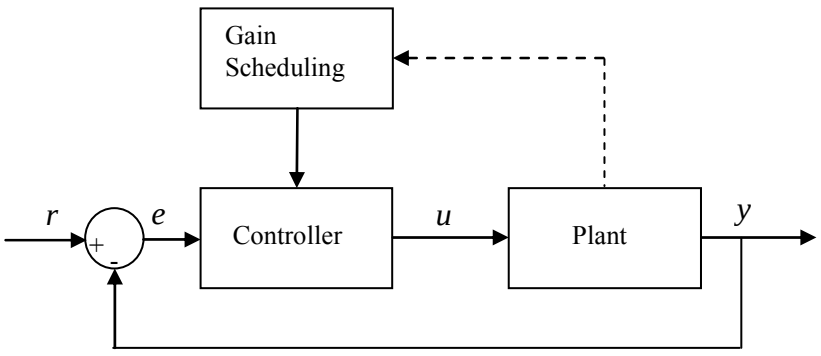


Figure 2.26 Gain scheduling adaptive control scheme

Model Reference Adaptive Control (MRAC) Method: In MRAC, parameter tuning process takes into account the outputs of the reference model, actual plant and controller. The reference model of MRAC which is, indeed, a stable model of the plant, provides to cope with external disturbances and parameter uncertainties. The difference between the actual plant output and reference model output serves as model error. Gradient based algorithms for minimizing squared model error are widely used for updating controller parameters. MIT rule is such a method (Astrom & Wittenmark, 1994; Hatwell et al., 1991; Narendra & Balakrishnan, 1997; Slotine & Li, 1991).

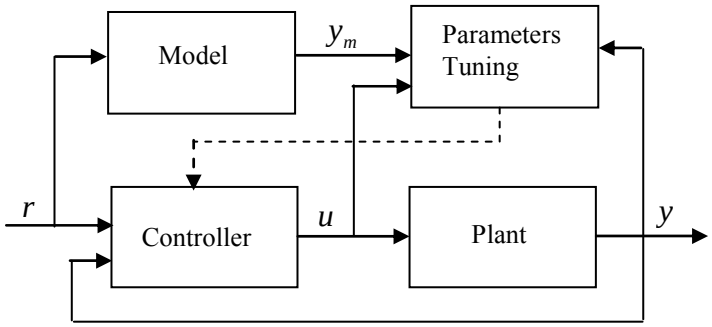


Figure 2.27 MRAC scheme

The reference model is chosen according to the degree of the model. Its structure should fit to the actual closed loop system structure.

Self Tuning Regulators (STR) Method: In STR, the controller parameters are updated according to the estimation of the plant parameters and given performance criteria. The basic idea of STR can be stated as first find plant parameters in an online fashion by an identification method, and then update the controller parameters again online (Astrom & Wittenmark, 1994; Slotine & Li, 1991).

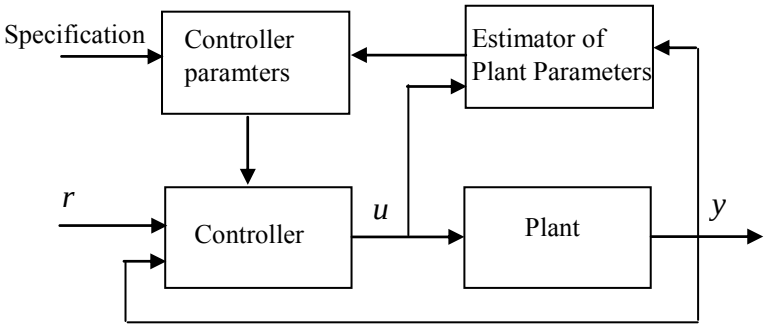


Figure 2.28 STR adaptive controls

Dual Control Method: As opposed to the above given three adaptive control methods all of which do not take into account parameter uncertainties, dual control employs nonlinear stochastic control theory for handling uncertainties in parameter estimation. In dual control, the controller takes special actions under poor knowledge about the plant parameters. (Astrom & Wittenmark, 1994; Filatov & Unbehauen, 2000; Wittenmark, 1995). As shown in Figure 2.29, the controller comprised two parts; i) nonlinear estimator and ii) feedback controller. The estimator generates the hyperstate, i.e the conditional probability distribution of the state, from the measurements.

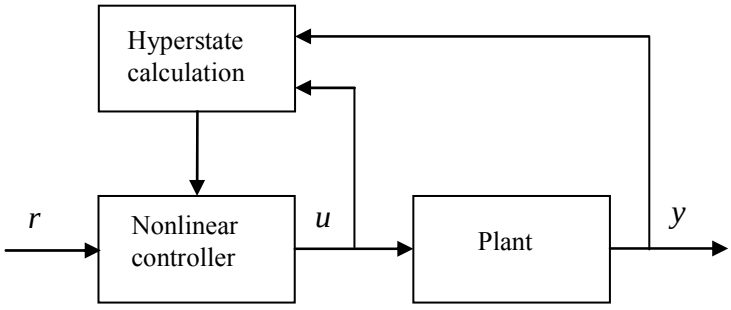


Figure 2.29 Dual adaptive control scheme

CHAPTER THREE

REAL TIME SIMULATION FOR CONTROLLER DESIGN, TEST AND REDESIGN

A microcontroller based low-cost platform managed by a graphical user interface for the design-test-and-redesign of controllers through examining the simulated, emulated or real controller candidates on the simulated, emulated or real plants are under the real-time and real environmental conditions, e.g. analog disturbances, created by a hardware peripheral unit. The platform provides a set of novel real-time operating modes as well as the well-known simulation modes such as hardware-in-the-loop and software-in-the-loop etc. The operating modes are described in a comparative way by an introduced taxonomy and also softly categorized based on their suitability to the design, test and redesign stages of an introduced controller design procedure. The simulation modes and design procedure offered are verified on several benchmark plants and on a real plant, i.e. a micro DC motor. The reliable operating frequency ranges of the platform are determined by an investigation through a set of experiments. It is observed that synchronization of coupled oscillators, chaotic ones in particular, where the master system realized in one sub-unit and the slave in the other constitutes a useful benchmark for validation of such real-time simulation platforms requiring synchronized operation of different sub-units, herein the PC, microcontroller and hardware peripheral unit.

3.1 Structure and Functions of the CDTRP

The developed platform CDTRP comprises four main units which are a real-time simulator running in a PC, a real-time plant emulator realized in a plant emulator card, a hardware peripheral unit for recreation a real environment for the plant emulator, and a GUI in the PC which manages the whole platform. The structure and the interconnection of the sub-units of CDTRP are depicted in Figure 3.1. The first unit is the software simulator implemented in the PC. It aims to simulate the controller, the plant and/or the peripheral unit components depending on the

operating modes of the platform, i.e. it simulates, for instance, the plant if it is active in the chosen mode of operation. The simulation program is in terms of the MATLAB codes and it requires controller algorithm and plant model as MATLAB's M-files. The simulator is designed to run essentially in real-time; however, it can run in any time step faster or slower than real-time which might be preferred depending on the application.

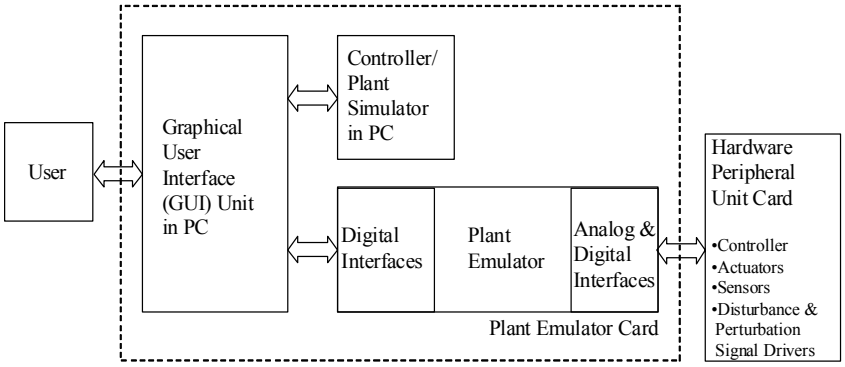


Figure 3.1 Structure of Controller-Design-Test-Redesign-Platform (CDTRP)

The second unit is the emulator card whose core is the PIC microcontroller 18F452. The microcontroller is devoted to emulate the controller, the plant and/or the peripheral unit components depending on the operating modes of the platform. The plant emulator card also possesses digital and analog interfaces for the communication of the plant emulator with the other units of CDTRP, i.e. the GUI and the Hardware Peripheral Unit Card. The PIC is programmed by PICC software ran in the PC before installing it into the plant emulator card, so become ready to be managed by GUI and to communicate with the simulator in the PC and with the hardware peripheral unit card via the GUI. (The details on the hardware realization of the emulator card are given at the end of this section, i.e. Subsection 3.4.) The third unit of the CDTRP, i.e. the hardware peripheral unit card is the most flexible part of the platform: Depending on the application, it contains (analog and/or digital) hardware controller, actuators, sensory devices and signal drivers corresponding to the external disturbance and parameter perturbations to recreate the real environmental conditions for the plant emulator. The fourth unit, the GUI, of the CDTRP provides the management of whole platform. The GUI is implemented with

more than 2000 lines of MATLAB codes in the MATLAB's GUI designer tool. The GUI serves to the users as a monitoring and controlling unit for the platform. The GUI manages all operating modes listed in Tables 3.1-3 and the environmental conditions provided by the hardware peripheral card. A view of the front panel of the GUI is given in Figure 3.2 where the monitoring and controlling tools supplied to the users by the GUI are visible.

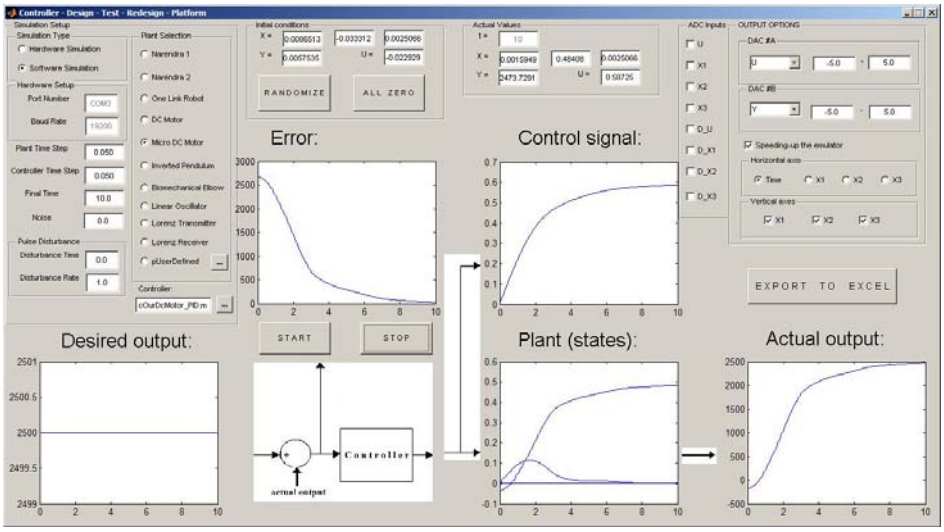


Figure 3.2 Front panel of the GUI for the platform

The main features and management (i.e. monitoring and controlling.) facilities of the GUI are listed below:

- It is a user-friendly GUI, i.e. easy to be used via a set of interface entries such as start, stop, export to excel, randomizing and all zeroing of the initial conditions of the simulator and the emulator.
- Selection buttons exist for switching the simulation type for the plant as simulator in PC or emulator on the plant emulator card, so for determining the operation mode of the CDTRP. It should be noted that there is no restriction on the numbers of input and output of the plant in the simulator, but the emulator allows up to 8 inputs and 2 outputs for the plants. The emulator is actually designed for realizing the plants which are single-input, single-output. If the inputs such that the one reserved for the additive (to the control input) disturbance signal, three of them for the multiplicative (to the

possible three states) parameter perturbation signals and the rest four for sensory devices and/or for other peripheral units are used in the specific application, then they may be used for the plant emulator as extra input and output terminals.

- A set of benchmark plants are available in the menu of CDTRP such that one of them can be chosen by the user via the associated button in the front panel in order to test a considered controller. The users can also define, in the user-defined button, their own plant models as in the form of state equations by uploading the MATLAB's M-file corresponding to the considered plant via browser.
- The controller algorithm to be run in the PC simulator can be loaded on the front panel via browser as a MATLAB's M-file.
- The time steps for the controller and the plant can be determined in an arbitrary way and also independently from each other irrespective to whether or/not the plant is implemented in the simulator or in the emulator. The final time can be set by using the final time box in the front panel of the GUI to a real time value or set to "inf" option which means the simulator or the simulator to run indefinitely.
- The external analog signals, which are intended to be generated in the hardware peripheral unit and then fed to the emulator, are selected by using the menu of ADC inputs, i.e. U, X1, X2, X3, D_U, D_X1, D_X2 and D_X3 which take place in the front panel. U, X1, X2 and X3 are for hardware-in-the-loop simulations (See Tables 3.1-3).
- "Speeding-up the emulator" option controlled from the front panel can be used if the emulator is communicated with the hardware peripheral unit only and there is no need to transfer data from the emulator to the GUI in any way. It inactivates the interconnection between the emulator and the GUI so allowing the emulator to spare its computing power for the communication with the hardware peripheral unit.
- Noise box in the front panel is used for magnifying or reducing the amplitude of the noise constituting a part of the external analog signals generated D_U

in the hardware peripheral unit card which is aimed to be used as additive to the control signal.

- Parameter variation signals, which constitute a second part of the external analog signals generated in the hardware peripheral unit, can be applied via D_X1, D_X2 and D_X3 ports of the plant emulator card in a multiplicative way to each state of the plant emulator. D_U, D_X1, D_X2 and D_X3 values are displayed in real-time on the fourth row of the LCD display of the plant emulator card.
- The multiplicative output disturbance set-up of the front panel, which is included for applying an single short pulse disturbance multiplicative to the plant output, enables to set the time when the pulse is applied and the pulse amplitude for the operating modes such as S-S-S, S-E-S, S-E-E, S-R-R in Tables 3.1-3.
- The initial conditions for the plant implemented both in the PC simulator and in the emulator can be assigned in a random way as using by a randomizing button placed in the front panel which activates MATLAB's standard randomizing function and also they can be set to the user defined values.
- The front panel of the GUI has five monitors, each displays one of the implemented control system's variables, i.e. the desired (plant) outputs, the error signals which are indeed the inputs of the controller, the control signals which are the outputs of the controller and at the same time the inputs of the plant excluding the environmental signals, the state variables of the plant (up to three states), and the actual outputs of the plant. It should be noted that the monitors corresponding to more than one variable displays all the variables together in the same plane where the horizontal axis, i.e. the time and the vertical axis are in the same scale. Only the state variable monitor has an additional feature of replacing the time axis with one of the state variables, so enabling to plot the two-dimensional projections of the state space, i.e. the phase portraits.
- The front panel has also monitors indicating the precise (MATLAB) values (up to eight digit lengths) of the states of the plant, the control signal and the actual plant output at the current time step set for the plant.

- Two (digital to analog) converters' outputs of the plant emulator card can be configured via the output options menu in the front panel as to be devoted to two of the system states and/or the actual output and/or the control signal. The operating ranges of the outputs of the converters are also adjusted again by the output options menu.
- The data related to the control systems variables, which are all monitored via the GUI, can be exported to the excel file as a data logger.

3.2 Taxonomy of Real Time Simulation Modes Realized by CDTRP

With the features of the GUI mentioned in Subsection 3.1, the developed simulation platform CDRTP becomes self-contained and it can implement 24 different real time operating modes given in Tables 3.1-3 where three letters coding is used for representing operating modes. The first, second and third letters of them stand for the controller, plant and peripheral unit, respectively.

Table 3.1 Taxonomy of real-time simulation modes of CDTRP: peripheral unit implemented in the simulator (PC)

Controller \ Plant	Simulator (in PC)	Emulator	Real (Hardware)
Simulator (in PC)	S-S-S	S-E-S	S-R-S
Emulator	E-S-S	E-E-S	E-R-S
Real (Hardware)	R-S-S	R-E-S	R-R-S

Table 3.2 Taxonomy of real-time simulation modes of CDRTP: peripheral unit implemented in the emulator

Controller \ Plant	Simulator (in PC)	Emulator	Real (Hardware)
Simulator (in PC)	S-S-E	S-E-E	S-R-E
Emulator	E-S-E	E-E-E	E-R-E
Real (Hardware)	R-S-E	R-E-E	R-R-E

Table 3.3 Taxonomy of real-time simulation modes of CDRTP: peripheral unit implemented as real hardware

Controller \ Plant	Simulator (in PC)	Emulator	Real (Hardware)
Simulator (in PC)	-	S-E-R	S-R-R
Emulator	-	E-E-R	E-R-R
Real (Hardware)	-	R-E-R	R-R-R

- The first column modes are not realized in CDTRP without using an additional analog interface extension for the PC.

The operating modes in Tables 3.1-3 are obtained depending on the implementation of controller, plant and peripheral unit as the simulator (PC), emulator or real (analog/digital) hardware. Other than S-S-S, E-E-E and R-R-R modes, all modes will be called as mixed modes since at least two of controller, plant and peripheral unit is implemented in different units of CDTRP, i.e. the simulator (PC), the emulator and real hardware. Note that the controllers are applied to the plants in the unity feedback configuration in all modes. However, the modes are by no means restricted to this particular (yet quite general) feedback configuration which means the other possible configurations can be created by making some modification on the introduced GUI. The simulator and emulator of CDRTP can be run faster than real time or run without hard time limitations but these cases (mentioned in Isermann et al., 1999) are out of consideration in the presented work since the focus is on test and design of controllers under the criterion of working well for real time operations.

The first column modes in Table 3.3 can not be realized in CDTRP without using an analog interface feeding the output of the real peripheral unit to the PC where the plant is simulated. (Of course such an extension is possible but with an additional cost.) The actuator, sensors, disturbance and parameter perturbation effects which are created in the peripheral unit using analog/digital hardware can be embedded to the emulator by means of the emulator interface. So, the S-E-R, E-E-R and R-E-R modes of CDTRP constitute a contribution to the real time simulation literature since there

is no such an interface possibility of the simulators running on the PCs and the available emulators in the literature.

R-R-R operating mode corresponds to a real control system not a simulation mode. S-S-S mode where all parts of the control system are simulated in a PC is the most commonly used mode in the literature so that many software tools are available. All of the real time S-S-S, S-E-S, E-S-S, E-E-S, S-S-E, S-E-E, E-S-E and E-E-E operating modes where the controller, plant and peripheral unit are all implemented in the simulator (PC) or possibly in the emulator would be considered as so called "software-in-the-loop simulation" in the literature (Isermann et al., 1999). In a similar way, the modes S-R-S, E-R-S, R-R-S, S-R-E, E-R-E, S-R-R and E-R-R where the plant under test is real would be considered as so called "prototyping mode" (Isermann et al., 1999) and the modes R-S-S, R-E-S, R-S-E, R-E-E and R-R-E where the controller is implemented as real hardware while the plant as simulated would be considered as so called "hardware-in-the-loop" simulation (Dufour et al., 2007; Facchinetti & Mauri, 2009; Hanselmann, 1996; Isermann et al., 1999; Li et al., 2006; Lu et al., 2007; Steurer et al., 2009). It should be noted that giving a single name for different modes yields confusion, so it might be better to use the three-letter based index provided in the Tables 3.1-3 for distinguishing different modes. A different alternative might be the separation of the modes into subclasses such as simulated-controller, simulated-plant, simulated-peripheral, emulated-controller, emulated-plant, emulated-peripheral, real-controller, real-plant or real-peripheral classes of modes each of which refers to a set of modes whose common property is specified by the class name, e.g. the real-plant class consists of all modes where the plant is real but the controller and peripheral might be simulated, emulated or real. Throughout the paper three-letter index will be used for the individual modes and the lastly mentioned classes will be used for the associated set of modes whenever appropriate.

On the other hand, the set of operating modes can be extended by assigning extra letters for each element of peripheral sub-unit, i.e. actuator, sensor, disturbance and perturbation depending on their implementation in simulator or emulator or as real

hardware. Hardware realization for controllers and components of the peripheral unit is application specific, i.e. depending on the application the hardware might be a fully analog system (as in the chaotic synchronization application presented in next Subsections) or a microcontroller or digital signal processor card equipped with analog interface to communicate emulated or real plant.

3.3 Categorization of Modes of CDTRP Based on the Suitability to the Design, Test and Redesign Stages

According to the main issue addressed which is to design controllers having high performance on the real plant under real environmental conditions, a controller design procedure following a three stage path is proposed as follows:

(Initial) Design Stage: Use a simulation mode of CDTRP which enables to implement any kind of controller, plant and peripheral components without any insufficiency in terms of memory and time and with a high flexibility of changing the models and parameters of controller, plant and peripheral components in an efficient way. Of course, the most suitable mode for the initial design stage is the S-S-S. Depending on the control application, on the chosen (initial) set of candidate control methods and also on the experience and knowledge of the control system designer, the modes taking place at the first rows and columns of Tables 3.1-3 can be preferred well in the initial design stage.

Test Stage: Use an operating mode of CDTRP which examines the control methods in terms of their performances under the conditions as close as possible to the real world and so provides a tool for eliminating the controller candidates of poor performances on the real plant under real environment. In other words, the modes suitable to the test stage are the ones which provide real world performance features giving information enough to make a right decision on their usability in real world applications and yet have the flexible implementation efficiency with the ability of creating and changing the models and their parameters for controller, plant and peripheral components, of course in a lesser extent as compared to the initial design

stage. In this sense, the modes taking place at the last rows and columns of Tables 3.1-3 would be preferred in the test stage. However, even S-S-S mode can be used for test purposes to eliminate some controller candidates in the early phases of the design process. On the other hand, before starting the design process one of the first issues which should be clarified is the final operating mode for the control system under consideration. In order to understand beforehand how the real control system will behave under real world conditions, the final test mode should be chosen close to the real operating conditions. Depending on the control application, hardware realization possibilities, plant dynamics and research/educational needs, the final test mode might be chosen as the R-R-R mode or any other mode reflecting the reality sufficiently yet implementable in the laboratory conditions. For instance, R-R-R seems to be the most appropriate choice for final test of the analog chaotic control systems since their simulation and emulation do not reflect reality due to the their large bandwidth dynamics (See also coupled Lorenz systems example in Subsection 3.6.3 studied for another purpose.). As another example, S-R-R mode of CDTRP is chosen for the final test in the control application where the PC is used for hardware implementation of the controller (Facchinetti & Mauri, 2009; Li et at., 2006; Lin, 1997; Lu et al., 2007; Rodriguez & Emadi, 2007; Yamamoto et al., 2009) and also the controller design process for DC motor in Subsection 3.6.2).

Redesign Stage: Redesign may be defined as going back to the design stage for updating models and/or parameters of controller, plant and/or peripheral units and then test new controller candidates in the same or in new environments. In one extreme case, redesign requires to enlarge the set of control methods to be applied. In the other extreme case, it requires tuning the controller parameters only. In the former ones, redesign may end after many loops of design-test-redesign. In this setting, the operating modes which can be used in the redesign stage would be said the ones suitable to the initial design and test stages. However, it can be stated that the operating modes taking place at the last rows and columns of Tables 3.1-3 would more likely be preferred in the redesign stage since a small set of models and parameters remain to be tested, so it is needed less implementation flexibility and also less distance to real world after early phases of the design process.

The above design procedure defines a soft category specifying which operating modes are suitable to the initial design, test and redesign stages. Actually, the suitability of an operating mode to a considered stage of a controller design process is highly dependent on the control application, i.e. on the plant, actuator, sensors and environmental conditions. So, it seems to be impossible to give a crisp category of modes based on their suitability to the design, test and redesign stages. However, for a specific control application, it may be possible to determine the best suitable operating mode to the design, test and redesign stages by successive implementation of some operating modes and then assessing/evaluating the eventually selected controller's performance, so observing which mode yields the controller having the best real world performance.

3.4 Implementation of the Plant Emulator Card with PIC Microcontroller

Microcontrollers which are single-chip computers with limited computer' features (İbrahim, 2006) are widely used for control applications usually for implementing controllers together with digital and analog input interfaces. The plant emulator card of CDTRP has been realized with a PIC18F452 microcontroller. The main reasons for such a choice for the emulator hardware are as follows: i) It is quite cheap and easy to be programmed device, so can be reproducible easily by the instructors and the students for some educational purposes, by the engineers for industrial applications and also by the researchers for examining their controllers on emulated plants. ii) Its capability is enough to implement many benchmark plants and even synchronization systems (linear dynamical and Lorenz oscillators), and also analog and digital interfaces needed to communicate with peripheral units for recreation a real environment in the developed controller-design-test-redesign-platform.

The schematic diagram of the implemented hardware of the plant emulator card is given in Figure 3.3. The PIC18F452 microcontroller based plant emulator card has i) 32KB of internal flash Program Memory, ii) 1536-byte RAM area, iii) 256-byte internal EEPROM, iv) eight channel analog inputs via 10-bit A/D converter, v) four channel digital input/outputs, and vi) two analog output ports via D/A converter. An

LCD character display with four lines and 20 columns is also connected to the microcontroller to indicate the outputs and the control inputs of the emulated plant with respect to current time. Where the PIC18F452 microcontroller is programmed with 5672 lines of C program codes which uses 95% of the ROM and 27% of the RAM of the microcontroller.

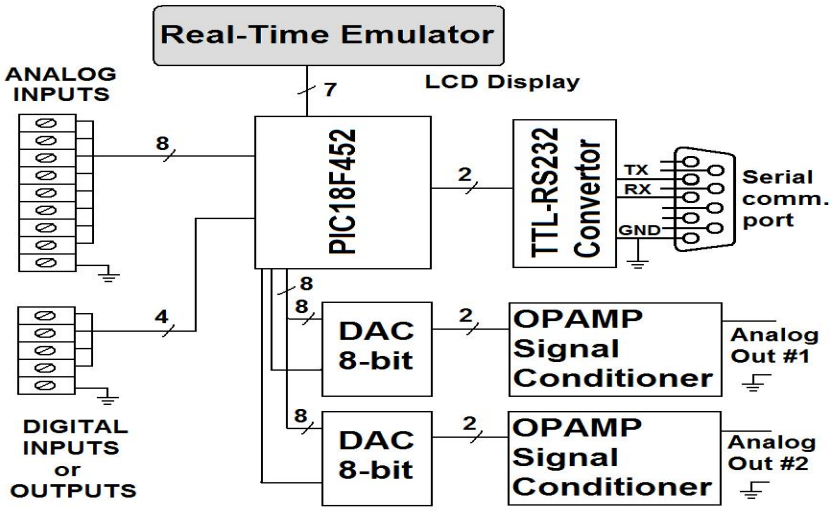


Figure 3.3 The schematic diagram of the hardware of the plant emulator card

An image of the hardware realization of the plant emulator card is given in Figure 3.4 where the physical locations of the blocks depicted in the schematic diagram of the plant emulator card are indicated with description tags.

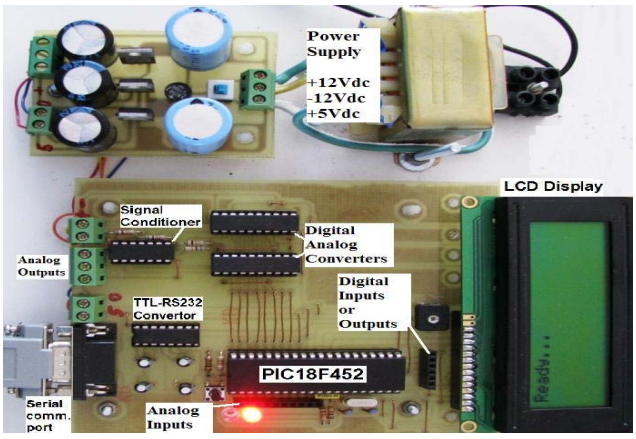


Figure 3.4 The real hardware of the plant emulator card

3.5 Experimental Set-up of the Developed CDTRP

The experimental set-up of the developed CDTRP platform is given in Figure 3.5. It has been used for the design-test-redesigning controllers of the benchmark plants in Table 3.4 by means of operating modes in Table 3.1-3. The CDTRP platform set-up is composed of a plant emulator card, a PC, a signal generator, power supplies and an oscilloscope. The specifications of the plant emulator card are already given in Subsection 3.4. The PC is chosen as having a Centrino processor and a 1GB memory which constitute a minimum configuration necessary for running MS Windows XP and MATLAB 7.04 software. The signal generator as a part of the hardware peripheral unit card is included as providing an external analog signal noise (D_U). Power supplies constituting another part of the hardware peripheral unit card are used as dc variacs changed manually for generating the perturbations of the plant variables (D_X1, D_X2 and D_X3).

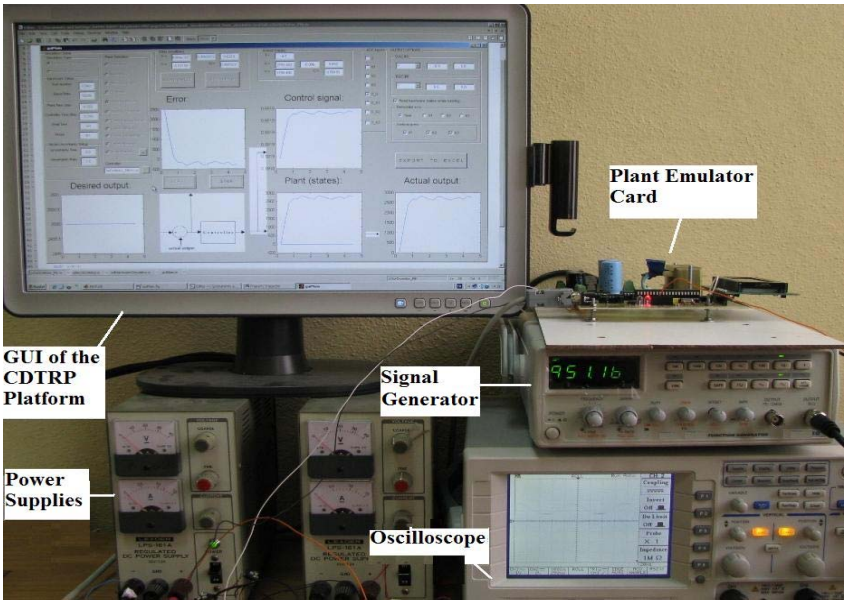


Figure 3.5 The experimental set-up for the whole CDTRP platform

3.6 Verification and Validation of the CDTRP Platform Based on Benchmark Plants

This subsection presents verification and validation of the CDTRP platform based on benchmark plants which are implemented as simulation, emulation or real hardware.

3.6.1 Benchmark Plants Implemented in CDTRP for Verification of Operating Modes:

In this subsection, the benchmark plants which are implemented as simulation, emulation or real hardware for verification of the operating modes of the CDTRP platform are listed. A set of benchmark plants which can be chosen by the users via the plant selection button in the front panel of the GUI are included in the menu of the CDTRP Platform for helping the users to examine their control methods on these selected benchmark plants in a quick way. This menu is open to be extended by the users to cover a larger class of benchmarks or any other plants of interest which must be defined by at most three dimensional state equations (This restriction is due the capacity limit of the chosen emulator hardware, i.e. PIC18F452.).

The benchmark plants available in the menu which are listed in Table 3.4 are used in this presented work also for the verification of the operating modes of the developed controller-design-test-redesign-platform. The benchmark plants in Table 3.4 are chosen among the benchmarks extensively used in the literature in order to compare the system identification methods and/or to compare the controller design methods (It means that they are hard to be identified and/or hard to be controlled.) or they are of practical importance in some sense.

Table 3.4 Benchmark Plants of Control-Design-Test-Redesign-Platform

Benchmark Plant (BP)	State Model	Descriptions and Associated References	Operating Modes
BP1	$x_1(k+1)=\sin(x_1(k))+u(k)(5+\cos(x_1(k)u(k)))$ $y(k)=x_1(k)$	A first-order plant: $u(k)$ control input, $y(k)$ output (Narendra, 1996)	S-S-S, S-E-S, S-E-E, S-E-R
BP2	$x_1(k+1)=0.1x_1(k)+2\frac{u(k)+x_2(k)}{1+(u(k)+x_2(k))^2}$ $x_2(k+1)=0.1x_2(k)+u(k)(2+\frac{u^2(k)}{1+x_1^2(k)+x_2^2(k)}$ $y(k)=x_1(k)+x_2(k)$	A second-order plant: control input $u(k)$, output $y(k)$, states of the plant $x_1(k)$ and $x_2(k)$ (Narendra, 1996; Narendra & Mukhopadhyay, 1997).	S-S-S, S-E-S, S-E-E, S-E-R
BP3	$\ddot{q}_1=-\frac{MgL}{I}\sin q_1-\frac{k}{I}(q_1-q_2)$ $\ddot{q}_2=\frac{k}{J}(q_1-q_2)+\frac{1}{J}u$ $y=q_1$	One-link robot (flexible joint mechanism): control torque input $u(k)$, the output $y(k)$ and the angular positions of the plant $q_1(k)$ and $q_2(k)$. Parameters: Moments of inertia $I=0.031kgm^2$ and $J=0.004kgm^2$, mass-gravity-distance $MgL=0.8Nm$, spring constant $k=31$ Nm/rad (Abdollahi et al., 2006, Ghorbel et al., 1989; Khalil 1996, Slotine & Li, 1991; Spong et al., 1987; Spooner et al. 2002).	S-S-S, S-E-S, S-E-E, S-E-R
BP4	$\dot{x}_1=-\frac{R}{L}x_1-\frac{k_m}{L}x_2+\frac{1}{L}u$ $\dot{x}_2=\frac{k_m}{J}x_1-\frac{b_m}{J}x_2$ $y=x_2$	A DC motor: $u(k)$ control input, $y(k)$ output, $x_1(k)$ and $x_2(k)$ states of the plant. Parameters: $R=5\Omega$, $L=0.5H$, inertia $\mathrm{J}=0.1\mathrm{kg\text{-}m^2}$, damping constant $\mathrm{b_m}=0.2$ N-m/rad/s, and constant $k_m=1V-s$ (Lewis et al., 2004).	S-S-S, S-E-S, S-E-E, S-E-R
BP5	$\dot{x}_1=-2x_1+8448\ u$ $y=x_1$	Realized real micro DC motor: control input $u(k)$, output $y(k)$, states of the plant $x_1(k)$	S-S-S, S-E-S, S-E-E, S-E-R, S-R-S, S-R-R
BP6	$\dot{x}_1=x_2$ $\dot{x}_2=\frac{M+m}{Ml}x_1-\frac{1}{Ml}u$ $y=x_1$	An inverted pendulum: control input $u(k)$, output $y(k)$, states of the plant $x_1(k)$ and $x_2(k)$. Parameters: The parameters: $M=2kg$, mass of the ball $m=0.1kg$, and $l=0.5$ m (Ogata, 1997).	S-S-S, S-E-S, S-E-E, S-E-R
BP7	$\dot{x}_1=x_2$ $\dot{x}_2=-\frac{B}{I}x_1-\frac{r}{I}u$ $y=x_1$	A biomechanical elbow: control input $u(k)$ is difference between flexion muscle F_f and extensor muscle F_e forces, output $y(k)$ states of the plant $x_1(k)$ and $x_2(k)$. Parameters: the inertia of the link $I=0.25kgm^2$, the moment arms $r=0.04$m, and damping constant $B=0.2$ Nms/rad (Micera et al., 1999).	S-S-S, S-E-S, S-E-E, S-E-R
BP8	$\dot{x}_{r1}=-wx_{r2}$ $\dot{x}_{r2}=wx_{r1}$	An oscillator system (Undamped linear pendulum): States of the plant $x_{r1}(k)$ and $x_{r2}(k)$. If the oscillator system's state $x_{r1}(k)$ is chosen as $x_{t1}(k)$ which could be one of the states of another oscillator, this system is used for synchronization of coupled oscillators. Parameter: $w>0$.	S-S-S, S-E-S, S-E-E, S-E-R, R-E-S, R-E-E, R-E-R
BP9	$\dot{x}_{t1}=\sigma(x_{t2}-x_{t1})$ $\dot{x}_{t2}=rx_{t1}-x_{t2}-20x_{t1}x_{t3}$ $\dot{x}_{t3}=5x_{t1}x_{t2}-bx_{t3}$	Realized Lorenz chaotic transmitter: states of the plant $x_1(k), x_2(k)$ and $x_3(k)$. The equations are chosen as in (Cuomo & Openheim, 1993) but with different parameters $\sigma=2.12$, $r=120.5$ and b=10.6 in order to reduce into the main harmonics around 1.5 Hz.	S-S-S, S-E-S, S-E-E, S-E-R, E-R-S, E-R-E, E-R-R
BP10	$\dot{x}_{r1}=\sigma(x_{r2}-x_{r1})$ $\dot{x}_{r2}=rx_{t1}-x_{r2}-20x_{t1}x_{r3}$ $\dot{x}_{r3}=5x_{t1}x_{r2}-bx_{r3}$	Realized Lorenz chaotic receiver: states of the plant $x_1(k), x_2(k)$ and $x_3(k)$. Parameters: $\sigma=2.12$, $r=120.5$ and b=10.6.	S-S-S, S-E-S, S-E-E, S-E-R, R-E-S, R-E-E, R-E-R

3.6.2 Controller-Design-Test-Redesign by CDTRP Platform on a Real Plant: DC Motor Case

This subsection is devoted to demonstrate how to implement the controller design-test-redesign procedure introduced in former Section for reaching a controller working well for a real plant under the real environmental conditions. A micro DC motor is chosen as the real plant and the final operating mode is determined as S-R-R where the PC is used as the controller, the real plant is the micro DC motor, the sensor and actuator i.e. the encoder and driver are realized in a PIC microcontroller driver card (See Figure 3.6). In the initial design stage of the controller design-test-redesign procedure, S-S-S mode was found the most suitable mode as a consequence of the simulation efficiency of the simple yet realistic DC motor models in addition to its implementation flexibility as observed in a set of experiments conducted within the S-S-S, S-E-S and S-E-E modes. In the test stage, S-E-E mode was firstly implemented for testing the considered controller design methods on the emulated plant under emulated environmental conditions. Then, the controller candidates were examined in S-E-R mode. It was found that S-E-R mode is the most suitable mode for testing the considered control methods since it provides to examine the controller candidates under quite realistic conditions still having flexibility in simulating different controller candidates efficiently. In the redesign stage, S-R-S and S-R-E modes were implemented at the beginning and then S-R-R mode was implemented as the final mode where the parameters of the controller found the best in the test stage were tuned. The details of the above explained steps of the implementation of the controller design-test-redesign procedure for the DC motor example are given in the sequel.

Implementation of the real plant together with its real actuator and sensor units: A micro DC motor was considered. The DC motor driver card (See Figure 3.6) was realized with a PIC18F452 microcontroller driver unit which has a serial interface to communicate with the PC. The speed of the motor was considered as the output and was measured by an encoder to feedback to the system. The microcontroller based driver unit drives the DC motor speed revolutions per minute (rpm) via pulse width modulation (PWM) changing between 0% and 100%. That is to say, in the

implemented S-R-S, S-R-E and S-R-R operating modes, control signal from CDTRP converted into PWM signal and can control real DC motor speed rpm.

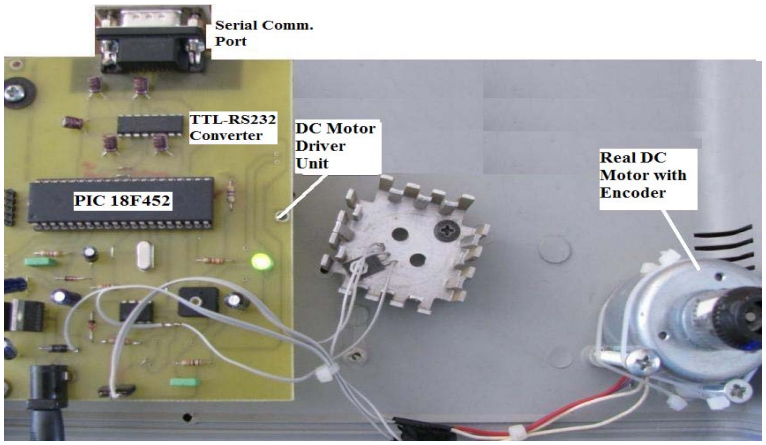


Figure 3.6 Real DC motor hardware whose driver unit is based on PIC for S-R-R mode

Identification of DC motor to obtain a model to be simulated and emulated: In order to simulate and also emulate the real DC motor in the simulated-plant and emulated-plant modes to be used for the design and test stages, a system identification procedure was applied to the real DC motor and then may be the simplest yet realistic model for the considered micro DC motor was obtained. (It should be noted that although more complicated models better suited to the real data measured from the DC motor can be derived, it was preferred to work with this simple model for the plant not only for taking the advantage of efficient simulation and emulation of the DC motor but also for focusing more on testing the controllers' performances on the considered DC motor and on its simulated/emulated model in a comparative way rather than on simulating/emulating the DC motor more realistically by more complicated models. In fact, it was observed that choosing a simple first order dynamical model for the DC motor which was identified by using a step response method provides sufficiently close responses to the ones measured from the real DC motor and also provides to see the differences among the performances of the different controllers implemented.)

A step function, which is obtained by changing PWM sharply, has been fed to the real DC motor plant. The input-output data pairs of the plant necessary for the identification have been measured via the hyper-terminal of the PC. After the process of data gathering, the transfer function of the plant is found under the assumption of single-input single-output (SISO) linear dynamical system for the plant. It is known (Khalil, 1996; Wang, 2009) that the real DC motor plant has been well modeled as a first-order delayed dynamical system according to the response of the plant due to the step input. The step response of the first-order system defined with three parameters is given in the Laplace and time-domain, respectively, as follows.

$$H(s) = \frac{K}{Ts+1} e^{-Ls} \tag{3.1}$$

$$h(t) = K(1 - e^{-(t-L)/T}) \cong K(1 - e^{-t/T}) \tag{3.2}$$

Where, T is the time constant, L is the dead-time and K is the gain (Astrom & Hagglund, 1995). If the dead-time L is too small compared to the time-constant T of the plant, the step response of the system can be approximated as shown in Equation 3.2. Thus, a first order micro DC motor model (BP5 in Table 3.4) has been found according to the measured maximum motor speed 4224 rpm, the time constant 0.5 second and the dead-time 0.011 second. This model has been used for the S-S-S, S-E-S, S-E-E, and S-E-R operating modes of the developed platform CDRTP. Although the first order system is enough to model the real micro DC motor for testing the candidate controllers, a second order model is also identified and implemented for validating the emulator of the platform through examining the emulation performance of the platform with a more realistic model. The considered second-order delay dynamical system model is defined in the Laplace domain as follows.

$$H(s) = \frac{Kw_n^2}{s^2 + 2\zeta w_n s + w_n^2} e^{-Ls} \tag{3.3}$$

$$Po = e^{-\frac{\zeta\pi}{\sqrt{1-\zeta^2}}} \tag{3.4}$$

$$T_s = 3\tau = \frac{3}{\zeta w_n} \tag{3.5}$$

Where, K is the gain, L is the dead-time, ζ is the damping ratio and w_n is the natural frequency. The damping ratio and natural frequency are determined by view of Po overshoot percentage in Equation 3.4 and T_s settling time in Equation 3.5 (Astrom & Hagglund, 1995). The second order micro DC motor model has been found according to the measured Po as 10%, T_s as 4.46 and the dead-time as 0.011 second.

Recreation of disturbance and parameter perturbation effects: The disturbance and parameter perturbation effects which can be embedded to the emulator by means of the signal generator and power supplies components of the hardware peripheral unit were produced in the implemented S-E-R, and S-R-R modes for recreating a real environment. Since there is no such an interface possibility of the simulators running on the PCs and emulators in the simulation/emulation platforms available in the literature, it makes the emulator of the developed CDTRP platform superior to the other platforms.

Controller design-test-redesign stage: DC motor speed tracking problem was chosen as the case. The following four different types of controllers were designed using mainly the S-S-S, S-E-S and S-E-E modes, then tested using S-E-E and S-E-R modes on the identified model BP5 and finally using S-R-S, S-R-E and S-R-R modes on the realized micro DC motor: i) The Proportional-Integral-Derivative (PID) controller designed by Ziegler-Nichols (ZN) method (Astrom & Hagglund, 1995; Ogata, 1997; Ziegler & Nichols 1942), ii) The PID controller designed by Chien, Hrones and Reswick (CHR) method (Chien et al., 1952), iii) The robust controller designed by the Partitioned Robust Control (PRC) method (Craig, 1986) and iv) The

direct adaptive controller designed by the Model Reference Adaptive Control (MRAC) method (Astrom & Wittenmark, 1994).

A PID controller defines the control signal u in terms of the error is given as follows.

$$u = K_p e + K_i \int e dt + K_d \dot{e} \tag{3.6}$$

Where, e stands for the error between the desired and actual output of the plant, $\dot{e}$ for the derivative of the error, K_p for the proportional, K_i for the integral and K_d for the derivative parameters. In the first method, the parameters of the PID controller have been calculated based on the BP5 plant model as $K_p = 54.54$, $K_i = 2479.3$ and $K_d = 0.3$ by using Ziegler-Nichols step response method. In the second method (CHR) which is preferred for yielding minimum overshooting (Astrom & Hagglund, 1995), the PID controller parameters have been calculated (It might be said redesigned!) as $K_p = 43.18$, $K_i = 1635.6$ and $K_d = 0.1995$. The partitioned robust controller is designed as having two separate parts: i) Proportional-Derivative (PD) control u_e and ii) auxiliary control u_y (Craig, 1986; Hsia, 1989). In this method, given a plant model $\dot{x} = -ax + bu$ (e.g. BP5 model), control signal is calculated as follows.

$$u = u_e + u_y = K_v \dot{e} + K_p e - ax - \Delta a x \tag{3.7}$$

Where, K_v and K_p gains related to the control input u_e have been chosen greater than zero and for the auxiliary controller u_y $\hat{a} = 1.2a$ has been chosen for removing the parameter perturbations and plant uncertainties. In the fourth method, the MRAC controller is composed of a first order reference model $\dot{x}_m$, two adaptive controller parameters $\dot{u}_{kr}$, $\dot{u}_{ky}$ and a control signal u as follows

$$\begin{aligned}
\dot{x}_m &= -a_m x_m + b_m r \\
\dot{u}_{ky} &= -\lambda(x_m - x) y a_m (e^{-a_m}) \\
\dot{u}_{kr} &= -\lambda(x_m - x) r a_m (e^{-a_m}) \\
u &= u_{kr} + u_{ky}
\end{aligned} \tag{3.8}$$

Where, *y* stands for the actual output, *r* stands for the reference signal, designed parameter is chosen as $\lambda = 0.5$ and the values of the reference model parameters a_m and b_m are two times the values of the BP5 plant parameters. The equations in (3.8) generate the control signal.

The simulation, emulation and real measurement results obtained along the whole design process implemented by CDTRP: The step responses of the real DC motor and the BP5 implemented in the emulator, which are both controlled by the same controllers designed with the above mentioned methods, for the desired output 2500 rpm are given in Figures 3.7 (a), (b), (c) and (d), respectively. It can be seen from the responses depicted in Figure 3.7 that one may prefer the first two controllers, i.e. the ones designed by ZN and CHR methods since the step responses of the emulator have relatively short rising times and are close to the reference signal, i.e. the step function. However, the responses of the real DC motor controlled by these two controllers for the same step input are not close to the emulated first order model's responses such that the real DC motor demonstrates second order dynamical behavior rather than a first order. (As seen in Figure 3.7 (a)-(b), the responses of the emulated second order system model are much closer to the real DC motor responses.) On the contrary, the responses of the real DC motor and the emulated first order model to the step input are so close to each other for the MRAC case. It can be concluded the followings: i) It can be expected that the real plant will behave similar to the emulator for the PRC and MRAC controllers even if the plant model is poor to reflect the behavior of the real plant. ii) It can be expected that the real plant will behave similar to the emulator for the PID controllers designed by the ZN and CHR methods only when the plant model is realistic as capable of well reflecting the behavior of the real plant. So, the analysis results obtained from the developed CDTRP platform in the operating modes needing the emulator (and also the

simulator) to run are reliable on the realistic plant models for any kind of controller design methods and further reliable even on poor plant models for the PRC and MRAC controller design methods.

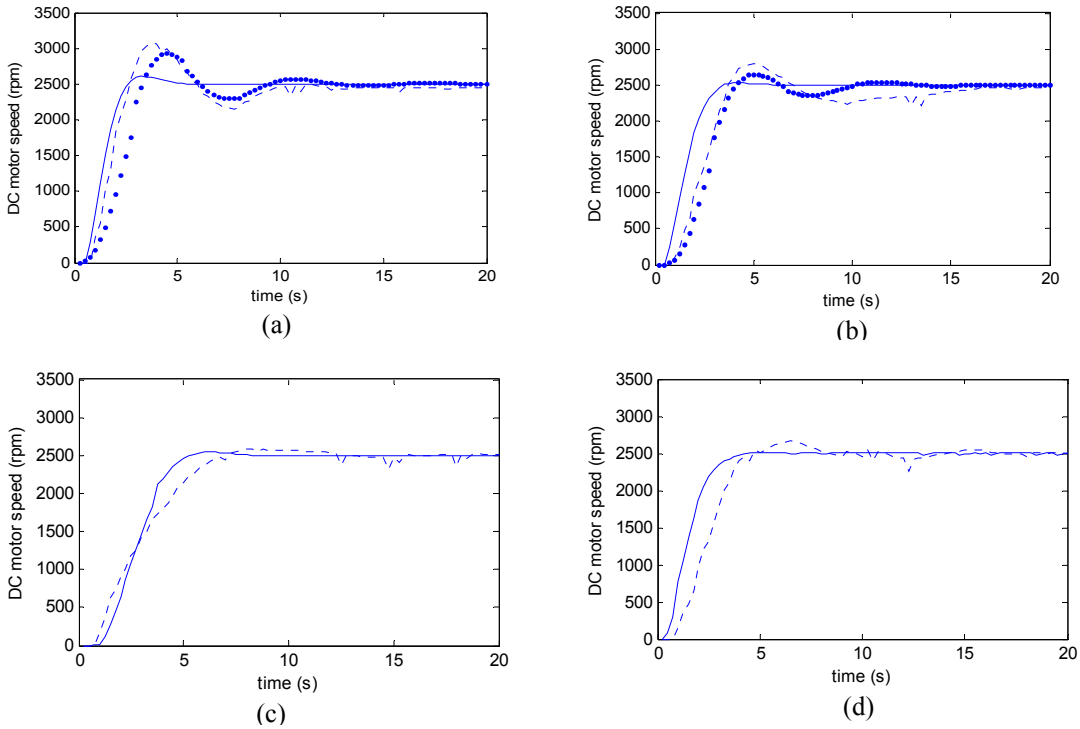


Figure 3.7 Real DC motor S-R-R results (dashed), real-time emulator S-E-S results for 1st order model (solid) and real-time emulator S-E-S results for 2nd order model (dotted) with (a) PID whose parameters are designed by ZN method, (b) PID whose parameters are designed by CHR method, (c) PRC robust controller and (d) MRAC controller.

Recreation of parameter perturbations in the S-E-R mode: Parameter perturbations are created as a multiplicative effect for the model parameters. The perturbation signals are provided by DC power supplies in 0-5 Volt (DC) range and are fed to the hardware peripheral unit card via D_X1 port which is activated by the ADC options in GUI. Where, 1V for D_X1 corresponds to the nominal plant parameters case. The observed responses of the emulated BP5 controlled by the PRC robust controller for four different parameter values are given in Figure 3.8 (a). The responses observed for the MRAC controller case are given in Figure 3.8 (b). In addition, the responses obtained for the PRC and MRAC controllers are compared each other in terms of their performances for two parameter perturbation values in Figure 3.9. No results are depicted for PID controller case since they behave very

poorly due to the parameter variations. It can be said that the responses under parameter variations are close to each other for the PRC and MRAC controllers. As observed in the above analysis part, i) choosing more realistic models yield better emulation results to be obtained by the CDTRP platform and ii) for the MRAC controller and in a lesser extent for the PRC controller cases, the responses of the emulator are very close to the real plant. In the light of these facts, it may be concluded that the performances of the controllers on the real plants under parameter perturbations can be examined using S-E-R mode in such a way by the developed CDTRP platform.

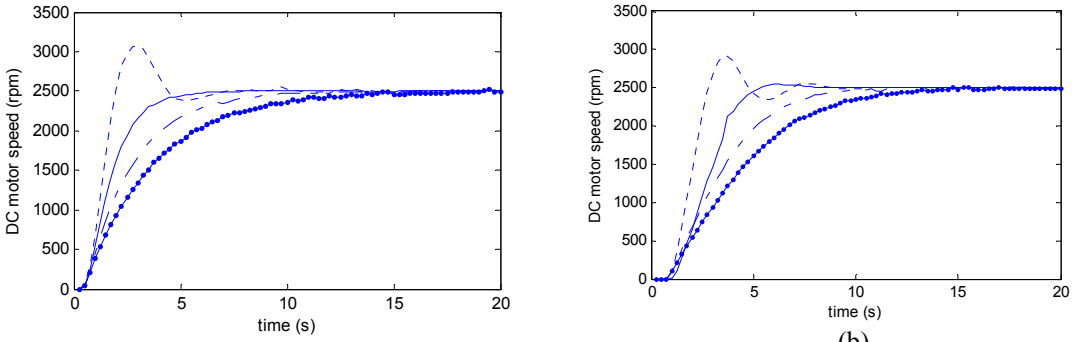


Figure 3.8 The responses of BP5 to the step input for the parameter variation coefficients as 0.5 (dashed,--), 1 (solid,-), 1.5 (dotted-solid,-•) and 2 (dotted on solid,-•-). Note that the coefficient values are the factors multiplying the nominal parameter value to obtain the perturbed parameter. The responses in (a) and (b) are for PRC and for MRAC controllers, respectively.

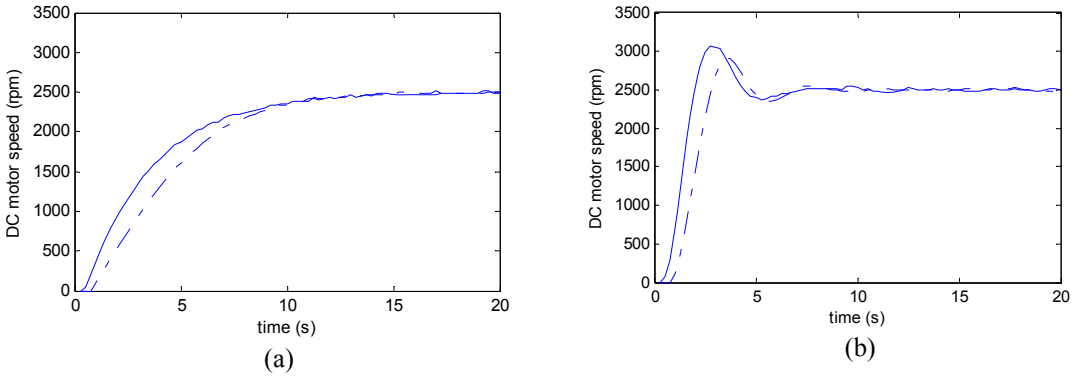


Figure 3.9 The responses of BP5 to the step input for the parameter variation coefficients as 2 in (a) and 0.5 in (b). The responses (solid) and (dotted-solid,-•) were obtained for PRC controller and for MRAC controller, respectively.

Recreation of noise disturbance in the S-E-R mode: Noise disturbances were created as additive to the control signal i.e. input of the plant model. The noise signal was provided by the signal generator whose range is given in Table 3.5. It was fed to the hardware peripheral unit card via D_U port which is activated by the ADC options in GUI. The noise signal amplitude can be scaled via the front panel of the GUI for CDTRP platform applications. For the noise signal du=0.1sin(2πf) with f=1Hz which was created by the noise scaling factor of 0.1 set by using the front panel of the GUI, the corresponding responses obtained for PID, PRC and MRAC controllers are given in Figure 3.10. As expected, CDTRP platform confirms that the PID has a quite poor performance under the applied additive noise despite the PRC and MRAC perform well. It should be noted that the MRAC shows the best performance in the steady state.

Table 3.5 Noise Disturbances Range Scaling

Volt (AC) from Signal Generator	Volt (DC) for ADC
+1V	5V
0V	2.5V
-1V	0V

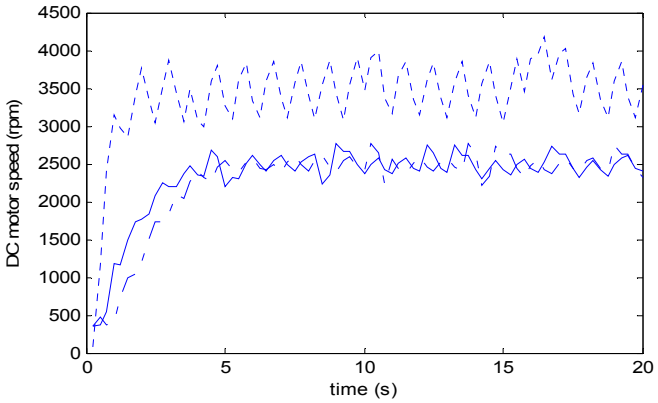


Figure 3.10 Responses of the emulator controlled by PID controller (dashed,--), PRC controller (solid,-) and MRAC controller (dotted-solid,-•) under the noise du=0.1sin(2π1)).

Response to single short-in-time large-in-amplitude pulse disturbance in the S-E-R and S-R-R modes: Recreation of the effects of relatively small amplitude noise disturbances in the emulation are already presented above. Now, another kind of disturbance effect, i.e. single short-in-time large-in-amplitude pulse disturbance effect was created and applied to both of the emulated BP5 model and the real DC motor for a better understanding of the validity of the developed platform in mimicking the behavior of the real plants under real disturbances. The disturbance was created as a multiplicative effect to the output of the model. The pulse time and amplitude were chosen in the front panel of the GUI. In the tests, the pulses were created after the transient regime (e.g. at 10th seconds as shown in Figure 3.11) to mimic the disturbances appearing in the steady state working conditions for the plant. The measured responses of the real DC motor and the emulator controlled by the PID, robust and MRAC controllers are given in Figure 3.11.

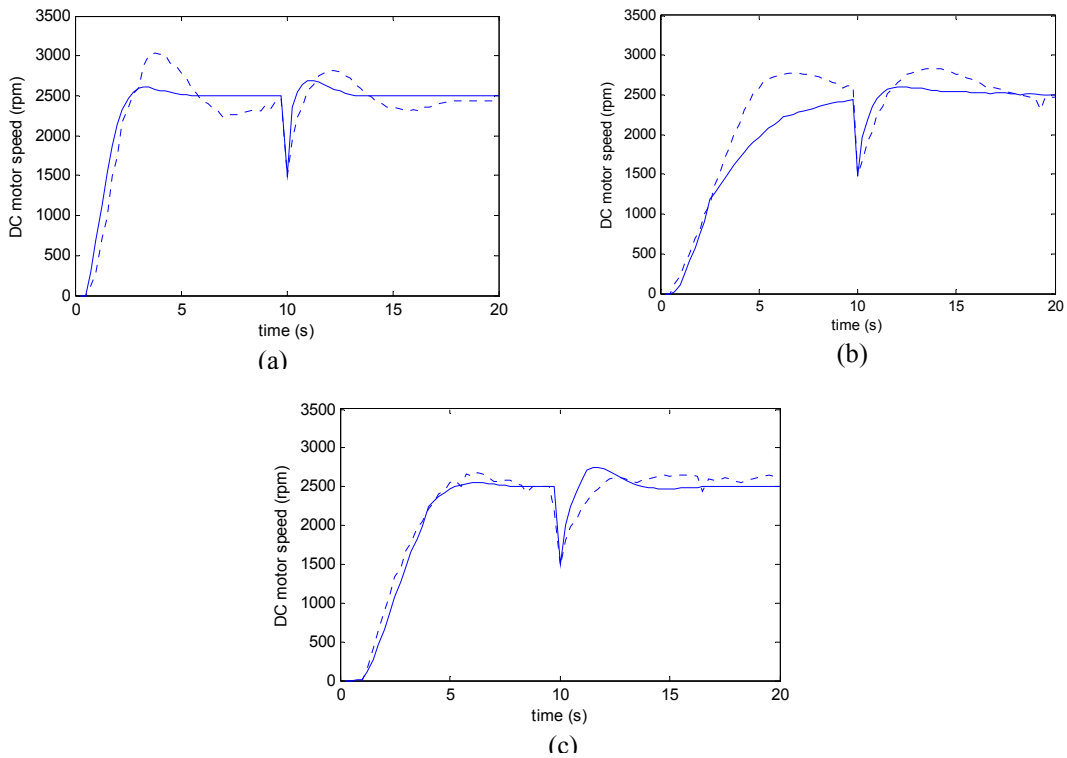


Figure 3.11 Responses of the real DC motor and the emulator controlled by the PID (whose parameters calculated with ZN method), robust and MRAC controllers are given, respectively, in (a), (b) and (c). The pulse amplitude is chosen as k=0.6 and the time when the pulse is applied 10th second. Note that (dashed,--) is for the real plant and (solid,-) for the emulator of CDTRP.

3.6.3 Investigation of Reliable Operating Frequency of Mixed Modes of CDTRP: Coupled Oscillators as Benchmarks

All mixed operating modes of CDTRP require at least two of controller, plant and peripheral components to be implemented in different units of CDTRP, i.e. in two of the simulator (PC), the emulator and hardware peripheral unit. So, the mixed modes need the real time compatibility of the PC, emulator and hardware peripheral unit. In other words, these units have to communicate in a (real time) synchronized fashion. To examine this ability of the CDTRP platform, Lorenz systems based chaotic synchronization is chosen as the benchmark application. As will be seen below, chaotic synchronization is a good example to understand the operating (frequency) limits of the CDTRP platform.

It was observed in a set of experiments conducted that i) the emulator and the simulator can not be synchronized to each other and to an external hardware due to the high frequency dynamics intrinsic to the chaotic (Lorenz) system, ii) as a consequence of the maximum achievable sampling frequency which can be realized in the MATLAB environment used for the PC simulator and in the emulator implemented by using PIC microcontroller, the simulator and the emulator of the developed CDTRP platform can be synchronized to each other when implementing the dynamics up to 25 Hz while the simulator and the emulator can implement the dynamics up to 300 Hz and up to 25 Hz, respectively, if they are operated as uncoupled, and iii) the frequency range of the simulated/emulated system dynamics for which mutual synchronization among the emulator, the simulator and the external hardware are achievable can be enlarged by using a suitable feedback control. (It should be noted that the last (interesting) observation can be interpreted as: The interfaces among the simulator, emulator and external hardware have delays due to the interrupt routines which can be modeled by (complex frequency) poles determining an upper cutoff frequency. And, these poles limiting the frequency range, where the synchronization is achieved, can be shifted to a higher frequency point by using a feedback control.)

In the sequel, the experimental results obtained for a set of different kind implementations of synchronized coupled Lorenz systems are given first, and then, in order to find the limit for the frequency which enables synchronous operation for the implemented dynamics, i.e. the coupled linear undamped pendulums are examined.

Analog Hardware Implementation of Synchronized Lorenz Chaotic Systems: Before examining the CDTRP platform in the E-R-R and R-E-R operating modes for the synchronized system of coupled Lorenz systems, the systems were firstly implemented as analog hardware, i.e. in the R-R-R mode. The master-slave configuration proposed in (Cuomo et al., 1993) was used for the implementation. The transmitter and receiver circuits of the Lorenz chaotic systems, whose scaled state equations are given, respectively, as BP9 and BP10 in Table 3.4, were realized with the circuit configuration in Figure 3.12 using the analog multiplier AD633, opamp LF353 and passive circuit elements (R1,R2,R6,R7=100kΩ, R3,R5,R8,R10=10kΩ, R4,R9=1MΩ, RV1,RV3=100kΩ, RV2,RV4=220kΩ and C1,…,C6=100nF).

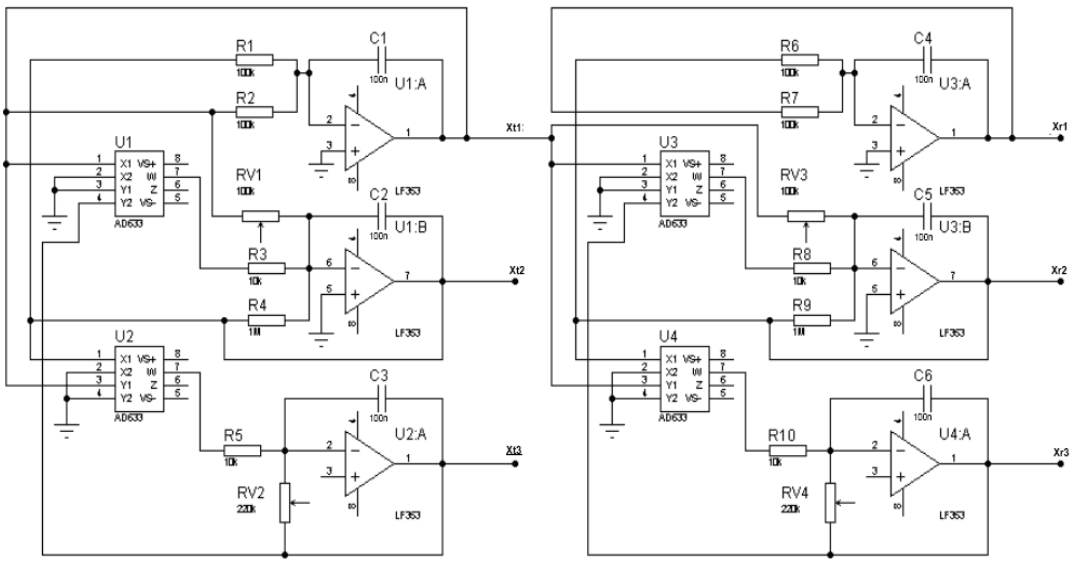


Figure 3.12 Realized analog circuit for master-slave synchronization of Lorenz chaotic systems

(Note that the circuit configuration is chosen as given in (Horowitz, 2009) but the BP9-BP10 coefficients as $\sigma = 10$, $r = 56.6$ and $b = 5.02$ for which Lorenz systems produce chaotic oscillations with the main harmonics around 8.5Hz.) The

synchronization result for the analog hardware realization of the master-slave synchronization of the coupled systems is shown in Figure 3.13.

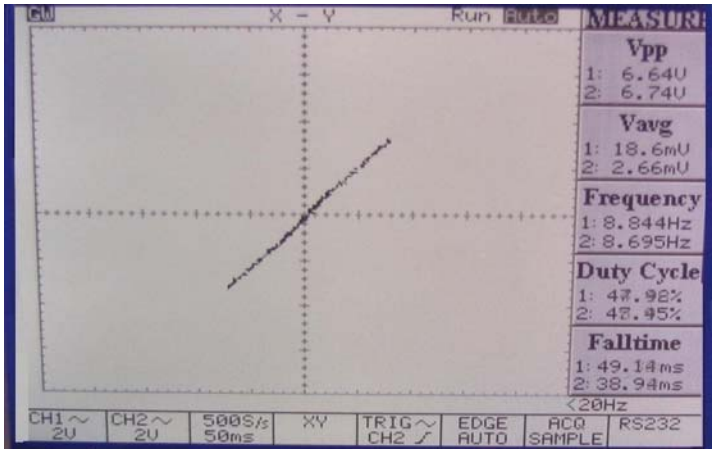


Figure 3.13 A snapshot on the X-Y mode of the oscilloscope where the signals in the X and Y channels are, respectively, the first state variables of the master and slave Lorenz circuits

Synchronization of (Lorenz transmitter) Emulator with a Real Analog (Lorenz Receiver) Plant: As a second implementation of master-slave synchronization of coupled Lorenz systems, the CDTRP platform was operated in the E-R-R mode such that the transmitter BP9 was implemented in the emulator of the CDTRP and the receiver implemented as an analog hardware shown in Figure 3.12 was used as the plant. X1 state was observed from the hardware peripheral unit card via DAC port which was activated by the DAC options in GUI. The implementation of the E-R-R mode is given in Figure 3.14. As seen in Figure 3.15, when operating in the HE-RP mode, the emulator of the CDTRP platform can roughly mimic the chaotic behavior of the Lorenz system. However, during the experiments realized, the master-slave coupled Lorenz systems one of which was realized in the emulator of the CDTRP and the other was realized as the analog hardware, were never observed as synchronized.

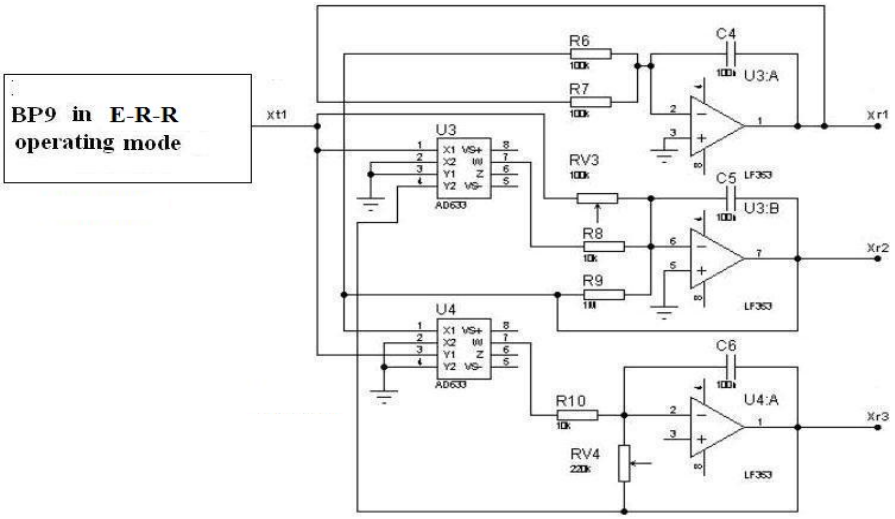


Figure 3.14 Implementation of the E-R-R mode of the CDTRP for the synchronized Lorenz systems

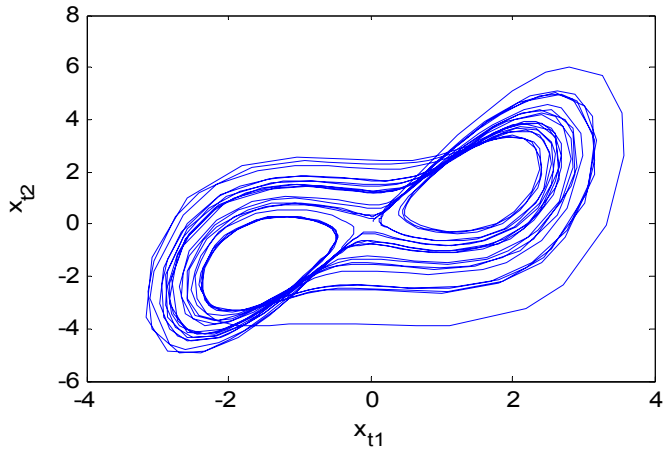


Figure 3.15 Phase portrait for the x_{t1} and x_{t2} states of the emulated BP9 Lorenz chaotic system (Observed via GUI when the CDTRP platform is operated in the E-R-R mode.)

Synchronization of (Lorenz Receiver) Emulator with a Real Analog (Lorenz Transmitter) Hardware: As a third implementation of master-slave synchronization of coupled Lorenz systems, the CDTRP platform was operated in the R-E-R mode such that the receiver BP10 was implemented in the emulator of the CDTRP and the transmitter implemented as an analog hardware shown in Figure 3.12 was used as the master. X1 state was applied to the emulator of the CDTRP via the hardware peripheral unit card by X1 port which was activated by the ADC options in GUI. The implementation of the R-E-R mode is given in Figure 3.16.

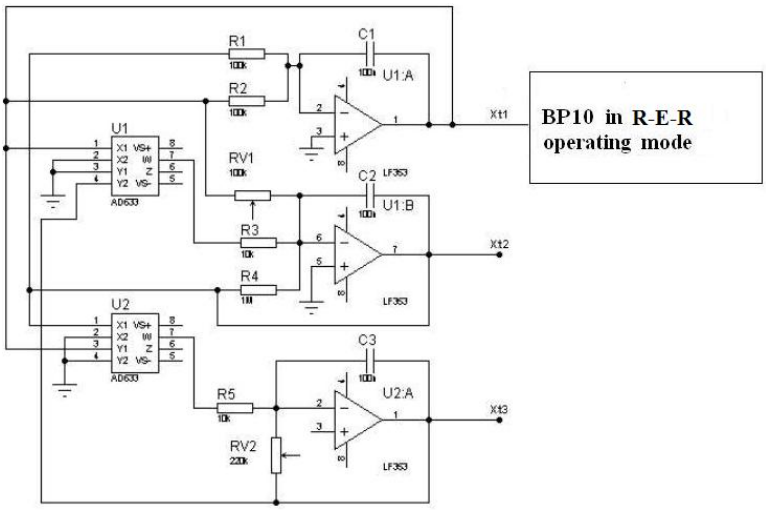


Figure 3.16 Implementation of the R-E-R mode of the CDTRP for the synchronized Lorenz systems

As seen in Figure 3.17, when operating in the R-E-R mode, the emulator of the CDTRP platform fails to mimic the chaotic behavior of the Lorenz system. However, during the experiments realized, the master-slave coupled Lorenz systems one of which was realized in the emulator of the CDTRP and the other was realized in the analog hardware card, were never observed as synchronized.

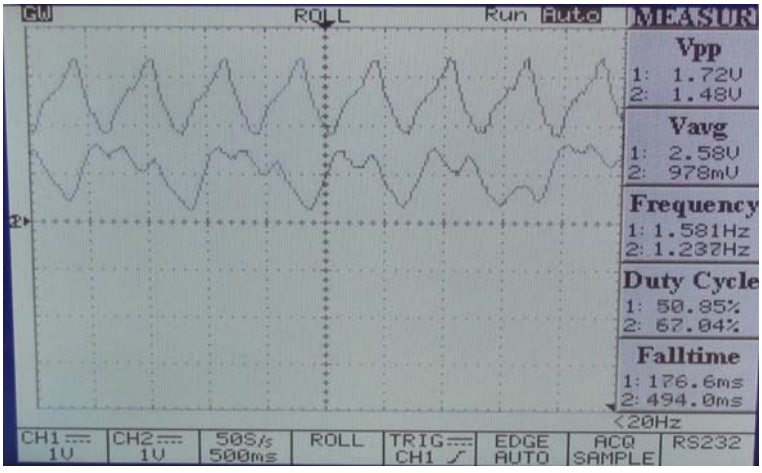


Figure 3.17 First state variables of the analog transmitter and the emulated receiver in the R-E-R mode for synchronized Lorenz systems (The receiver state is the above one; the other is the transmitter state.)

Synchronization of (Lorenz Transmitter) Simulator with (Lorenz Receiver) Emulator: As a fourth implementation of master-slave synchronization of coupled Lorenz systems, the CDTRP platform was operated in the S-E-E mode such that the receiver BP10 was implemented in the emulator of the CDTRP and the transmitter was implemented in the simulator of the CDTRP. As seen in Figure 3.18, although both of the simulator and emulator can roughly mimic the chaotic behavior of the Lorenz system, they fail to be synchronized to each other in the master-slave configuration.

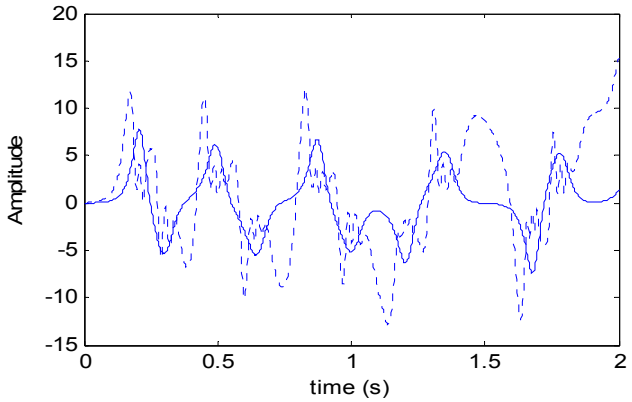


Figure 3.18 Time-waveforms of the first states of the Lorenz transmitter and receiver which are implemented, respectively, in the simulator and the emulator

Effect of Feedback on Chaotic Synchronization: The above implemented master-slave configuration for chaotic synchronization is indeed an open loop control system where the receiver is the plant and the transmitter output is the reference signal to be tracked by this plant. One can argue that not only the lack of implementing the high frequency components of the chaotic signals is the source of failing to get the chaotic synchronization but also the master-slave configuration which, as an open loop control, is sensitive to internal/external disturbances and also to delays, is another source of dissynchronization. In order to clarify this point, in a unity feedback closed loop configuration, a PID controller with the parameters $K_p = 1$, $K_i = 100$ and $K_d = 0.01$ is used to provide a suitable control input to the receiver Lorenz system for deriving its output to track the reference chaotic signal produced by the transmitter

Lorenz system. As seen in Figure 3.19, the PID controller with unity feedback closed-loop configuration provides the desired synchronization for the Lorenz receiver system whose output tracks the reference chaotic signal at least for the main harmonics corresponding to low frequency components.

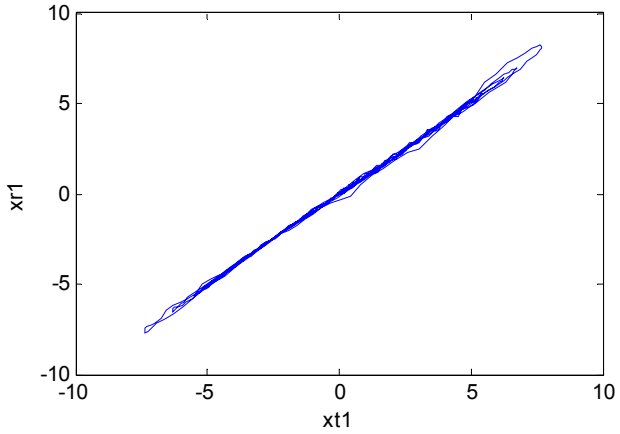


Figure 3.19 First state variable of the receiver Lorenz system as compare to the reference signal which is the first state variable of the transmitter Lorenz system in the S-E-E mode

The above mentioned limits of the CDTRP platform in implementing the synchronized Lorenz systems are natural consequences of the chaotic dynamics of the Lorenz systems intrinsically possessing high frequency components and also the maximum achievable sampling frequencies which can be realized in the MATLAB environment used for the PC simulator and in the emulator implemented by the PIC microcontroller. As seen in the last closed loop implementation of coupled Lorenz systems based on a simple PID controller, the synchronization can be achieved by using a suitable feedback at least for the main (low frequency) harmonics of the chaotic signals. The exact frequency range for the control systems' dynamics which allows synchronous operations of the units of the CDTRP platform in implementing these dynamics was investigated by considering the linear undamped pendulums may be the simplest yet challenging example.

3.6.3.1 The Synchronization of Coupled Linear Undamped Pendulums

Firstly, two identical BP8 models, i.e. the pendulums of which the natural oscillation frequencies are the same are attempted to be synchronized in the open loop master-slave configuration (See Figure 3.20(a)). Then, a PID controller with the parameters $K_p = 1$, $K_i = 100$, and $K_d = 0.001$ in the unity feedback closed-loop configuration is used to control the receiver pendulum to track the output of the transmitter pendulum.

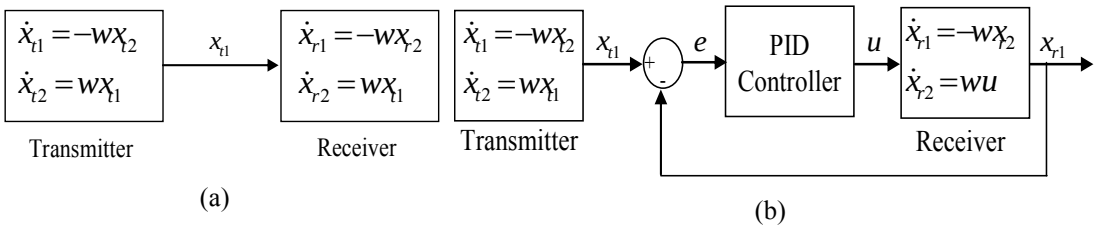


Figure 3.20 Linear undamped pendulums master-slave configurations for S-S-S or S-E-E modes (a) open loop (b) closed loop.

For both of the configurations, the CDTRP platform is operated in the S-S-S mode and also in the S-E-E mode. It is observed that both of the implemented modes yield similar results, so the results obtained for the S-S-S mode are given in Figure 3.21. In the open loop configuration, the synchronization which is achieved for f=1Hz is observed to be failed after f=10Hz. On the other hand, it is observed for the closed loop configuration (See Figure 3.21(b)) that the synchronization is sustained up to f=20Hz which actually determines the limit of the frequency for the control systems dynamics whose real-time implementations in the simulator and the emulator units of the CDTRP platform are reliable in the sense that they can be considered as valid real-time implementations.

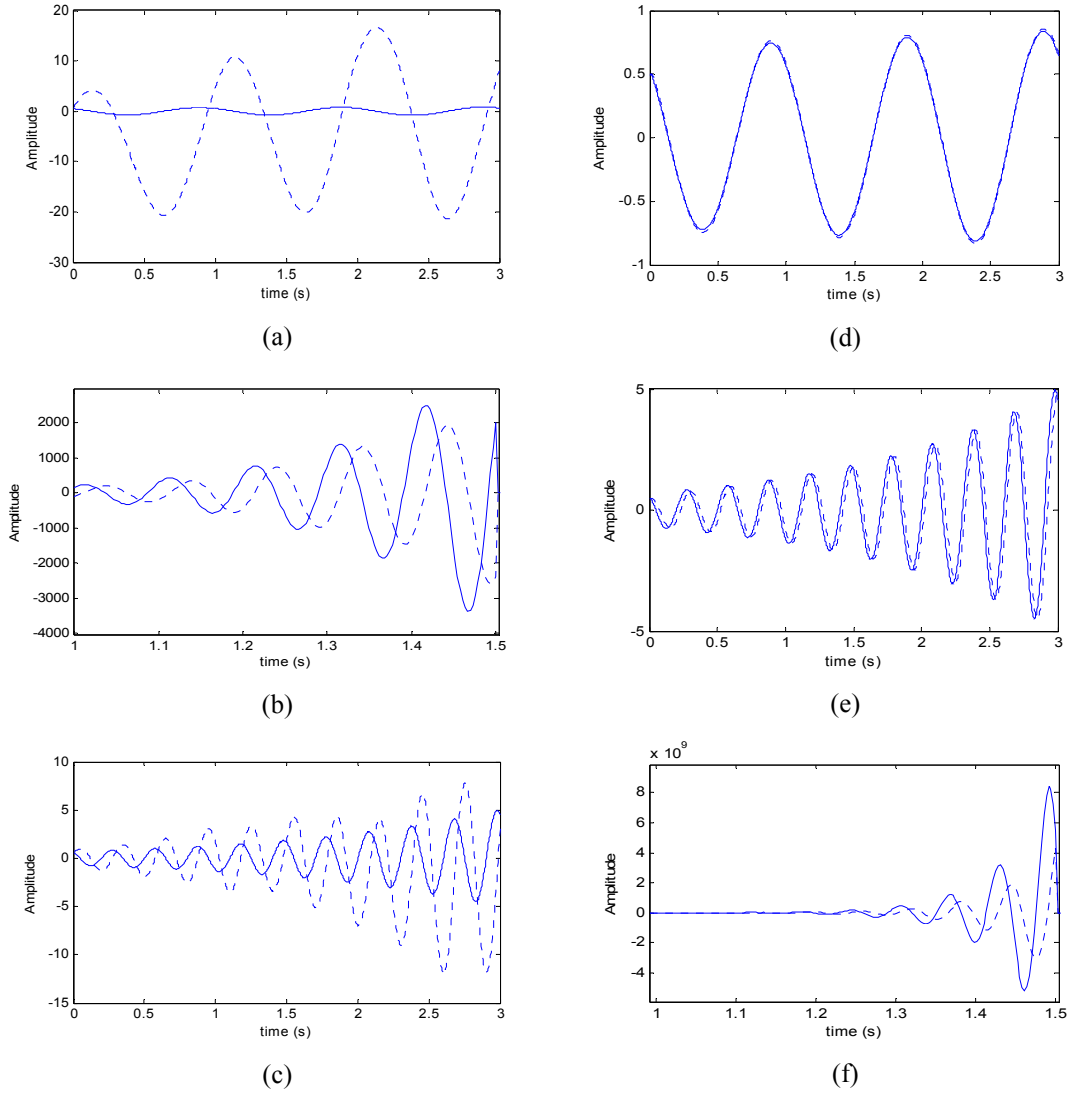


Figure 3.21 (a) Master-slave synchronization for coupled pendulums oscillating at f=1Hz. (b) Master-slave synchronization failure at f=10Hz. (c) Master-slave synchronization failure at f=20Hz. (d) PID based closed-loop synchronization for coupled pendulums oscillating at f=1Hz. (e) PID based closed-loop synchronization for coupled pendulums oscillating at f=10Hz. (f) PID based closed-loop synchronization failure at f=20Hz. Note that (dashed,--) is for the receiver signal and (solid,-) for the (reference) transmitter signal.

3.6.3.2 Synchronization of (Linear Undamped Pendulum) Receiver with (Signal Generator) Transmitter Simulator

In order to determine the reliable operating frequency range for the simulator and emulator when communicating with each other, firstly, a signal generator was used

as a transmitter in a master-slave configuration for deriving the emulator or simulator where BP8 pendulum model was implemented (See Figure 3.22(a).). Then, a PID controller with the parameters $K_p = 1$, $K_i = 100$ and $K_d = 0.001$ in the unity feedback closed-loop configuration was used to control the receiver pendulum to track the output of the signal generator. For both of the configurations, the CDTRP platform was operated in the S-E-E mode and also in the S-S-S mode (See Figure 3.22(b).). It was observed that both of the implemented modes yield almost identical results up to f=25Hz, as shown by the results obtained for the S-S-S mode given in Figure 3.23.

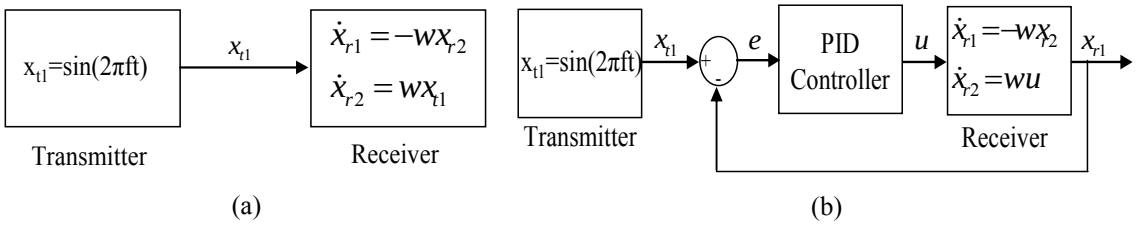


Figure 3.22 (a) Receiver emulator or simulator was derived by a simulated signal generator (x_{t1}) in open loop master-slave configuration for S-S-S or S-E-E modes of the CDTRP platform. (b) Receiver emulator or simulator was derived by a simulated signal generator (x_{t1}) in closed loop for S-S-S or S-E-E modes of the CDTRP platform.

In the open loop configuration, the synchronization which was achieved for f=1Hz was observed to fail after f=10Hz. On the other hand, it was observed for the closed loop configuration that the synchronizations are sustained up to f=100Hz and f=25Hz in the S-S-S mode and the S-E-E mode, respectively. These observations actually determine the limit of the operating frequency for the control systems dynamics whose real-time implementations in the S-E-E and S-S-S modes of the CDTRP platform are reliable in the sense that they can be considered as valid real-time implementations, or say simulations.

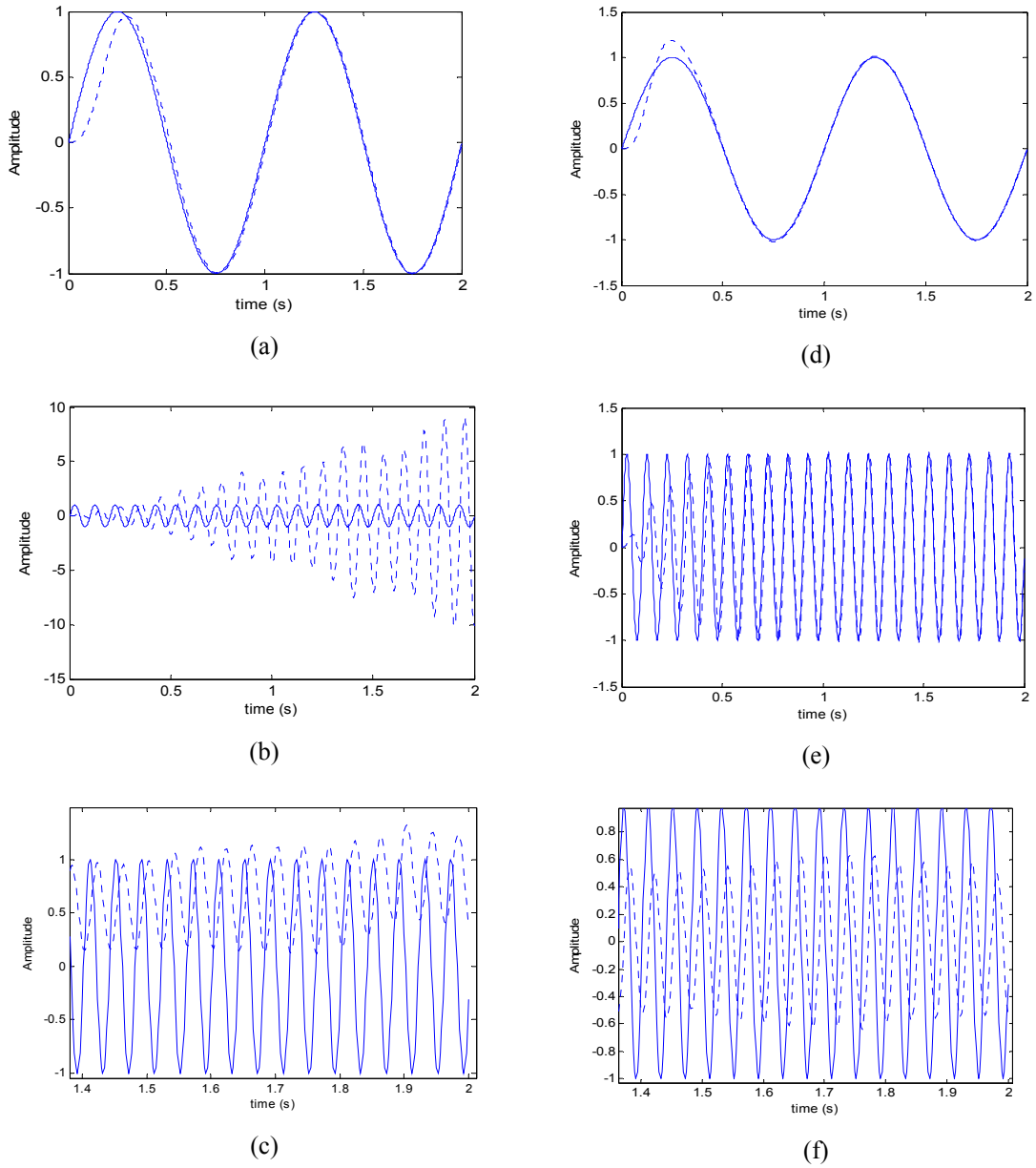


Figure 3.23 (a) Master-slave synchronization at f=1Hz. (b) Master-slave synchronization failure at f=10Hz. (c) Master-slave synchronization failure at f=100Hz. (d) PID based closed-loop synchronization at f=1Hz. (e) PID based closed-loop synchronization at f=10Hz. (f) PID based closed-loop synchronization at f=100Hz. Note that (dashed,--) is for the receiver (pendulum) signal and (solid,-) and for the transmitter (generator) signal in S-S-S mode of CDTRP platform

Synchronization of (linear undamped pendulum) receiver emulator with (signal generator) transmitter: In order to determine the reliable frequency range for the emulator in the R-E-E and R-E-R modes of CDTRP platform where it receives signal from an external analog hardware, the receiver BP8 was implemented in the emulator

and analog signal generator test equipment was used for the transmitter as shown in Figure 3.24. x_t analog signal is applied to the emulator via the hardware peripheral unit card by U port which is activated by the ADC options in the GUI. The difference between the experiments done in S-E-E mode of Figure 3.15 and in the R-E-E and R-E-R modes of Figure 3.24 is in the transmitter part such that in the former one the transmitter realized in the simulator and in the other one the transmitter is an analog hardware. The analog signal received by the emulator and also the output of the pendulum created in the emulator are transferred from the emulator to the GUI. Therefore, the unique additional source of limiting the frequency range of the emulated dynamics in this experiment is the usage of input U port of the emulator supplied by the ADC. As shown in Figure 3.25, it is observed that in the open loop configuration of Figure 3.24(a), the synchronization is achieved up to f=1.52Hz. It should be noted that the frequency range relatively narrower than the one obtained for S-E-E mode is reduced due to the ADC interface of the PIC microcontroller.

Figure 3.24(b) shows that the above given frequency limit can be extended up to f=2.92 Hz (See Figure 3.26) by using a PID controller with the parameters $K_p = 1$, $K_i = 100$, and $K_d = 0.001$ in the unity feedback closed-loop configuration for controlling the receiver pendulum to track the output of the analog signal generator. Where, the PID controller is implemented as analog hardware (Leybold LH 734 06 PID-Controller Lab Equipment is used.), so the CDTRP was operated in the (closed-loop) R-E-E and R-E-R modes.

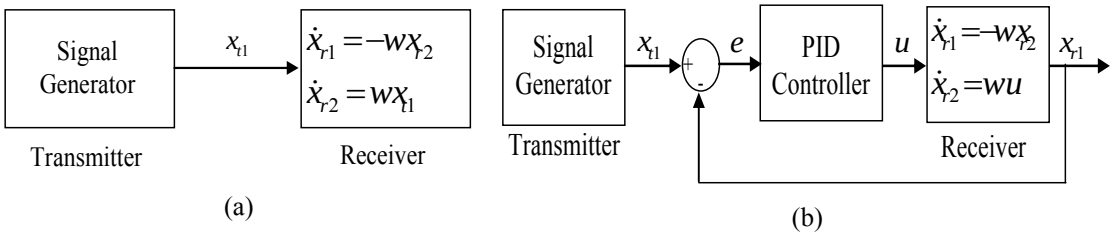


Figure 3.24 (a) Receiver (pendulum) emulator was derived by analog signal generator (b) Receiver (pendulum) emulator controlled by a PID tracks transmitter signal in R-E-E and R-E-R modes.

Note that the frequency limit is lower than f=25Hz which is the one observed for the closed loop (signal generator–pendulum) synchronization in the S-E-E mode since the analog output of the DAC interface of the plant emulator card is also used in addition to the analog input of the ADC interface of the emulator (Observe from Figure 3.24(b) that the analog output of the emulator is fed to the oscilloscope and also to the analog PID hardware.). The above results show that the real-time implementations of the control systems dynamics realized in the R-E-E and R-E-R modes are reliable up to the f=2.92 Hz frequency.

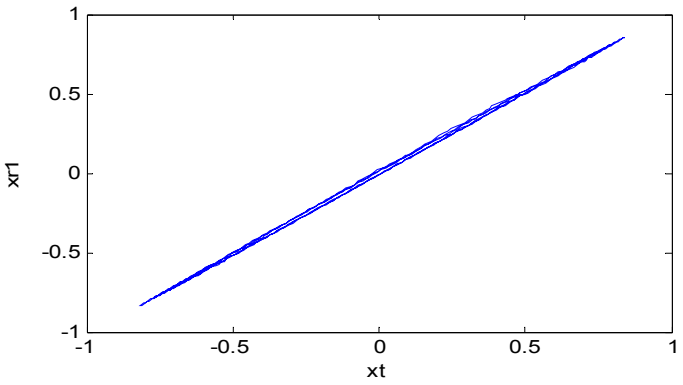


Figure 3.25 Master-slave synchronization between analog generator signal and first state of pendulum realized in the emulator (Observed from data transferred to GUI.)

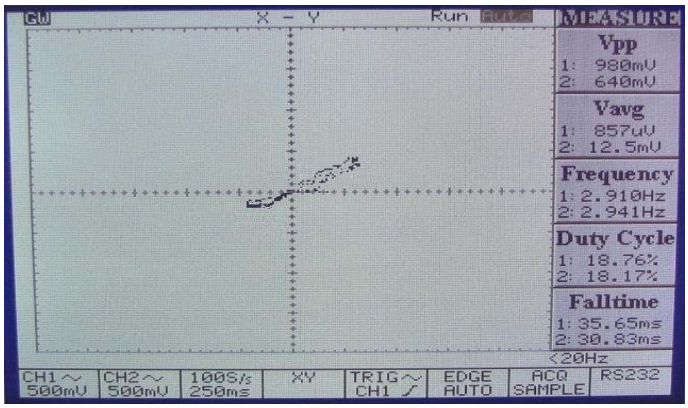


Figure 3.26 A snapshot on the X-Y mode of the oscilloscope where the signals in the X and Y channels are, respectively, the signal generator signal and the pendulum emulator output in the R-E- E and R-E-R modes.

3.6.3.3 Synchronization of a Real Analog (Lorenz Receiver) Hardware with Transmitter Emulator

As the last implementation, the emulator implementing transmitter was used for deriving an analog receiver, i.e. the Lorenz system. Then, the reliable frequency range for the emulator in the E-R-E and E-R-R modes of CDTRP platform was examined. The pure sinusoidal signal generated in the emulator was applied via the DAC interface of the emulator in additive way to the first state of Lorenz system implemented in the hardware peripheral unit card (See Figure 3.27). Note that the Lorenz system is realized by the capacitances for reducing the main harmonics of the chaotic signal around the maximum reliable real time operation frequency of the emulator. The source of limiting the frequency range of the implemented dynamics in this experiment is the usage of analog output of the DAC interface of the plant emulator card. As shown in Figure 3.28 and 3.29, it is observed that the synchronization is achieved up to f=4.05Hz in the open loop configuration of Figure 3.27 and achieved up to f=9.09Hz for the closed loop configuration with a PID controller with the parameters $K_p = 1, K_i = 100$, and $K_d = 0.001$ in the unity feedback in Figure 3.29. (Where, LH 734 06 PID-Controller Lab Equipment is used again.) It should be noted that the Lorenz system derived by the pure sinusoidal transmitter signal is observed no more chaotic Lorenz signal for the large amplitude values of transmitter signal and that the synchronization is indeed achieved for the main harmonic of the disturbed Lorenz signal, so the phase synchronizations seen from Figures 3.28 and 3.30 are not exact due to the sub-harmonics of the disturbed Lorenz signal. The results show that the real-time implementations of the control systems dynamics realized in the E-R-E and E-R-R modes are reliable up to the f=9.09Hz frequency.

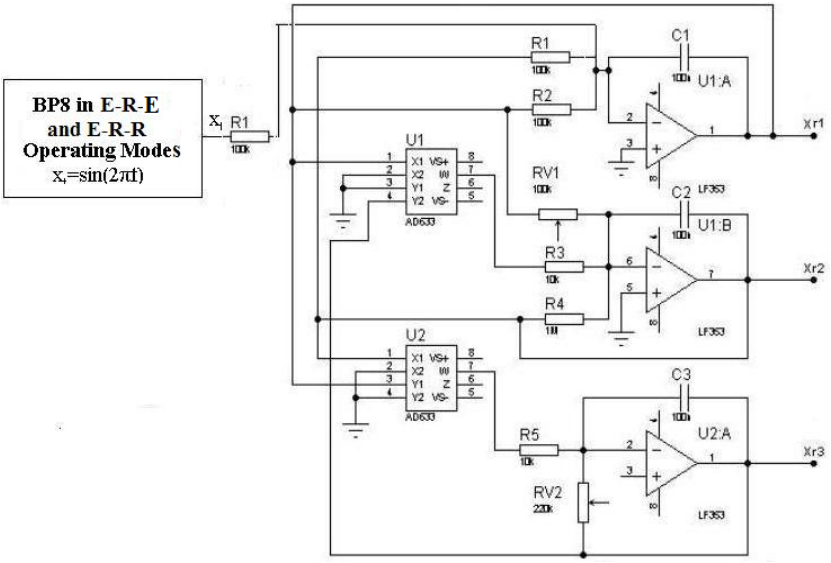


Figure 3.27 Transmitter emulator derives analog Lorenz system receiver in E-R-E and E-R-R modes of the CDTRP (A chaotic state modulation system where message signal is injected to chaotic system as additive to the first state.)

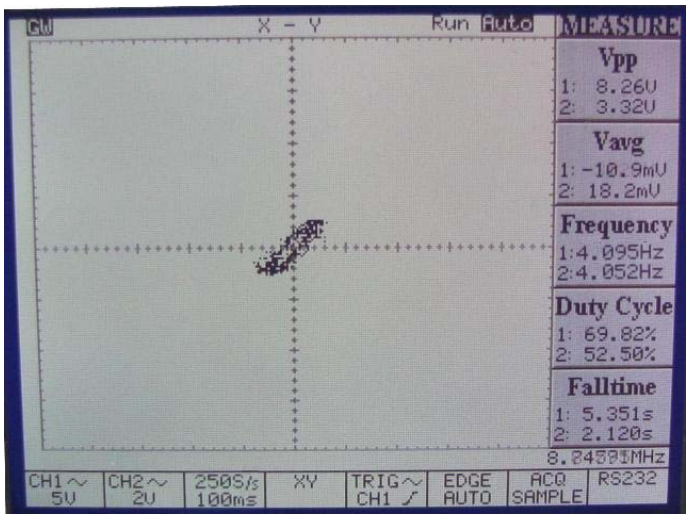


Figure 3.28 A snapshot on the X-Y mode of the oscilloscope where the signals in the X and Y channels are, respectively, the transmitter signal and the first state variable of Lorenz system

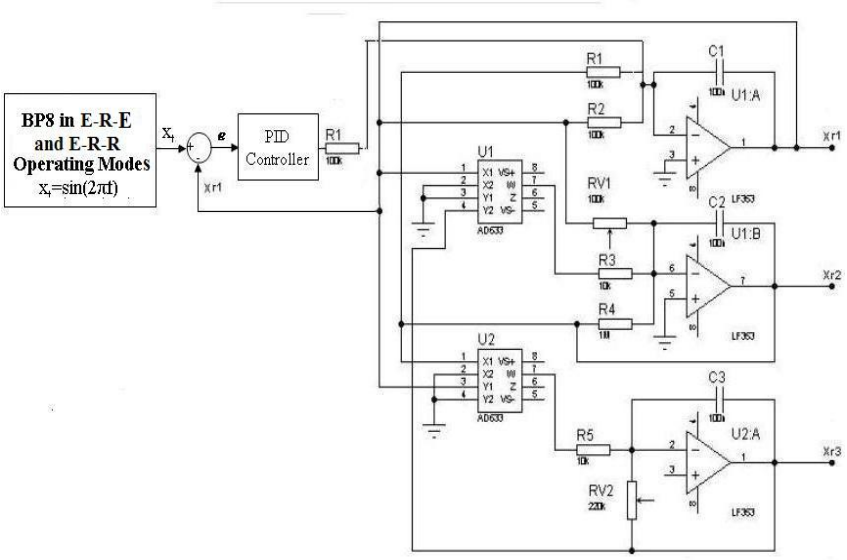


Figure 3.29 PID based closed loop control for Lorenz system receiver to track transmitter emulator's output in E-R-E and E-R-R mode of the CDTRP

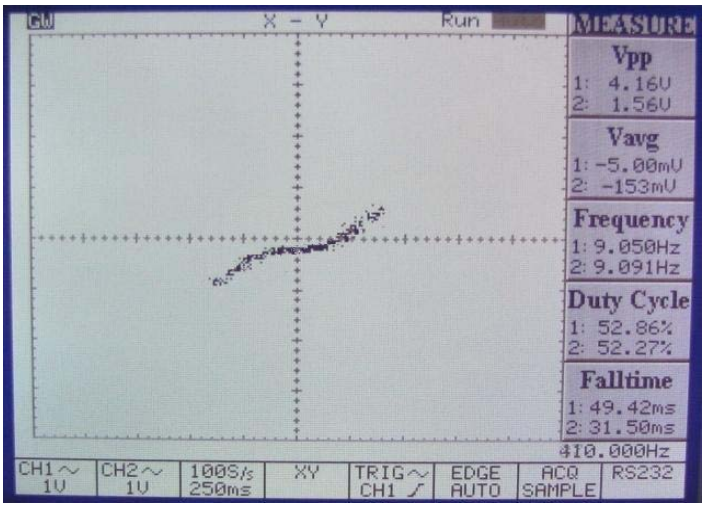


Figure 3.30 A snapshot on the X-Y mode of the oscilloscope where the signals in the X and Y channels are, respectively, the transmitter signal and the first state variable of the Lorenz system in PID based closed loop control

CHAPTER FOUR

LEARNING ALGORITHMS FOR ADAPTIVE NONLINEAR DYNAMICAL CONTROLLER DESIGN

This chapter presents a novel adaptive control method which depends on real plant data. A new input-output data based nonlinear dynamical adaptive controller design method is proposed with adaptive control algorithm. It employs ARMA and NARMA input-output models both for plant and the closed-loop system. In the linear case, it can be viewed as an algorithm solving Diophantine equation in real-time using data measured from the plant not a model of the plant (Astrom, 1987; Astrom & Wittenmark, 1980, 1994, 1997). The proposed adaptive controller has the possibility of implementing as an Artificial Neural Network (ANN) choosing appropriate basis function, e.g. Radial Basis Function Network (RBFN).

The developed adaptive nonlinear dynamical controllers, in particular PID controller, are designed and tested in the CDTRP and also is applied for controlling a real DC motor. The linear and nonlinear versions of the proposed adaptive control method and PID special case together with the ideas and fundamental concepts which the method relies on are given in the following subsections.

4.1 Control System Design as Supervised Learning of Partially Known Systems

A closed loop control system is a partially known system. The known part is the plant to be controlled which is usually given by an identified model with known parameters. The unknown part is the controller. So, design of a control system is equivalent to find controller parameters when feedforward and feedback configurations are already determined in an early phase of the design.

A general feedback control scheme with two degrees of freedom (Astrom, 1987; Astrom & Wittenmark, 1980, 1994, 1997) is depicted in Figure 4.1. The reference $r(k)$ constitutes the closed loop system input and the plant output $y(k)$ constitutes

the actual output of the closed loop system. The desired output $y_d(k)$ of the closed loop system might be a constant or time-varying reference signal $r(k)$ or the output of a reference model $y_m(k)$.

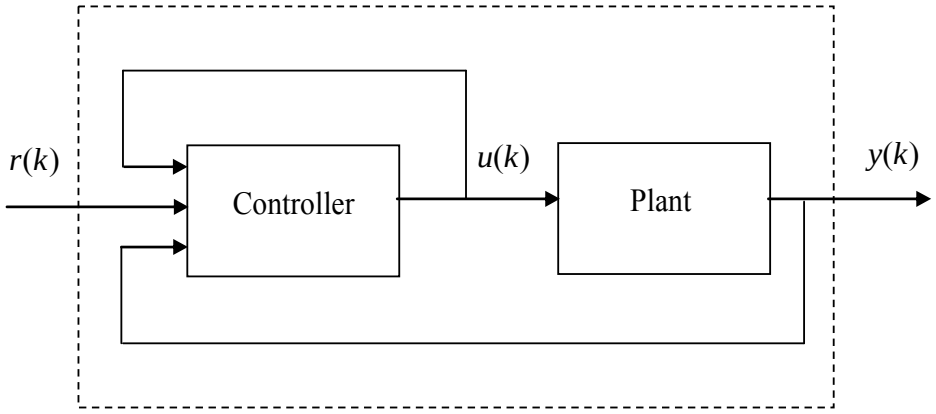


Figure 4.1 General two degrees of freedom control scheme

So, designing control systems of two degrees of freedom can be defined as a supervised learning problem for a partially known input-output system model. In explicit terms, the design of the controller in Figure 4.1 can be posed as the minimization of the error between the desired output $y_d(k)$ and actual output $y(k)$ corresponding to reference input signal $r(k)$ in terms of the parameters of the controller. Such a controller is trained in the closed loop configuration and it becomes, in general, a nonlinear dynamical controller. Moreover, such a controller design approach is suitable for ANN implementations and for adaptive and also robust control schemes to be incorporated in order to handle changes in the plant and environment.

Two successive subsections present, in the linear and nonlinear setting, respectively, the proposed nonlinear dynamical adaptive control method developed based on the approach posing the controller design as supervised learning of a partially known system. It should be noted that the introduced approach is different from all of the inverse system based ANN controller design approaches described in Subsection 2.1.1.4 where the controllers are trained under the open loop

configurations. However, the method has similarities with the model reference adaptive control and self-tuning regulator methods of adaptive control literature where the controller parameters are updated, according to the changes in the plant, in a direct way for the former and in an indirect way for the latter method.

4.2 Linear Case

Assume the following ARMA model for a SISO plant in Figure 4.1.

$$y(k) = \sum_{n=1}^{N} a_n y(k-n) + \sum_{n=0}^{M} b_n u(k-n) \tag{4.1}$$

Equation 4.1 can also be written in an implicit form Equation 4.2 with $a_0 = -1$.

$$\sum_{n=0}^{N} a_n y(k-n) + \sum_{n=0}^{M} b_n u(k-n) = 0 \tag{4.2}$$

Next, assume that the controller with two degrees of freedom in Figure 4.1 has an ARMA representation in Equation 4.2.

$$u(k) = \sum_{m=1}^{P} f_m u(k-m) + \sum_{m=0}^{R} c_m r(k-m) + \sum_{m=0}^{Q} d_m y(k-m) \tag{4.3}$$

Equation 4.2 can also be written in an implicit form with $f_0 = -1$.

$$\sum_{m=0}^{P} f_m u(k-m) + \sum_{m=0}^{R} c_m r(k-m) + \sum_{m=0}^{Q} d_m y(k-m) = 0 \tag{4.4}$$

To obtain an input-output representation for the closed loop system, one can take the weighted sum of the both sides of the plant ARMA representation in Equation 4.2 with f_m weights, so obtain Equation 4.5.

$$\sum_{m=0}^{P} f_m \sum_{n=0}^{N} a_n y(k-m-n) + \sum_{m=0}^{P} f_m \sum_{n=0}^{M} b_n u(k-m-n) = 0 \tag{4.5}$$

The control input can be eliminated using Equation 4.4 and interchanging the sums over *m* and *n* in the second term of Equation 4.5, so the closed ARMA representation is obtained as in Equation 4.6.

$$\sum_{m=0}^{P} f_m \sum_{n=0}^{N} a_n y(k-m-n) + \sum_{n=0}^{M} b_n \left\{ -\sum_{m=0}^{R} c_m r(k-m-n) - \sum_{m=0}^{Q} d_m y(k-m-n) \right\} = 0 \tag{4.6}$$

With newly defined parameters α_n and β_n, the closed loop ARMA representation takes the following form.

$$y(k) = \sum_{n=0}^{\hat{N}} \alpha_n y(k-n) + \sum_{n=0}^{\hat{M}} \beta_n r(k-n) \tag{4.7}$$

Where, $\hat{N} =: \max\{P+N, M+Q\}$ and $\hat{M} =: M+R$. The relations among the closed loop parameters α_n, β_n, the plant ARMA parameters a_n, b_n and the controller parameters f_m, c_m, d_m can be obtained for different choices of P, N, M, R and Q. These relations can be given as in Equations 4.8-4.12, assuming $N = M = P = R = Q$ so $\hat{N} = \hat{M} = 2N$ without loss of generality.

$$\alpha_0 =: 1 + a_0 f_0 - b_0 d_0 \tag{4.8}$$

$$\alpha_i =: \sum_{j=0}^{i} a_j f_{i-j} - \sum_{j=0}^{i} b_j d_{i-j} \quad \text{for } i \in \{1,2,\cdots,N\} \tag{4.9}$$

$$\alpha_i =: \sum_{j=i-N}^{N} a_j f_{i-j} - \sum_{j=i-N}^{N} b_j d_{i-j} \quad \text{for } i \in \{N+1, N+2,\cdots,2N\} \tag{4.10}$$

$$\beta_i =: -\sum_{j=0}^{i} b_j c_{i-j} \qquad \text{for } i \in \{0,1,\cdots,N\} \tag{4.11}$$

$$\beta_i =: -\sum_{j=i-N}^{N} b_j c_{i-j} \qquad \text{for } i \in \{N+1, N+2,\cdots,2N\}. \tag{4.12}$$

The Equations 4.8-4.10 indeed define the well known Diophantine equation (Astrom, 1987; Astrom & Wittenmark, 1980, 1994, 1997). Different control schemes, so called model following control, pole placement, self tuning regulator adaptive control and model reference adaptive control (Astrom, 1987; Astrom & Wittenmark, 1980, 1994, 1997) can be derived based on the closed loop ARMA representation given by Equations 4.7-4.12. The linear version of the adaptive control method proposed in this thesis is also derived from the above closed loop ARMA representation as described in the sequel.

Consider two sets of input-output measurements $\{(u[s,N], y[s,N]\}_{s=1}^{K}$ and $\{(r[s,N], y_d[s,N]\}_{s=1}^{L}$ where $x[s,N] =: [x(s), x(s-1),\cdots, x(s-N)]$ represents the vector of current and past $N-1$ sample values of a signal $x(k)$ at time instant s. The former set is obtained from the plant during an arbitrary time interval $[k, k+K]$ with length K while the latter defines a desired input-output behavior for the closed loop system in an arbitrary time interval $[l, l+L]$ with length L.

Identification of the plant parameters in the time interval $[k, k+K]$ can be realized, in the first stage, by any algorithm minimizing the following identification error in Equation 4.13 in terms of the plant ARMA parameters a_n, b_n.

$$\frac{1}{K}\sum_{s=1}^{K}\left[y(s) - \sum_{n=1}^{N} a_n y(s-n) - \sum_{n=0}^{N} b_n u(s-n) \right]^2 \tag{4.13}$$

On the other hand, the controller parameters can be determined, in the second stage, by minimizing the following closed loop output (tracking) error in Equation

4.14 in terms of the controller parameters f_m, c_m, and d_m which are related to the closed loop parameters α_n, β_n via the Equations 4.8-4.12.

$$\frac{1}{L}\sum_{s=1}^{L}\left[y_d(k) - \sum_{n=0}^{2N}\alpha_n y(s-n) - \sum_{n=0}^{2N}\beta_n r(s-n) \right]^2 \tag{4.14}$$

The second stage, where control parameters are determined, defines a new kind of controller design method which is, indeed, a supervised learning scheme applied on a partially known system since a part of the system parameters, i.e. the plant parameters, is already known in this stage. The introduced controller design method can lead algorithms for determining controller parameters which can be computed in constant and adaptive modes. In the former mode, the plant parameters are determined first by minimizing Equation 4.13 for the whole set of plant input-output measurements and then the controller parameters f_m, c_m, and d_m are found as minimizing Equation 4.14 for the whole set of closed loop system input-output samples. In the adaptive mode, the plant parameters are updated whenever a change in the plant is identified according to K samples $\{(u[s,N], y[s,N]\}_{s=1}^{K}$ and then the controller parameters f_m, c_m, and d_m are updated according to L samples $\{(r[s,N], y_d[s,N]\}_{s=1}^{L}$. Where, K and L are chosen sufficiently large for estimating plant and controller parameters within reasonable accuracies and on the other hand chosen sufficiently small for responding changes in the plant reasonably fast.

4.3 Nonlinear Case

Consider a SISO discrete-time nonlinear plant which is input-output linearizable system. The transformed state model of such a system can be given as in Equation 4.15.

$$
\begin{aligned}
&z_1(k+1) = z_2(k)\\
&z_2(k+1) = z_3(k)\\
&\vdots\\
&z_{n-1}(k+1) = z_n(k)\\
&z_n(k+1) = f_n(z_1(k), z_2(k), \cdots, z_n(k)) + g_n(z_1(k), z_2(k), \cdots, z_n(k))u(k)\\
&y(k) = z_1(k)
\end{aligned}
\tag{4.15}
$$

The input-output relation corresponding to Equation 4.15 is given in Equation 4.16.

$$y(k+n) = f(y(k), y(k+1), \cdots, y(k+n-1)) + g(y(k), y(k+1), \cdots, y(k+n-1))u(k) \tag{4.16}$$

The NARMA system in Equation 4.16 can be rewritten as in Equation 4.17.

$$y(k) = f(y[k-1, n]) + g(y[k-1, n])u(k-n) \tag{4.17}$$

Where, $y[k-1, n] =: [y(k-1), y(k-2), \cdots, y(k-n)]$ and $g(y[k-1, n]) \neq 0$, so there exists $\alpha(\circ): R^n \to R$ and $\beta(\circ): R^n \to R$ providing the input in terms of the current and future outputs as given in Equation 4.18.

$$u(k) = \alpha(y[k+n-1, 0]) + \beta(y[k+n-1, 0])y(k+n) \tag{4.18}$$

Where $y[k+n-1, 0] =: [y(k+n-1), y(k+n-2), \cdots, y(k)]$.

It should be noted that Equation 4.17 and also 4.18 require exact knowledge of the degree n of the plant. In order to have a practically more meaningful identification, Equation 4.17 and 4.18 are extended, in this thesis, into the following ones. Where, N is chosen not less than n based on some knowledge about the upper bound for the degree of the plant.

$$y(k) = f(y[k-1,N]) + \sum_{i=0}^{N} g_i(y[k-1,N])u(k-i) \tag{4.19}$$

$$u(k) = \alpha(y[k+N-1,0]) + \sum_{i=0}^{N} \beta_i(y[k+N-1,0])y(k+i) \tag{4.20}$$

Now, further assume that $f(\circ): R^n \to R$, $g(\circ): R^n \to R$, $\alpha(\circ): R^n \to R$, $\beta(\circ): R^n \to R$, $g_i(\circ): R^n \to R$ and $\beta_i(\circ): R^n \to R$ are all in the range of a set of basis functions $\{\varphi_j\}_{j=1}^{J}$, so the following NARMA representations in Equations 4.21-4.24 are obtained.

NARMA-I: $$y(k) = \sum_{j=1}^{J} a_j \varphi_j(y[k-1,n]) + \sum_{j=1}^{J} b_j \varphi_j(y[k-1,n])u(k-n) \tag{4.21}$$

NARMA-II: $$u(k) = \sum_{j=1}^{J} a_j \varphi_j(y[k+n-1,0]) + \sum_{j=1}^{J} b_j \varphi_j(y[k+n-1,0])y(k+n) \tag{4.22}$$

NARMA-III: $$y(k) = \sum_{j=1}^{J} a_j \varphi_j(y[k-1,N]) + \sum_{i=0}^{N}\sum_{j=1}^{J} b_{ij} \varphi_{ij}(y[k-1,N])u(k-i) \tag{4.23}$$

NARMA-IV: $$u(k) = \sum_{j=1}^{J} a_j \varphi_j(y[k+N-1,0]) + \sum_{i=0}^{N}\sum_{j=1}^{J} b_{ij} \varphi_{ij}(y[k+N-1,0])y(k+i) \tag{4.24}$$

It should be noted that $\{\varphi_j\}_{j=1}^{J}$ may be the basis functions of a well known transformation, e.g. wavelet, or a function approximation, e.g. Gaussian or sigmoidal functions which are used ANN based implementations.

Based on the above four NARMA models for the plant which are all originated from input-output linearization and considering some special form of the following general NARMA representation for the controller, several closed loop representations can be obtained by eliminating the control input terms.

$$\sum_{m=0}^{J} f_m \varphi_m(u[k,N]) + \sum_{m=0}^{J} c_m \varphi_m(r[k,N]) + \sum_{m=0}^{J} d_m \varphi_m(y[k,N]) = 0 \tag{4.25}$$

In the sequel, controller NARMA is assumed to have the following form.

$$u(k) = -\sum_{m=0}^{J} c_m \varphi_m(r[k,N]) - \sum_{m=0}^{J} d_m \varphi_m(y[k,N]) \tag{4.26}$$

Now, further assuming NARMA-III model for plant, one can obtain in Equation 4.27.

$$\begin{aligned} y(k) = & \sum_{j=1}^{J} a_j \varphi_j(y[k-1,N]) \\ & - \sum_{i=0}^{N} \sum_{j=1}^{J} b_{ij} \varphi_{ij}(y[k-1,N]) \left(\sum_{m=0}^{J} c_m \varphi_m(r[k-i,N-i]) + \sum_{m=0}^{J} d_m \varphi_m(y[k-i,N-i]) \right) \end{aligned} \tag{4.27}$$

The relation Equation 4.27 can be rewritten as in Equation 4.28 in order to see more clearly that Equation 4.27 gives a NARMA representation for the closed loop system such that the system output is linear in terms of the controller parameters.

$$\begin{aligned} y(k) = & \sum_{j=1}^{J} a_j \varphi_j(y[k-1,N]) - \sum_{i=0}^{N} \sum_{m=0}^{J} \sum_{j=1}^{J} d_m b_{ij} \varphi_{ij}(y[k-1,N]) \varphi_m(y[k-i,N-i]) \\ & - \sum_{i=0}^{N} \sum_{m=0}^{J} \sum_{j=1}^{J} c_m b_{ij} \varphi_{ij}(y[k-1,N]) \varphi_m(r[k-i,N-i]) \end{aligned} \tag{4.28}$$

In a similar way to the linear case studied in Subsection 4.2, in the first stage one can minimize the following identification error Equation 4.29 in terms of the plant NARMA parameters a_j and b_{ij} for a given set of plant input-output measurements $\{(u[s,N], y[s,N]\}_{s=1}^{K}$

$$\frac{1}{K}\sum_{s=1}^{K}\left[y(s)-\sum_{j=1}^{J}a_j\varphi_j(y[k-1,N])-\sum_{i=0}^{N}\sum_{j=1}^{J}b_{ij}\varphi_{ij}(y[k-1,N])u(k-i)\right]^2 \tag{4.29}$$

So, in the second stage, one can minimize the following closed loop output (tracking) error in Equation 4.30 in terms of the controller NARMA parameters c_m and d_m for a given set of (reference input)-(desired output) sample pairs $\{(r[s,N],y_d[s,N]\}_{s=1}^{L}$.

$$\begin{aligned}\frac{1}{L}\sum_{s=1}^{L}\Bigg[& y_d(k)-\sum_{j=1}^{J}a_j\varphi_j(y[k-1,N])-\sum_{i=0}^{N}\sum_{m=0}^{J}\sum_{j=1}^{J}d_m b_{ij}\varphi_{ij}(y[k-1,N])\varphi_m(y[k-i,N-i]) \\ & -\sum_{i=0}^{N}\sum_{m=0}^{J}\sum_{j=1}^{J}c_m b_{ij}\varphi_{ij}(y[k-1,N])\varphi_m(r[k-i,N-i])\Bigg]^2\end{aligned} \tag{4.30}$$

4.4 Convergence and Stability Issues

The linear, nonlinear, constant and adaptive versions of the methods developed in the previous two sections should also be analyzed further in regards of the convergence of a variety of algorithms originated from these methods and in regards of the stability of the resulting closed loop systems.

Since both of the identification and tracking errors are quadratic and convex in terms of the optimization variables, i.e. the plant and, respectively, controller parameters, then offline and online gradient descent algorithms converge to the minimum that is desired to be found for sufficiently small step sizes.

Stability can be studied either in the BIBO sense, for instance, applying small gain theorem or in the Lyapunov sense in different ways. In both senses, the stability analysis is not a trivial issue: Finding gains in BIBO analysis and finding a Lyapunov function which is valid for the most general closed loop model (4.28) either in fixed

plant and controller parameter case or in time-varying plant and/or controller parameter case.

Another approach to study the Lyapunov stability is to determine the contractivity of the discrete time mapping in Equation 4.28 in different update modes and different special cases. In this direction one can attempt to find the Lipschitz constant of the closed loop model in Equation 4.28 at least for some special cases in an analytical way.

4.5 Simulation Results

4.5.1 Proposed Adaptive Controller versus MRAC

In this example, MRAC and the proposed adaptive controllers are compared to each other in terms of their performances on a very simple tracking problem for conceptual understanding difference between two methods.

Consider the linear dynamical first-order system given by Equation 4.31.

$$\dot{y} = -a_p y + b_p u \tag{4.31}$$

Where a_p and b_p stands for the plant parameters, u is the control input and y is the output of the system (Astrom & Wittenmark, 1994). For MRAC controller design, a stable plant model given in Equation 4.32 is considered as the reference model.

$$\dot{y}_m = -a_m y_m + b_m r \tag{4.32}$$

The reference model has the transfer function in Equation 4.33 which is given for simulation purposes (See Figure 4.2.).

$$M = \frac{y_m(s)}{r(s)} = \frac{b_m}{s + a_m} \tag{4.33}$$

Controller signal is selected as a function depending on the reference input and the actual output.

$$u = k_r r + k_y y \tag{4.34}$$

Where, the controller parameters k_r and k_y are subject to be updated. Optimum controller parameters k_r and k_y can be found as matching the closed loop system obtained by the control input in Equation 4.34 to the stable reference system:

$$\dot{y} = -(a_p - b_p k_y) y + b_p k_r r \leftrightarrow \dot{y}_m = -a_m y_m + b_m r \tag{4.35}$$

$$k_y^* = \frac{a_m - a_p}{b_p} \qquad k_r^* = \frac{b_m}{b_p}$$

Now, MRAC law responding plant parameter changes can be derived by considering tracking error equation as follows. Let $\hat{k}_r$ and $\hat{k}_y$ be estimates for control parameters. Let $\tilde{k}_r =: \hat{k}_r - k_r^*$ and $\tilde{k}_y =: \hat{k}_y - k_r^*$ be the controller parameter estimation errors and $e =: y - y_m$ be the tracking error. So, the tracking error equation and its steady-state solution can be found as in Equation 4.36.

$$\begin{aligned} \dot{e} &= \dot{y} - \dot{y}_m = -(a_p - b_p \hat{k}_y) y + b_p \hat{k}_r r - (-a_m y_m + b_m r) =: -a_m e + a_m \tilde{k}_y y + a_m \tilde{k}_r r \\ e(t) &= a_m e^{-a_m} \tilde{k}_y y + a_m e^{-a_m} \tilde{k}_r r \end{aligned} \tag{4.36}$$

Where the first term in the steady-state tracking error is due to the estimation error $\tilde{k}_y$ and the second is due to $\tilde{k}_r$. So, the update rules for the controller parameters are derived as given in Equation 4.37.

$$\dot{\hat{k}}_y = -\gamma e a_m e^{-a_m} y, \quad \dot{\hat{k}}_r = -\gamma e a_m e^{-a_m} r \tag{4.37}$$

The overall MRAC system has the block diagram in Figure 4.2.

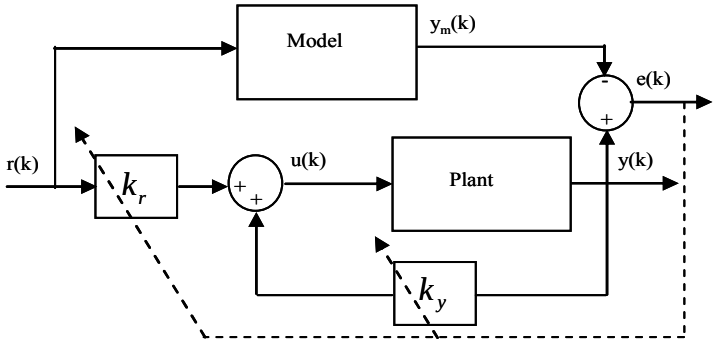


Figure 4.2 MRAC controller with two parameters update

The simulation block diagram for the plant parameters $a_p = 1$ and $b_p = 0.5$, the stable plant model with $a_m = 2$ and $b_m = 2$ is depicted in Figure 4.3 and the simulated time waveforms are given in Figure 4.4 (Astrom & Wittenmark, 1994).

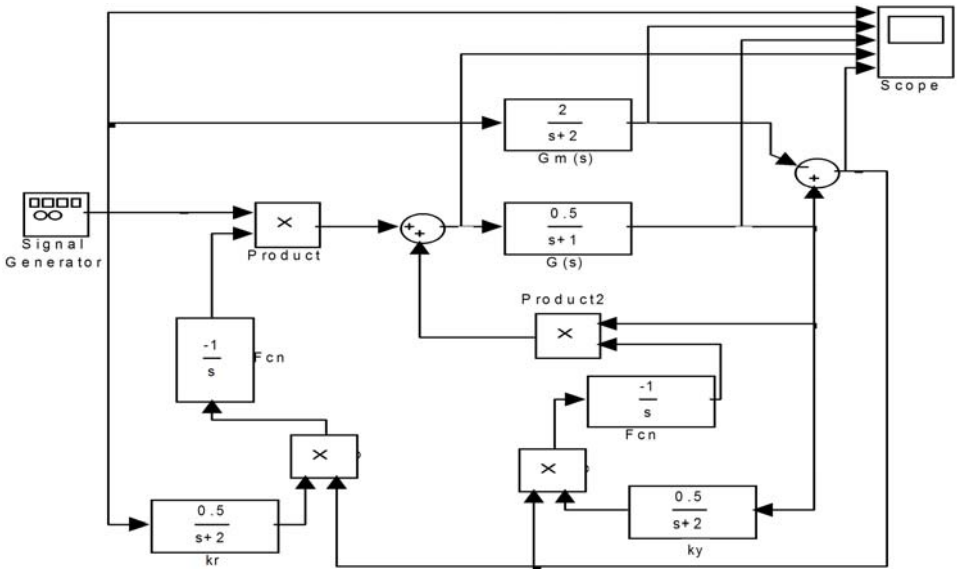


Figure 4.3 MRAC for first order plant (Astrom & Wittenmark, 1994)

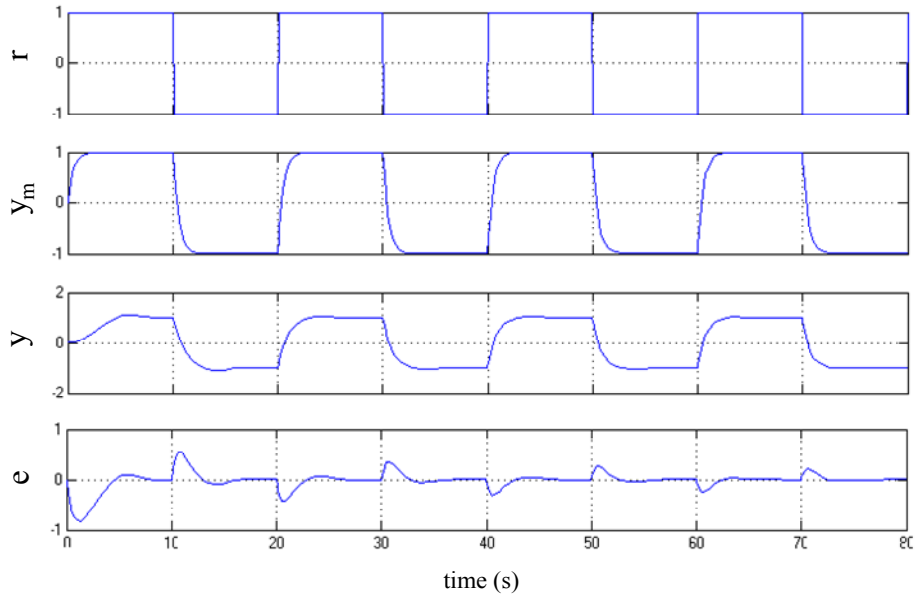


Figure 4.4 Time-waveforms of the MRAC for first order plant

Now, consider the linear version of the proposed adaptive control method by the following control law with two parameters.

$$u(k) = c_0 r(k) + d_0 y(k) \tag{4.38}$$

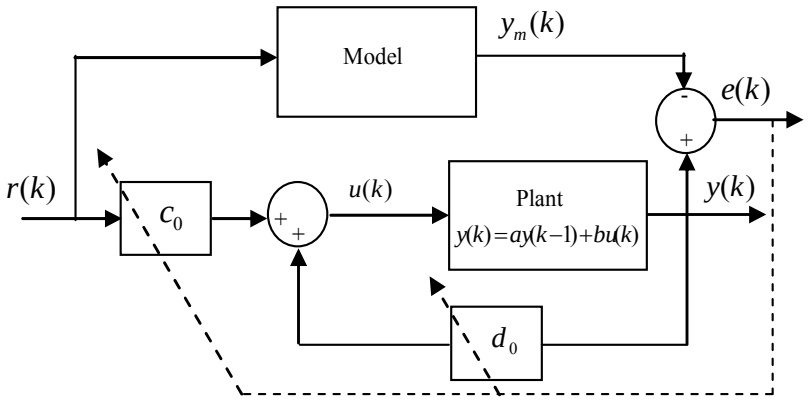


Figure 4.5 Proposed adaptive controller for first order plant

In this case, plant ARMA parameters a and b can be calculated as $a = \frac{1}{1+a_p} = 0.5$ and $b = \frac{b_p}{1+a_p} = 0.5$, respectively, in terms of the plant parameters

by employing backward Euler approximation to the continuous-time plant state equation.

The controller parameters c_0 and d_0 are determined by minimizing the closed loop output (tracking) error in Equation 4.39. Where, $\alpha_0 = bd_0$, $\alpha_1 = a$ and $\beta_0 = bc_0$.

$$\frac{1}{L}\sum_{s=1}^{L}\left[y_m(k) - \alpha_0 y(s) - \alpha_1 y(s-1) - \beta_0 r(s)\right]^2 \tag{4.39}$$

The following gradient algorithm is used for minimizing Equation 4.39 in terms of the controller parameters c_0 and d_0.

$$c_0(k+1) = c_0(k) - \eta\frac{\partial e(k)}{\partial c_0} \quad \text{and} \quad d_0(k+1) = d_0(k) - \eta\frac{\partial e(k)}{\partial d_0} \tag{4.40}$$

Where, the error is defined as $e(k) =: y_m(k) - \alpha_0 y(k) - \alpha_1 y(k-1) - \beta_0 r(k)$ and its partial derivatives with respect to controller parameters c_0 and d_0 are calculated using chain rule as in Equation 4.41.

$$\frac{\partial e}{\partial c_0}(k) = \frac{\partial e(k)}{\partial \beta_0}\frac{\partial \beta_0}{\partial c_0} = -r(k)b \quad \text{and} \quad \frac{\partial e}{\partial d_0}(k) = \frac{\partial e(k)}{\partial \alpha_0}\frac{\partial \alpha_0}{\partial d_0} = -y(k)b \tag{4.41}$$

The resulting time waveforms are depicted in Figure 4.6.

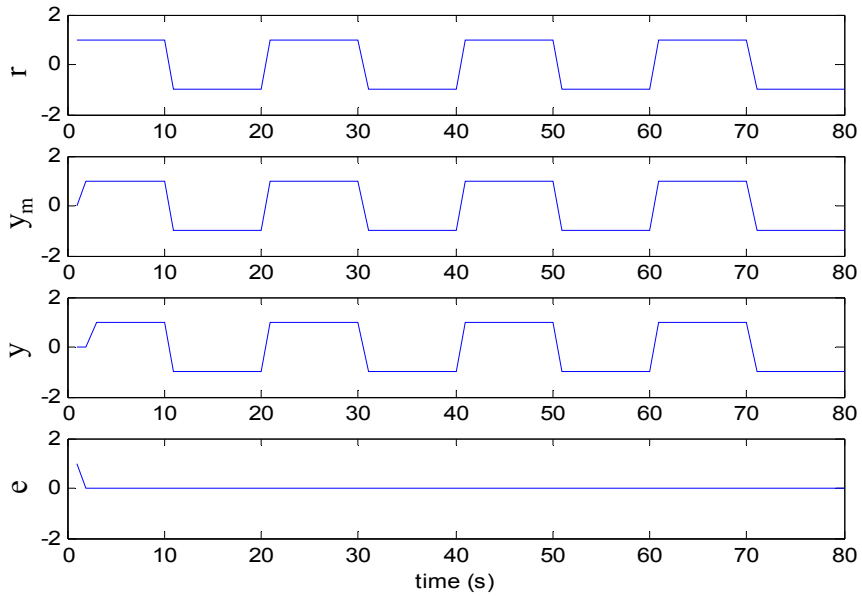


Figure 4.6 Reference input, reference model output, plant output and error time-waveforms obtained by the proposed adaptive control method

4.5.2 Finding PID Parameters

In this subsection, the developed adaptive control method is applied to find (time-invariant) and also (time-varying) adaptive PID controllers. Firstly, consider a PID controller in discrete-time as in Equation 4.42.

$$u(k) = u(k-1) + [K_p + K_i + K_d]e(k) + [-K_p - 2K_d]e(k-1) + K_d e(k-2) \quad (4.42)$$

Where, the error is $e(k) = r(k) - y(k)$, K_p, K_i and K_d stands for the proportional gain, the integral gain and the derivative gain, respectively. Its ARMA model can be written as in Equation 4.43 which is a special case of Equation 4.3.

$$\begin{aligned} u(k) = u(k-1) &+ (c_0 r(k) - d_0 y(k)) \\ &+ (c_1 r(k-1) - d_1 y(k-1)) + (c_2 r(k-2) - d_2 y(k-2)) \end{aligned} \quad (4.43)$$

Where, $c_0 = -d_0 = K_p + K_i + K_d$, $c_1 = -d_1 = -K_p - 2K_d$ and $c_2 = -d_2 = K_d$. So, for a SISO plant having a general ARMA representation in Equation 4.1, the closed loop ARMA representation is obtained as follows.

$$\begin{aligned} y(k) &= a_0 y(k) + (1 - a_0 + a_1) y(k-1) + \sum_{n=2}^{N} (a_n - a_{n-1}) y(k-n) - a_N y(k-N-1) \\ &+ c_0 [b_0 r(k) + b_1 r(k-1) + \sum_{n=2}^{M} b_n r(k-n)] + d_0 [-b_0 y(k) - b_1 y(k-1) - \sum_{n=2}^{M} b_n y(k-n)] \\ &+ c_1 [b_0 r(k-1) + \sum_{n=2}^{M} b_{n-1} r(k-n) - b_M r(k-M-1)] + d_1 [-b_0 y(k-1) - \sum_{n=2}^{M} b_{n-1} y(k-n) - b_M y(k-M-1)] \\ &+ c_2 [\sum_{n=2}^{M} b_{n-2} r(k-n) - b_M r(k-M-2) - b_{M-1} r(k-M-1)] + d_2 [-\sum_{n=2}^{M} b_{n-2} y(k-n) - b_M y(k-M-2)] \end{aligned} \tag{4.44}$$

In this example, plant ARMA parameters a_0, a_1 and b_0 are calculated by using input-output data measured from real plant. And then, the controller parameters c_0, c_1, c_2 and d_0, d_1, d_2 are determined by minimizing the closed loop output (tracking) error in Equations 4.45 in which $\alpha_0 = -1 - a_0 + d_0 b_0$, $\alpha_1 = -1 + a_0 - a_1 + d_1 b_0$, $\alpha_2 = a_1 + d_2 b_0$, $\beta_0 = -c_0 b_0$, $\beta_1 = -c_1 b_0$ and $\beta_2 = -c_2 b_0$.

$$\frac{1}{L} \sum_{s=1}^{L} [y_d(k) - \alpha_0 y(s) - \alpha_1 y(s-1) - \alpha_2 y(s-2) - \beta_0 r(s) - \beta_1 r(s-1) - \beta_2 r(s-2)]^2 \tag{4.45}$$

The following gradient algorithm is used for minimizing Equation 4.46 in terms of the controller parameters c_i and d_i.

$$c_i(k+1) = c_i(k) - \eta \frac{\partial e(k)}{\partial c_i} \text{ and } d_i(k+1) = d_i(k) - \eta \frac{\partial e(k)}{\partial d_i} \text{ for } i = 1,2,3 \tag{4.46}$$

Where, the error is defined as $e(k) = y_d(k) - \alpha_0 y(s) - \alpha_1 y(s-1) - \alpha_2 y(s-2) - \beta_0 r(s) - \beta_1 r(s-1) - \beta_2 r(s-2)$ and its partial derivatives with respect to controller parameters c_0, c_1, c_2 and d_0, d_1, d_2 are calculated using chain rule as in Equation 4.47.

$$\frac{\partial e}{\partial c_0}(k)=\frac{\partial e(k)}{\partial \beta_0}\frac{\partial \beta_0}{\partial c_0}=-r(k)b_0, \qquad \frac{\partial e}{\partial d_0}(k)=\frac{\partial e(k)}{\partial \alpha_0}\frac{\partial \alpha_0}{\partial d_0}=-y(k)b_0$$

$$\frac{\partial e}{\partial c_1}(k)=\frac{\partial e(k)}{\partial \beta_1}\frac{\partial \beta_1}{\partial c_1}=-r(k-1)b_0, \qquad \frac{\partial e}{\partial d_1}(k)=\frac{\partial e(k)}{\partial \alpha_1}\frac{\partial \alpha_1}{\partial d_1}=-y(k-1)b_0 \tag{4.47}$$

$$\frac{\partial e}{\partial c_2}(k)=\frac{\partial e(k)}{\partial \beta_2}\frac{\partial \beta_2}{\partial c_2}=-r(k-2)b_0, \qquad \frac{\partial e}{\partial d_2}(k)=\frac{\partial e(k)}{\partial \alpha_2}\frac{\partial \alpha_2}{\partial d_2}=-y(k-2)b_0$$

The real DC motor is controlled by proposed controller in S-R-R real-time operation mode. Desired output is chosen at 2500 rpm. The resulting time waveform is depicted in Figure 4.7.

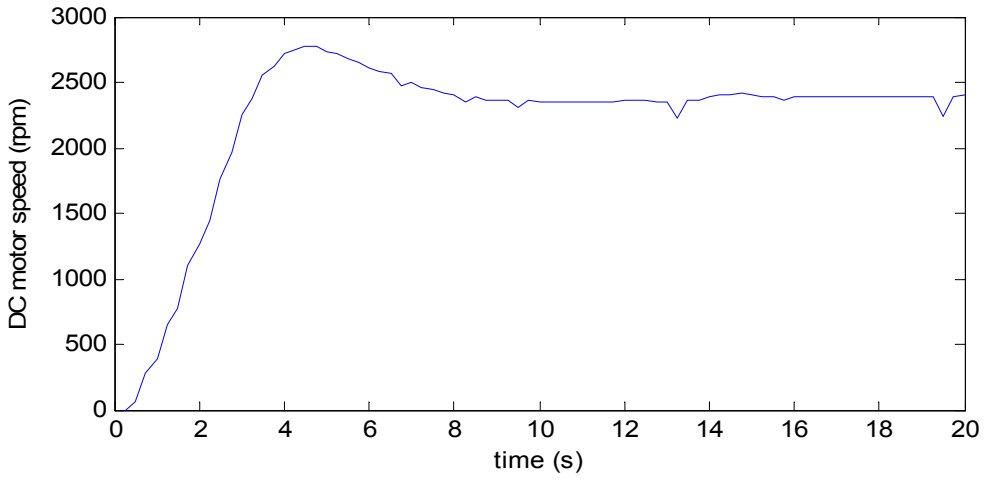


Figure 4.7 The real DC motor output time-waveform obtained by proposed adaptive control method

It can be seen from Figure 4.7 that one may prefer the proposed adaptive controller method since the step responses of the real DC motor has very small overshot percentage and it is close to the reference signal.

In addition to the above result, the proposed controller parameters can also be used for finding conventional PID controller parameters because, in Equation 4.43, PID controller' ARMA model is related to the Equation 4.42 in terms of controller

parameters such as $c_0 = K_p + K_i + K_d$, $c_1 = -K_p - 2K_d$ and $c_2 = -K_d$. These controller parameters c_0, c_1, and c_2 are observed as 0.00002441, 0.0006994 and -0.0003742, respectively, from CDTRP platform during steady state behavior of closed loop system. Then, conventional PID controller parameters are determined as $K_p = 0.000049$, $K_i = 0.0003496$ and $K_d = -0.0003742$. According to these parameters, the real DC motor is controlled by conventional PID controller in S-R-R real-time operation mode. Its desired output is chosen at 2500 rpm. The resulting time waveform is given in Figure 4.8.

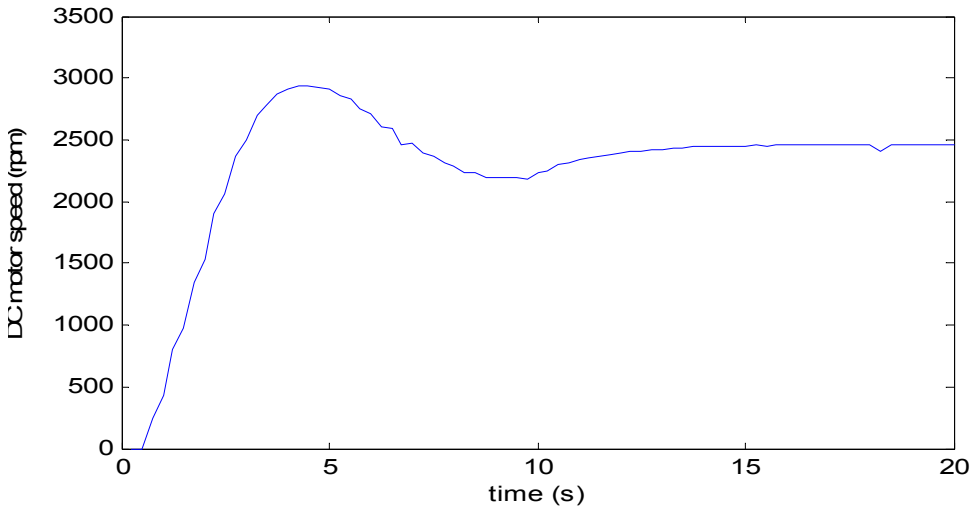


Figure 4.8 The real DC motor output time-waveform obtained by PID controller designed by proposed method

On the other hand, the real DC motor controlled by PID controller parameters designed by ZN method is depicted in terms of its performances for tracking value at 2500 rpm in Figure 4.9 (Note that, in Subsection 3.6.2, the parameters of the PID controller designed by ZN method are determined as $K_p = 54.54$, $K_i = 2479.3$ and $K_d = 0.3$).

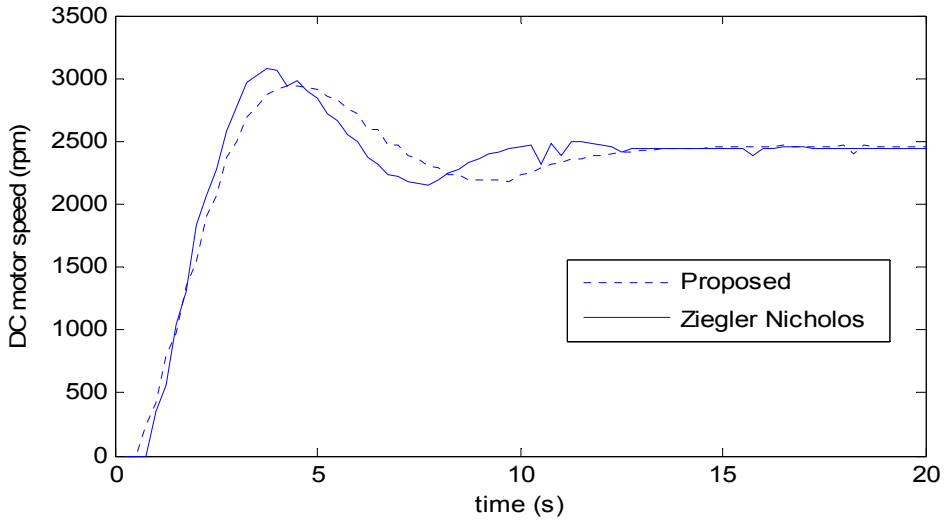


Figure 4.9 Time-waveforms of the real DC motor controlled by proposed parameter technique and conventional ZN technique for PID controller.

Both proposed and ZN methods for finding conventional PID controller parameters are compared with conventional PID controller each other in terms of their performances for tracking value at 2500 rpm in Figure 4.9. These observed tracking values are so close to each other. It can be said that the real plant might behave in same way for the proposed and ZN based PID controllers. So, the proposed adaptive controller parameters might be used for calculating PID controller parameters as another technique.

4.5.3 RBFN Based Proposed Adaptive Controller for Nonlinear Plant

In this subsection, the proposed adaptive control method is implemented with RBFN based NARMA-I model. The RBFN consists of three different layers: i) input layer, ii) hidden layer where neurons are constructed as radial basis functions and iii) output layer where outputs of the hidden neurons are summed in terms of their linear weights. RBFN structure with one output neuron is depicted in Figure 4.10.

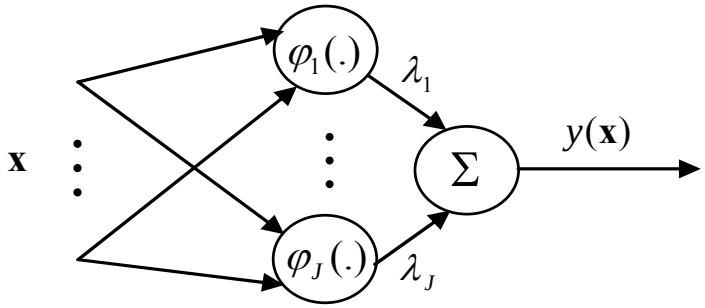


Figure 4.10 RBFN structure with one output.

$$y(\mathbf{x}) = \sum_{j=1}^{J} \lambda_j \varphi_j(\mathbf{x}) = \sum_{j=1}^{J} \lambda_j e^{-\frac{\|x-\theta_j\|_2^2}{\sigma_j^2}} \tag{4.48}$$

Where $y(\circ): R^n \to R$, $\mathbf{x} \in \Re^n$ stands for input vector, J is the number of hidden neurons, λ_j stands for linear weights, $\varphi_j(\circ): R^n \to R$ stands for the radial basis functions, herein the Gaussian functions $\varphi_j(\mathbf{x}) = e^{-\frac{\|x-\theta_j\|_2^2}{\sigma_j^2}}$, θ_j stands for the centers of the Gaussians, and σ_j^2 stands for the variances of the Gaussians.

Now, as a nonlinear plant example, consider one link robot defined in Table 3.4 as BP3. And then, consider the nonlinear version of the proposed adaptive control method by the following control law with two parameters in Equation 4.26 where controller parameters are chosen c_0, c_1, d_0 and d_1.

$$u(k) = -\sum_{m=0}^{1} c_m \varphi_m(r[k,N]) - \sum_{m=0}^{1} d_m \varphi_m(y[k,N]) \tag{4.49}$$

According to Subsection 4.2, in the first stage one can minimize the following identification error Equation 4.50 in terms of the plant NARMA-I parameters a_j and b_j for a given set of plant input-output measurements. Its identification scheme is depicted in Figure 4.11.

$$\frac{1}{K}\sum_{s=1}^{K}\left[y(s)-\sum_{j=1}^{J} a_j \varphi_j(y[k-1,n])+\sum_{j=1}^{J} b_j \varphi_j(y[k-1,n])u(k-n)\right]^2 \qquad (4.50)$$

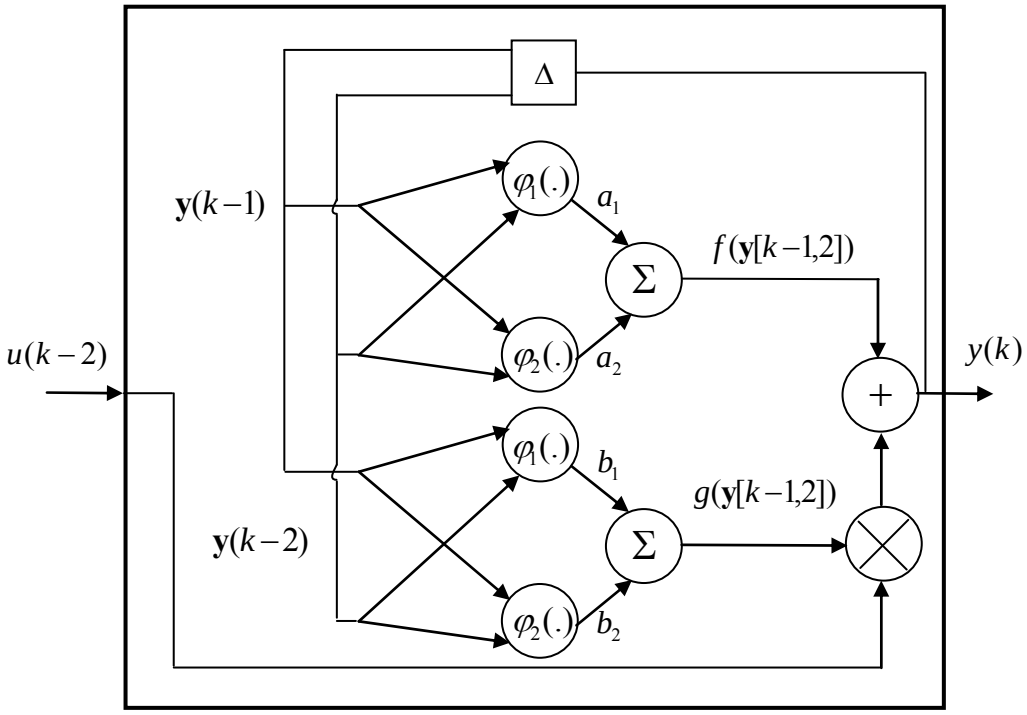
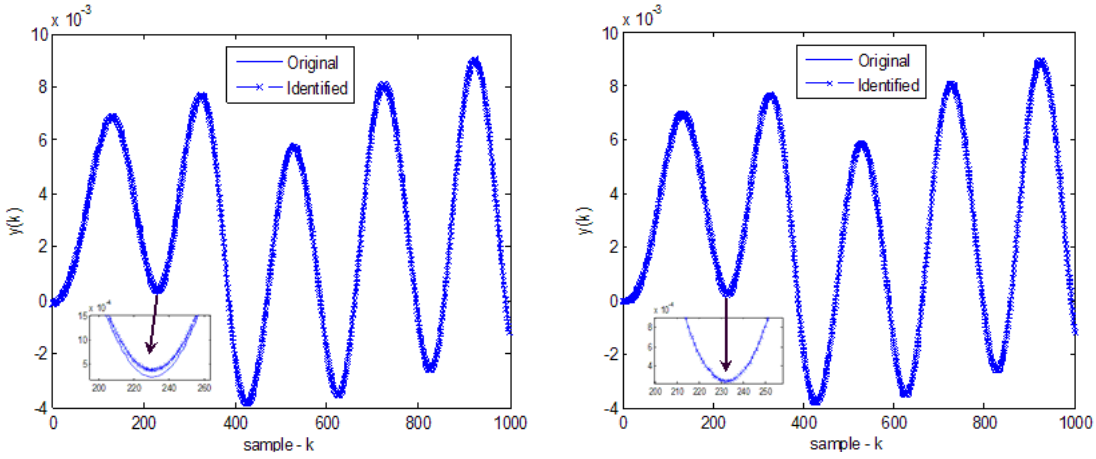


Figure 4.11 RBFN based identification scheme for NARMA-I.

The centers of the RBF are computed by K-means algorithm of MATLAB function as 0.0058 and -0.0004. The simulated batch and online identification modes of the one link robot are given in Figure 4.12 (a) and (b) respectively. Mean Square Error (MSE) values are observed as $4.8845*10^{-9}$ and $2.5445*10^{-16}$ for batch mode and online methods, respectively.



(a) (b)

Figure 4.12 The observed responses, i.e., plot of the batch (a) and online (b) modes for identification

In the second stage, closed loop output (tracking) error in Equation 4.51 can be minimized by terms of the controller NARMA parameters c_0, c_1, d_0 and d_1 for a given set of (reference input)-(desired output) sample pairs $\{(r[s,N], y[s,N]\}_{s=1}^{L}$.

$$y(k) = \sum_{j=1}^{J} a_j \varphi_j(y[k-1,n]) - \sum_{m=0}^{1}\sum_{j=1}^{J} d_m b_j \varphi_j(y[k-1,n])\varphi_m(y[k-1,n]) - \sum_{m=0}^{1}\sum_{j=1}^{J} c_m b_j \varphi_j(y[k-1,n])\varphi_m(r[k-1,n]) \quad (4.51)$$

The overall RBFN based proposed adaptive controller system has the block diagram in Figure 4.13 where identified plant NARMA model and controller NARMA parameters are depicted in detail.

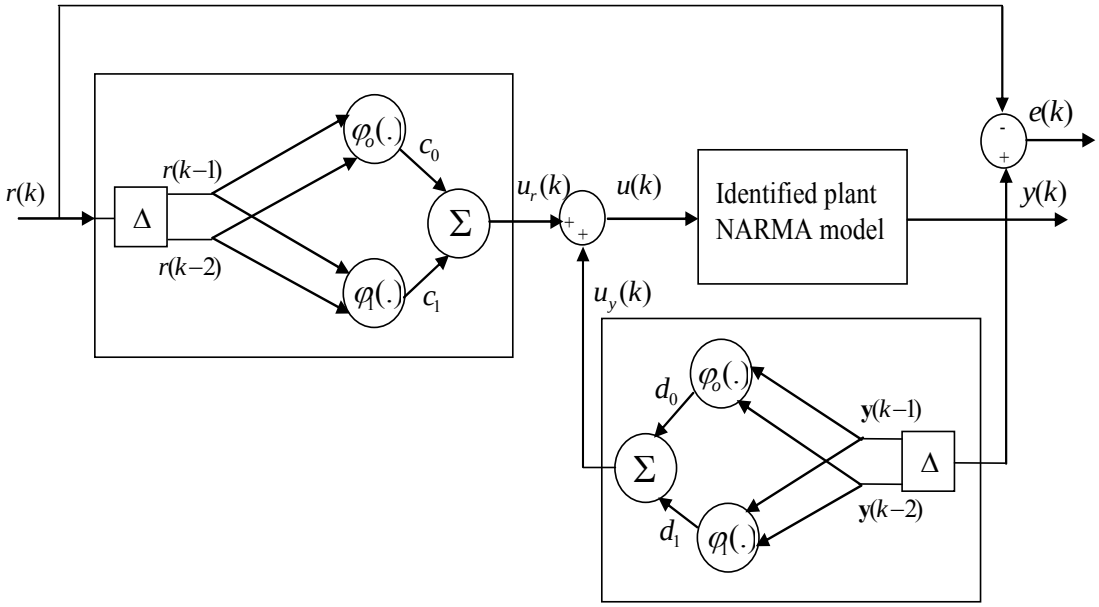
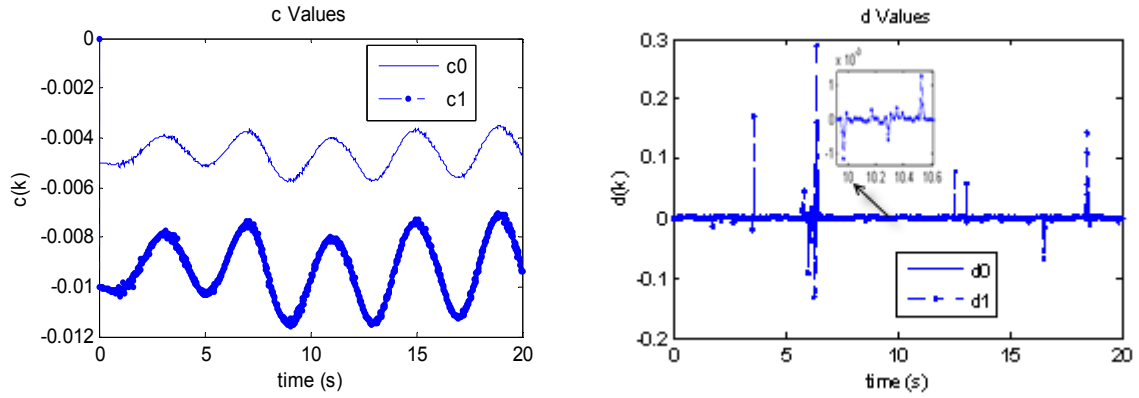


Figure 4.13 RBFN based proposed adaptive Controller for Nonlinear Plant

RBFN based proposed adaptive controller system is simulated by S-S-S and S-E-R real time operating modes in CDTRP platform. The simulated time waveforms of the controller NARMA parameters are given in Figure 4.14(a)-(b).



(a) (b)

Figure 4.14 Observed variations of the controller NARMA parameters c_0, c_1 and d_0, d_1 in (a) and (b) respectively.

In this case, the desired output angular position of the system is chosen as $r(t) = 0.5 * \sin 2\pi 10t$. The simulated time waveforms of the nonlinear plant states controlled by RBFN based proposed adaptive controller are given in Figure 4.15

where x_1 and x_2 angular position of the one link robot and x_1 of them is chosen as the actual output.

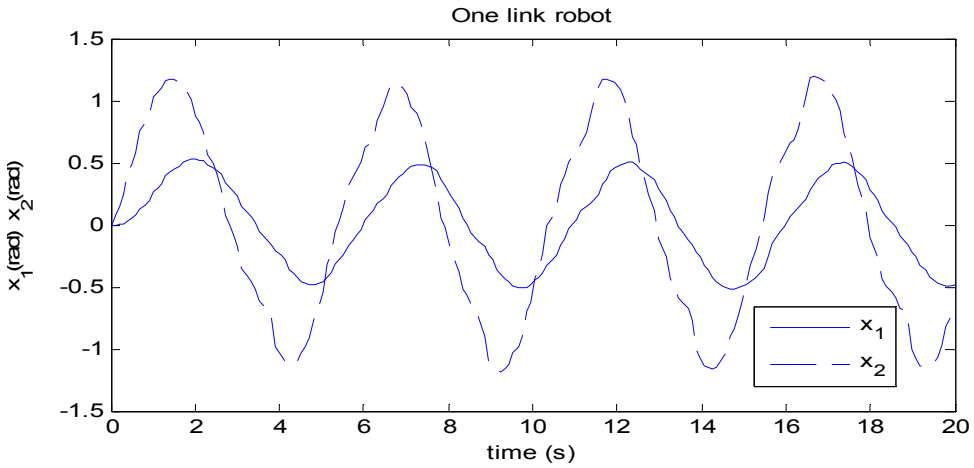


Figure 4.15 Observed waveforms of the one link robot states controlled by proposed NARMA controller

The observed actual and desired output waveforms of the one link robot controlled by RBFN based proposed adaptive controller are depicted in terms of its performances for tracking angular position values in Figure 4.16. Their MSE is obtained as 0.1340. So, proposed adaptive NARMA controller might be used for controlling nonlinear plants as another adaptive control technique.

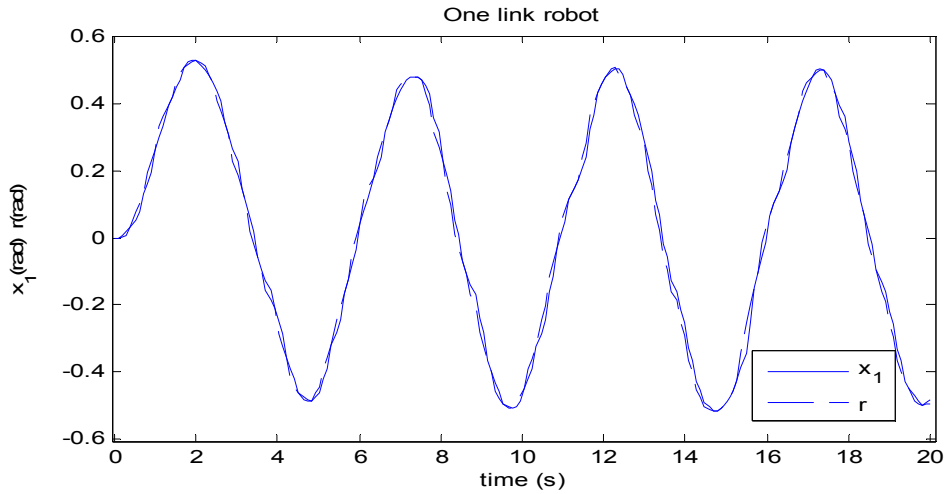


Figure 4.16 Observed waveforms of the one link robot states controlled by proposed NARMA controller

CHAPTER FIVE

MODEL BASED DYNAMICAL STATE FEEDBACK CHAOTIFICATION

This chapter introduces chaotic reference model based dynamical state feedback chaotification method which can be applied to any input-state linearizable (nonlinear) system including linear controllable ones as special cases. In the developed method, any chaotic system of arbitrary dimension can be used as the reference model with no need to transform it into a special form, so providing the advantage of exploiting the vast amount of information on chaotic systems and their implementations available in the literature. To demonstrate the potential effective applications of the method, a permanent magnet DC motor is chaotified by the proposed dynamical state feedback as matching the closed loop dynamics to the well known Chua's chaotic circuit. Then, an impeller mounted on the chaotified DC motor is used for mixing a corn syrup added acid-base mixture. It is observed in a non-intrusive way that mixing actuated by the chaotified DC motor is more efficient than constant and also periodical motor speed cases for the consideration of neutralization time and power consumption together.

5.1 Dynamical State Feedback Chaotification

It will be shown firstly that any linear, time invariant, controllable and observable single-input system can be chaotified by dynamical state feedback. This result will then be extended into the nonlinear, time-invariant, input-state linearizable and observable single-input systems.

5.1.1 Linear Systems Case

Consider an nth order linear, time-invariant and controllable single-input system defined by the state equations $\dot{\hat{\mathbf{x}}} = \hat{\mathbf{A}}\mathbf{x} + \hat{\mathbf{b}}u$. It is known that such a system is controllable if and only if the controllability matrix $[\hat{\mathbf{b}} \quad \hat{\mathbf{A}}\hat{\mathbf{b}} \quad \dots \quad \hat{\mathbf{A}}^{n-1}\hat{\mathbf{b}}]$ has rank *n*

(Rugh, 1996). It is also known that the state equations of a controllable system can be transformed into the following controllable canonical form $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{b}u$ by a linear change $\hat{\mathbf{x}} =: \mathbf{T}\mathbf{x}$ of variables.

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \vdots \\ \dot{x}_{n-1} \\ \dot{x}_n \end{bmatrix} = \begin{bmatrix} 0 & 1 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & 0 & 0 & 1 \\ -a_1 & -a_2 & \cdots & \cdots & -a_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} u \tag{5.1}$$

Where u stands for the scalar input, the state matrices and input vectors corresponding to the old $\hat{\mathbf{x}} =: [\hat{x}_1, \hat{x}_2, ..., \hat{x}_n]^T$ and new $\mathbf{x} =: [x_1, x_2, ..., x_n]^T$ state vectors are related to each other by the transformation matrix $\mathbf{T}$ as $\mathbf{A} = \mathbf{T}^{-1}\hat{\mathbf{A}}\mathbf{T}$ and $\mathbf{b} = \mathbf{T}^{-1}\hat{\mathbf{b}}$.

Now, consider a reference chaotic system, for instance in Equation 5.2 and Equation 5.3. Although there are many alternatives to choose (Lorenz, Rössler, Chen and so on), Chua's circuit defined by Equation 5.2 and Equation 5.3 (Chua et al., 1993) is taken as the reference chaotic system in this paper in order to exploit the huge amount of existing knowledge on efficient hardware realizations and qualitative/quantitative analyses for this special chaotic system.

$$\begin{aligned} \dot{x} &= -\hat{\alpha}x + \hat{\alpha}(y - F(x)) \\ \dot{y} &= x - y + z \\ \dot{z} &= -\hat{\beta}y \end{aligned} \tag{5.2}$$

$$F(x) = m_0 x + \frac{1}{2}(m_1 - m_0)\left[|x+1| - |x-1|\right] \tag{5.3}$$

Where $F(x)$ is a piecewise linear function whose segment slopes are $m_0 = -0.68$ and $m_1 = -1.27$. $\hat{\alpha} > 0$ and $\hat{\beta} > 0$ are scalar parameters whose values should be chosen appropriately to ensure the chaotic behavior of Chua's circuit. Redefining the

states of Chua's circuit as $x_n = x$, $x_{n+1} = y$, $x_{n+2} = z$ and choosing a dynamical state feedback control as $u = a_1 x_1 + \cdots + a_{n-1} x_{n-1} - (\hat{\alpha} - a_n) x_n + \hat{\alpha}[x_{n+1} - F(x_n)]$, the resulting nonlinear closed loop dynamics of the originally linear system in Equation 5.1 is obtained as in Equation 5.4 such that its last three equations match exactly to the Chua's circuit equations with the newly defined state variables (See Equation 5.5).

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \vdots \\ \dot{x}_{n-1} \\ \dot{x}_n \\ \dot{x}_{n+1} \\ \dot{x}_{n+2} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 & 0 \\ 0 & 0 & 0 & \cdots & -\hat{\alpha} & \hat{\alpha} & 0 \\ 0 & 0 & 0 & \cdots & 1 & -1 & 1 \\ 0 & 0 & 0 & \cdots & 0 & -\hat{\beta} & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \\ x_{n+1} \\ x_{n+2} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ -\hat{\alpha} F(x_n) \\ 0 \\ 0 \end{bmatrix} \tag{5.4}$$

$$\begin{aligned} \dot{x}_n &= -\hat{\alpha} x_n - \left[a_1 x_1 + \cdots + a_{n-1} x_{n-1} - (\hat{\alpha} - a_n) x_n - u \right] \\ &= -\hat{\alpha} x_n + \hat{\alpha}\left[x_{n+1} - F(x_n) \right] \\ \dot{x}_{n+1} &= x_n - x_{n+1} + x_{n+2} \\ \dot{x}_{n+2} &= -\hat{\beta} x_{n+1} \end{aligned} \tag{5.5}$$

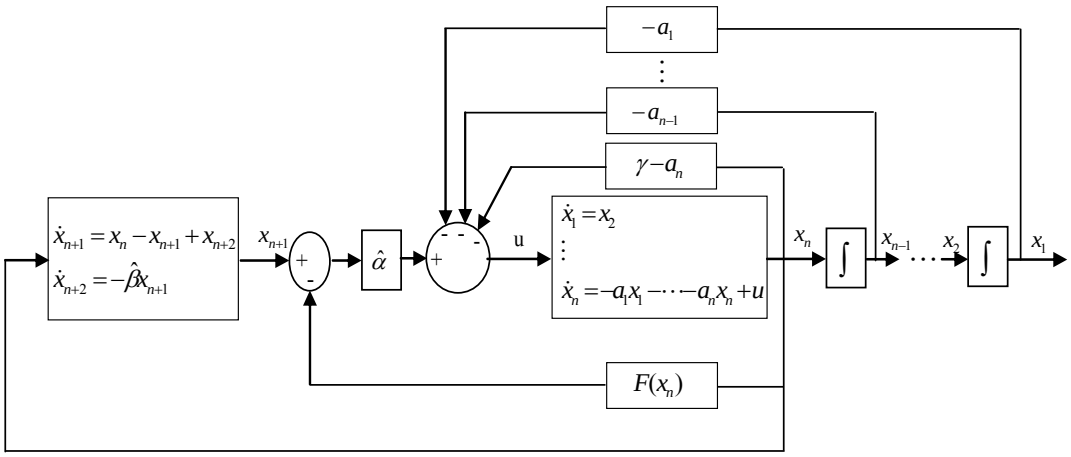


Figure 5.1 Block diagram of the proposed chaotification method based on Chua system for linear, time-invariant systems defined in the controllable canonical form

The above described chaotification which is also depicted in Figure 5.1 has linear, nonlinear and dynamical state feedback, so it is called as (nonlinear) dynamical state feedback chaotification.

The resulting chaotic system in Equation 5.1 is, indeed, equivalent to the reference chaotic system cascaded by (*n-1*) integrators as depicted in Figure 5.2. Although the nonlinear state feedback is always necessary for chaotification of linear systems, it may not be needed for a nonlinear system already having the nonlinearity same with the one required to produce the reference chaotic system.

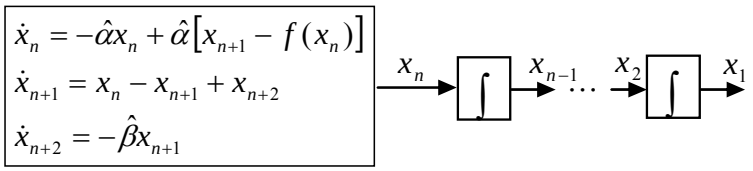


Figure 5.2 Reference chaotic systems cascaded by (n-1) integrators as the equivalent of the chaotified system

5.2 Comparison of Dynamical and Static Feedback Chaotification Methods

The proposed dynamical state feedback chaotification method is applicable under the conditions same with the ones for the method in (Morgül, 2003) and it needs the system states to be measured or to be determined indirectly, e.g. by an observer as in the method of (Morgül, 2003). The main difference between the dynamical state feedback chaotification and the static feedback chaotification in (Morgül, 2003) is follows. The reference based static feedback chaotification of (Morgül, 2003) requires transforming the reference chaotic system into the Brunovsky form given in Equation 5.6 before matching the system to the reference chaotic system while the dynamical state feedback chaotification keeps the original form of the reference chaotic system in matching, so providing the superiority of exploiting the efficient hardware realizations and also extensive analysis results for the reference chaotic systems already available in the literature.

$$\begin{aligned}
\dot{w}_1 &= w_2 \\
\dot{w}_2 &= w_3 \\
&\vdots \\
\dot{w}_{n-1} &= w_n \\
\dot{w}_n &= f(w_1, w_2, ..., w_n)
\end{aligned} \tag{5.6}$$

If the original system to be chaotified is not already in the controllable canonical form (Equation 5.1), the dynamical state feedback chaotifying control input described in takes the form $u=[a_1\cdots a_n]\mathbf{T}^{-1}\hat{\mathbf{x}}-\hat{\alpha}(\mathbf{T}^{-1}\hat{\mathbf{x}})_n+\hat{\alpha}[y-F((\mathbf{T}^{-1}\hat{\mathbf{x}})_n)]$ in terms of the original state variables $\hat{\mathbf{x}}$, meaning that the already available realizations of a reference chaotic system can be used at most by a linear change of variables for the developed dynamical state feedback chaotification method in contrast to the static state feedback chaotification method requiring, in general, a nonlinear transformation of the reference chaotic system if it is not in the form of Equation 5.6. One can see this fact by considering the well known Rössler chaotic system in Equation 5.7 (Rössler, 1979) and by transforming it into the Brunovsky form in Equation 5.9 via the nonlinear change Equation 5.8 of variables (Morgül & Solak, 1996).

$$\begin{aligned}\dot{x} &= ax+y\\ \dot{y} &= x-z\\ \dot{z} &= -cz+yz+b\end{aligned} \tag{5.7}$$

$$\begin{aligned}w_1 &=: x\\ w_2 &=: ax+y\\ w_3 &=: (a^2-1)x+ay-z\end{aligned} \tag{5.8}$$

$$\begin{aligned}\dot{w}_1 &= w_2\\ \dot{w}_2 &= w_3\\ \dot{w}_3 &= -cw_1+(ca-1)w_2+(a-c)w_3-aw_1^2-aw_2^2\\ &\quad +(a^2-1)w_1w_2-aw_1w_3+w_2w_3-b\end{aligned} \tag{5.9}$$

The transformed Rössler system in Equation 5.9 has a less efficient structure in terms of the hardware implementation as compared to the original form in Equation 5.7 since it requires five nonlinearities to be realized although the original system in Equation 5.7 only one. The huge amount of literature on the theoretical and experimental analyses of the original Rössler system in Equation 5.7 constitutes

another important reason to prefer to use the original Rössler system in Equation 5.7 as the reference chaotic system. The above discussion explains the situation about the implementation of the nonlinearities. The following gives the account of implementation of dynamical part of the resulting chaotified system.

On the other hand, the dynamical state feedback chaotification augments the degree of the original system by two as adding two more state equations corresponding to the unmatched part of the reference chaotic system when the reference chaotic system is of three dimensional. This is not the case for the static feedback chaotification retaining the degree of the original system the same. This means that the dynamical state feedback chaotification requires implementing the unmatched part of the reference chaotic system which consists of two integration units and some linear/nonlinear units. It can be concluded that the dynamical state chaotification constitutes a good alternative to the static state chaotification when the reference chaotic system does not have any efficiently implementable Brunovsky form in Equation 5.9.

5.3 Nonlinear Systems Case

Consider an nth order nonlinear, time-invariant and input-state linearizable single-input system defined by the bilinear state equations $\dot{\hat{\mathbf{x}}} = \mathbf{f}(\hat{\mathbf{x}}) + \mathbf{g}(\hat{\mathbf{x}})u$ where $\mathbf{f}(\circ)$ and $\mathbf{g}(\circ)$ are smooth vector fields. It is known that such a system is input-state linearizable if and only if i) The set of vector fields $\{\mathbf{g} \quad \mathbf{ad_f g} \quad \ldots \quad \mathbf{ad_f}^{n-1}\mathbf{g}\}$ is linearly independent and ii) The set of vector fields $\{\mathbf{g} \quad \mathbf{ad_f g} \quad \ldots \quad \mathbf{ad_f}^{n-2}\mathbf{g}\}$ is involutive (Slotine & Li, 1991). Herein, $\mathbf{ad_f g}$ is the Lie bracket $[\mathbf{f},\mathbf{g}] = (\nabla\mathbf{g})\mathbf{f} - (\nabla\mathbf{f})\mathbf{g}$ with ∇ denoting the gradient operator with respect to $\mathbf{x}$ and being involutive for a set of vector fields means that the Lie bracket of any pair of vector fields in the set is in the range of the same set of vector fields, i.e. any such Lie bracket can be written as a linear combination of the vector fields in this set (Slotine & Li, 1991). It is also known that the state equations of such an input-state

linearizable system can be transformed into the Brunovsky canonical form $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{b}v$ in Equation 5.10 by introducing a nonlinear state feedback as $u = \alpha(\hat{\mathbf{x}}) + \beta(\hat{\mathbf{x}})v$ with $v = \beta^{-1}(\hat{\mathbf{x}})(u - \alpha(\hat{\mathbf{x}}))$ defining the new input and by a nonlinear change $\hat{\mathbf{x}} =: \mathbf{\Phi}(\mathbf{x})$ of variables where $\mathbf{\Phi}(\circ) : \mathbf{R}^{\mathbf{n}} \to \mathbf{R}^{\mathbf{n}}$ is a C^1 diffeomorphism such that the continuously differentiable inverse transform $\mathbf{x} = \mathbf{\Phi}^{-1}(\hat{\mathbf{x}})$ is also defined (Slotine & Li, 1991).

$$\dot{\mathbf{x}} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} v = \mathbf{A}\mathbf{x} + \mathbf{b}v \tag{5.10}$$

Now, if Chua's circuit defined in Equation 5.2 and Equation 5.3 is considered as a reference chaotic system and the first equation of Equation 5.2 is chosen to be matched, then the dynamical state feedback chaotifying control v in terms of the original system variables $\hat{\mathbf{x}}$ would be $v = -\hat{\alpha}(\Phi^{-1}(\hat{\mathbf{x}}))_n + \hat{\alpha}[y - F((\Phi^{-1}(\hat{\mathbf{x}}))_n)]$, so the original system input $u = \alpha(\hat{\mathbf{x}}) + \beta(\hat{\mathbf{x}})[-\hat{\alpha}(\Phi^{-1}(\hat{\mathbf{x}}))_n + \hat{\alpha}[y - F((\Phi^{-1}(\hat{\mathbf{x}}))_n)]]$.

5.4 Dynamical State Feedback Chaotification of DC Motor

The developed dynamical state feedback chaotification method is employed in this section to chaotify a permanent magnet DC motor in order to be used as the actuator for a liquid mixing system. The proposed chaotifying control was implemented in the Controller-Design-Test-Redesign-Platform (CDTRP) which was developed in (Şahin et al., 2010). This section presents results obtained in two different operating modes of CDTRP. The first one is the real-time simulated plant (herein the DC motor.) and real-time simulated controller (herein the dynamical state feedback chaotifying controller.) mode of the CDTRP, more precisely the mode where both of the plant and the controller are implemented in the real-time simulator realized in a Personal Computer (PC) having a Centrino processor and a 1GB

memory. (Herein, the computer constitutes a minimum configuration necessary for running MS Windows XP and MATLAB 7.04 software on which a graphical user interface managing and monitoring the CDTRP is implemented.) The second one is the real DC motor and real-time simulated controller mode of the CDTRP where a real DC motor together with a derive card is controlled by the dynamical state feedback implemented in the real-time simulator.

The configuration of real (DC motor) plant and simulated (chaotifying) controller mode of CDTRP is given in Figure 5.3. The chaotifying control signal u is produced in the simulator (PC) and it is then converted into Pulse Width Modulation (PWM) signal in order to apply to the DC motor as the adjustable armature voltage for controlling DC motor speed i.e. the revolutions per minute (rpm). The DC motor driver card is realized with a PIC18F452 microcontroller driver unit which has a serial interface to communicate with the PC.

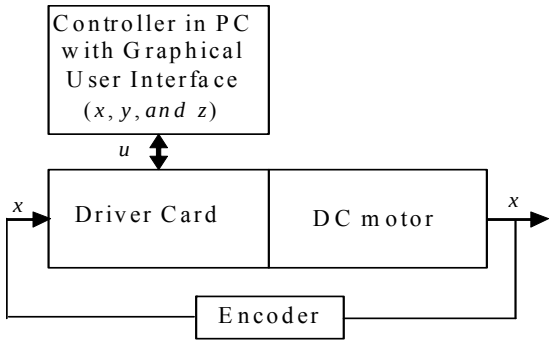


Figure 5.3 Block diagram for real DC motor plant and simulated chaotifying controller mode of CDTRP

For the used permanent magnet unidirectional DC motor, a first order state equation $\dot{x} = -a_m x + b_m u$ was taken as the model where x denotes the DC motor speed (rpm) measured by an encoder. The state x was scaled via signal conditioning for the unidirectional DC motor. The model parameters were identified from the responses due to the step input within the experimental setup described in Section 3 as: $a_m = 2$ and $b_m = 8448$ for unloaded case and as $a_m = 1$ and $b_m = 250$ for the loaded case, i.e. when mixing corn syrup added acid-base mixture.

To chaotify the DC motor by dynamical state feedback chaotification method, Chua's circuit was chosen as the reference chaotic system and its first equation was matched to the DC motor equation as follows.

$$\begin{aligned}\dot{x} &= -a_m x - (\hat{\alpha} - a_m)x + \hat{\alpha}(y - F(x)) = -a_m x + b_m u \\ \dot{y} &= x - y + z \\ \dot{z} &= -\hat{\beta} y\end{aligned} \tag{5.11}$$

The chaotifying input admits the following form.

$$u = \frac{1}{b_m}\left[-(\hat{\alpha} - a_m)x + \hat{\alpha}(y - F(x))\right] \tag{5.12}$$

Where, $\hat{\alpha} = 9$, $\hat{\beta} = 14.87$, $m_0 = -5/7$ and $m_1 = -8/7$ were chosen to yield the chaotic behavior of Chua's circuit (Chua et al., 1993). Figure 5.4 depicts the introduced dynamical state feedback chaotification of DC motor.

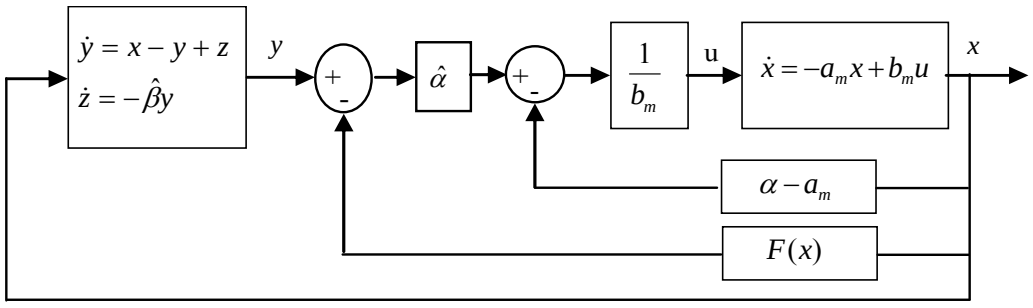


Figure 5.4 Block diagram representation of the dynamical state feedback chaotification of DC motor based on Chua's circuit

The closed loop dynamics of the Chua's circuit based chaotified DC motor defined by Equation 5.11 and Equation 5.12 was firstly simulated in the simulated plant and simulated controller mode of CDTRP and then realized thus monitored/measured in the real plant and simulated controller mode of CDTRP. Figure 5.5 and 5.6 show the obtained chaotic trajectory plotted in the $x - y - z$ space

and the bifurcation diagram of the first state variable x for the $\hat{\alpha}$ parameter, respectively, in the simulated DC motor plant and simulated chaotifying controller mode.

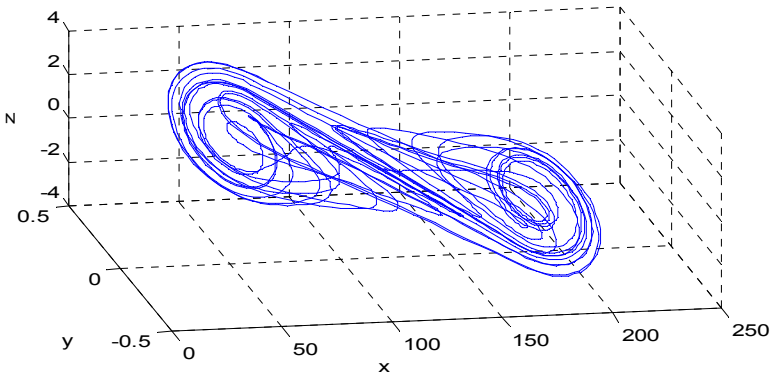


Figure 5.5 Phase portrait of Chua's circuit based chaotified DC motor

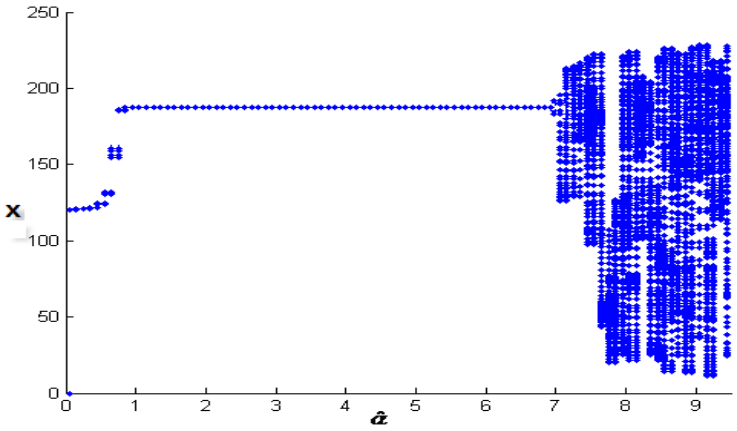


Figure 5.6 Bifurcation diagram obtained for $\hat{\alpha}$

In the real (DC motor) plant and simulated (chaotifying) controller mode of operation of CDTRP, the real DC motor speed (rpm) measured by the encoder is transferred via data lines, i.e. the serial port, to the PC and monitored by the graphical user interface. The rpm measurement data for the real DC motor chaotified by the dynamical state feedback method for $\hat{\alpha}=9$, $\hat{\beta}=14.87$, $m_0=-5/7$ and $m_1=-8/7$ Chua's circuit parameters and the first order DC motor model with the parameters $a_m=1$ and $b_m=250$ is plotted in Figure 5.7. In the experiments realized on real DC motor, the DC motor was operated in three different modes: i) the

constant speed, ii) periodically changed speed and iii) chaotically changed speed. As observed from the experiments, the dynamical state feedback chaotification provides a broad DC motor operating frequency band while the highest operating frequency attained for open loop periodical excitation becomes less than 1 Hertz. This fact is seen from rpm frequency spectra in Figure 5.8.

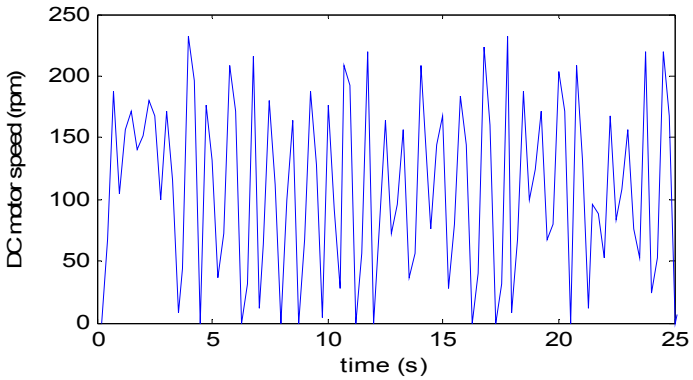


Figure 5.7 Real DC motor speed time waveform measured when dynamical state feedback chaotification was applied

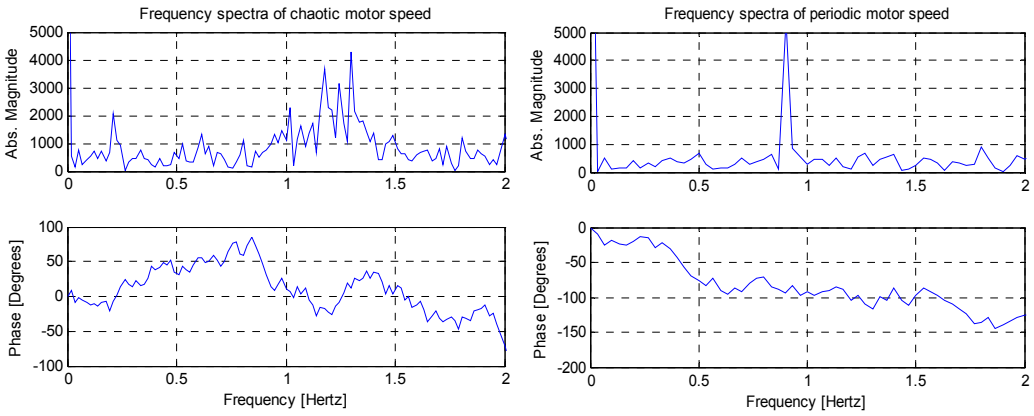


Figure 5.8 (a) Motor speed frequency spectra for chaotified DC motor (b) Motor speed frequency spectra for periodically excited DC motor

5.5 Experimental Results on Liquid Mixing by Chaotified DC Motor

The section presents experimental results on liquid mixing by the DC motor chaotified based on the developed chaotic reference system based dynamical state feedback. Figure 5.9 shows the experimental set-up of the chaotic mixing platform

which is established in Department of Control and Automation at Ege University Ege Vocational School. The details on the application, analysis and also implementation of the proposed chaotification of the considered permanent magnet DC motor are already given in Section 5.3. The chaotic mixing platform which consists of the chaotified DC motor, an impeller mounted on the shaft of the DC motor, the DC motor driver card, the PC for managing and monitoring the mixing platform, chemical substances and an ammeter for measuring the power consumption is indeed implemented on the general purpose controller-design-test-redesign-platform CDTRP (Şahin et al., 2010). The data obtained by the ammeter Brymen BM815 used for measuring the current of the DC motor is transferred via a serial interface connected to the PC which is different from the one used for transferring the data from the encoder to the PC. The impeller mounted to the DC motor is used for mixing the liquid, i.e. the chemical substances, in the cup.

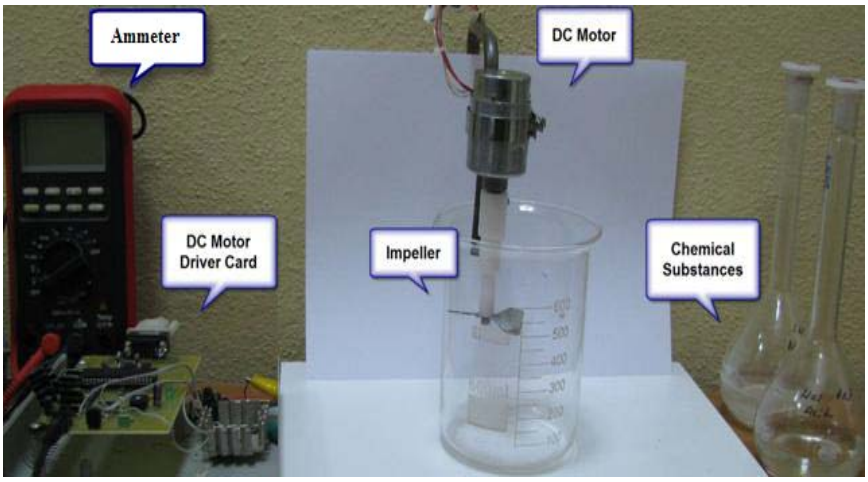


Figure 5.9 Experimental set-up for liquid mixing by chaotified DC motor

The chemical substances used in the experiments are Cargill GF 30 corn syrup (See the photograph in Figure 5.10(a)), 1N HCl acid, 1N NaOH base and Timol blue indicator. The chemical substances for the liquid mixing experiment were prepared in the following steps. 240ml light corn syrup was obtained in the cup in Figure 5.10 (a) by mixing 204ml corn syrup and 36 ml pure water, and then it was divided into two different cups shown in Figure 5.10 (b). 2ml 1N HCl acid and 0.4ml Timol blue indicator were added into the cup at the left in Figure 5.10 (b) which had 160ml portion of the divided light corn syrup and the resulting mixture was mixed well by

shaking manually until the color of the mixture turned to the magenta color. The 80ml portion of the light corn syrup in the cup at the right in Figure 5.10 (b) was mixed by 4ml 1N NaOH base and 0.8ml Timol blue indicator until the resulting mixture's color turned to the cyan color. The basic solution obtained in the cup at the right in Figure 5.10 (b) was then added to the cup containing acidic solution. As seen in Figure 5.10 (c), the resulting mixture is not homogeneous due to the prevention of diffusion by the light corn syrup at the beginning of the mixing process.

(a) (b) (c)

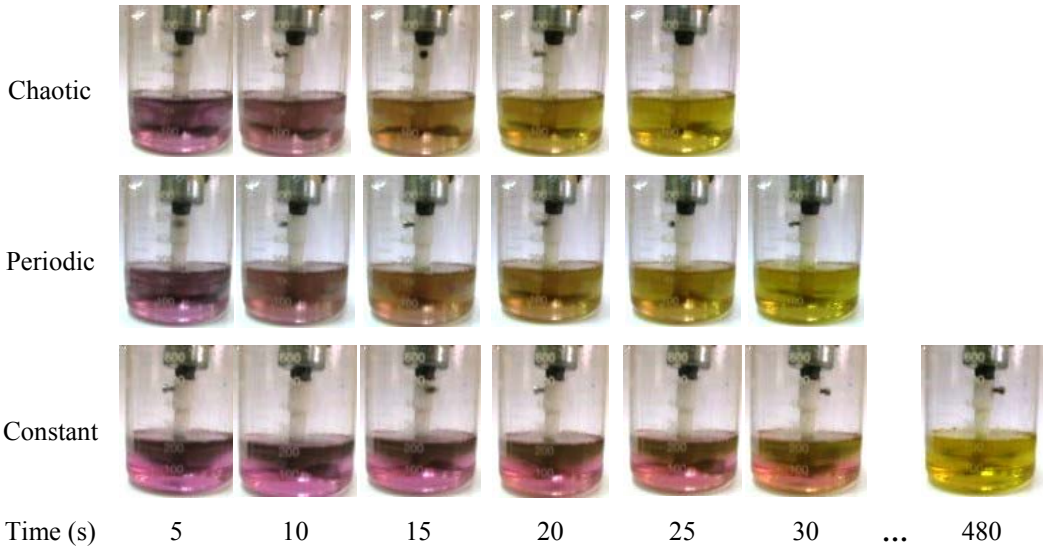
Figure 5.10 Preparing acid-base chemical substances for the chaotic liquid mixing experiment

Three different mixing modes, i.e. constant motor speed, (sinusoidal) periodic changing motor speed and chaotic changing motor speed, were implemented in the experiments. The modes were controlled via the graphical user interface of CDTRP. The constant motor speed was set to 120 rpm. The periodic changing speed was changed from 0 to 240 rpm as a pure (1Hz) sinusoid which was observed actually to be the maximum attainable frequency for the DC motor speed to track the sinusoidal PWM control signal. The chaotic motor speed was controlled to be changed in the interval of 0-240 rpm by signal conditioning. This means that DC motor was operated in a uni-directional way for all of the three mixing modes.

In order to evaluate the mixing performances of these different mixing modes, the evolutions of the mixing processes were recorded by a camcorder. The snapshots taken at each 5 seconds are given in Table 5.1. As shown in Table 5.1, neutralization as the sign of attained homogeneity of the corn syrup added acid-base mixture occurs

at 25 seconds, 30 seconds and 480 seconds for the chaotic, periodic and constant motor speeds, respectively.

Table 5.1 Snapshots along the evolutions of chaotic, sinusoidal periodic and constant mixing



To get a valid comparison, the power consumptions until the neutralization time were also recorded via measuring the DC motor current value using the ammeter. As reported in Table 5.2, the average current values and power consumptions are of comparable.

Table 5.2 Comparison of chaotic, periodic and constant mixing in terms of neutralization time and energy consumptions until neutralization

Mixing Type	Time (s)	Average Current (mA)	Average Power (W)	Energy (Wh)
Chaotic	25	119.73	1.795	0.012
Periodic	30	123.06	1.845	0.015
Constant	480	100.00	1.500	0.200

The least energy consumption which is actually the performance measure that would be taken into account for comparison is obtained for chaotic mixing since the time spent until the neutralization is the smallest for chaotic mixing while the power consumptions for all mixing modes are close to each other.

The chaotic mixing obtained by the DC motor chaotified based on the developed dynamical state feedback chaotification method provides slightly more efficient chaotic mixing as compare to the chaotic mixing method proposed in (Ye & Chau, 2007) which exploits the time-delay feedback type chaotification of (Wang et al., 2000). In (Ye & Chau, 2007), 300ml acid-base mixture is reported to be homogenously mixed by a similar mixing system actuated with an impeller mounted on a DC motor within 30 seconds for an average 4.2 W power consumption, outperforming the constant and also 0.5 Hz sinusoidal periodic speed mixing. In another chaotic mixing system (Zhang & Chen, 2005), rather than impeller, a plate supporting the tank containing sugar/sucrose in the same volume of water is rotated by a DC motor controlled with a chaotic PWM signal. It is reported in (Zhang & Chen, 2005) that the homogenous mixing is attained by this open loop chaotic mixing system within 24 minutes for an average 4.661675 W power consumption, again outperforming constant and also up to 1 Hz sinusoidal periodic speed mixing both almost doubling the required mixing time under the same power consumption. The liquid to be mixed is different from the acid-base mixture and the volume of the liquid is not reported in (Zhang & Chen, 2005). However, it can be argued just by considering the ratio, i.e. approximately 2, between the energy consumptions for the constant speed and chaotic speed cases of (Zhang & Chen, 2005) and by the ratio (16.66 = 200/12) following from Table 5.2 that the chaotic mixing system presented in this paper provides much more efficient mixing in terms of the consumed energy. The relative efficiency of the developed chaotic mixing system might be connected to the operating frequencies larger than 1 Hz (See Figure 5.8 (a).) that can be attained only for the DC motor chaotified by the dynamical state feedback.

5.6 Analog Circuit Application for DC Motor Chaotified by Lorenz Chaotic System

Considering wide spread of chaotification use in engineering applications and its generic feature, i.e. the reliance of its working principles on one of the four fundamental forces of the nature, i.e. electromagnetic interaction, the DC motor is chosen in this application as the case for the real system to be chaotified by the

introduced dynamical state feedback to possess chaotic dynamics, such as the one produced by the celebrated Lorenz system.

Now, consider the Lorenz system defined in Equation 5.13 (Cuomo & Oppenheim, 1993) as the reference chaotic system.

$$\begin{aligned} \dot{x} &= \sigma(y - x) \\ \dot{y} &= rx - y - 20xz \\ \dot{z} &= 5xy - bz \end{aligned} \tag{5.13}$$

Redefining the above Lorenz states x, y and z as

$$\begin{aligned} x_n &\hat{=} x \\ x_{n+1} &\hat{=} y \\ x_{n+2} &\hat{=} z \end{aligned} \tag{5.14}$$

and choosing the control input u as

$$u = a_1 x_1 + \cdots + a_{n-1} x_{n-1} - (\sigma - a_n) x_n + \sigma x_{n+1} \tag{5.15}$$

the linear system given in Equation 5.13 is augmented to the following nonlinear system of *(n+2)* nd order.

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \vdots \\ \dot{x}_{n-1} \\ \dot{x}_n \\ \dot{x}_{n+1} \\ \dot{x}_{n+2} \end{bmatrix} = \begin{bmatrix} 0 & 1 & \cdots & 0 & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 & 0 \\ 0 & 0 & \cdots & -\sigma & \sigma & 0 \\ 0 & 0 & \cdots & r & -1 & -20x_n \\ 0 & 0 & \cdots & 0 & 5x_n & -b \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \\ x_{n+1} \\ x_{n+2} \end{bmatrix} \tag{5.16}$$

With the above choice of the control input u, the last state equation of the system in Equation 5.13 becomes identical to the first Lorenz equation in Equation 5.14 under the change of variables in Equation 5.14. The state feedback defined by Equation 5.13 is a nonlinear dynamical state feedback since the control input u brings an extra state variable x_{n+1} which is nonlinearly and also dynamically dependent on the nth state variable x_n as can be observed from Equation 5.14 and Equation 5.15. The control input u in Equation 5.16 brings, in an indirect way, another extra state variable x_{n+2}, yielding the last three equations of the augmented system match to the Lorenz state equations in Equation 5.14 but with the new variables in Equation 5.15. So, the resulting nonlinear system in Equation 5.16 has chaotic dynamics exactly the same with the Lorenz system such that the dynamics of the last three state variables exactly matches to the states of the Lorenz system and the first *(n-1)* state variables are simply the integral, at different level, of the chaotic state x_n. It means that the new high dimensional system Equation 5.16 has a third order chaotic dynamics only.

The proposed chaotification scheme is depicted in Figure 5.11. Where, a part of the Lorenz system constitutes a dynamical controller together with the linear state feedbacks.

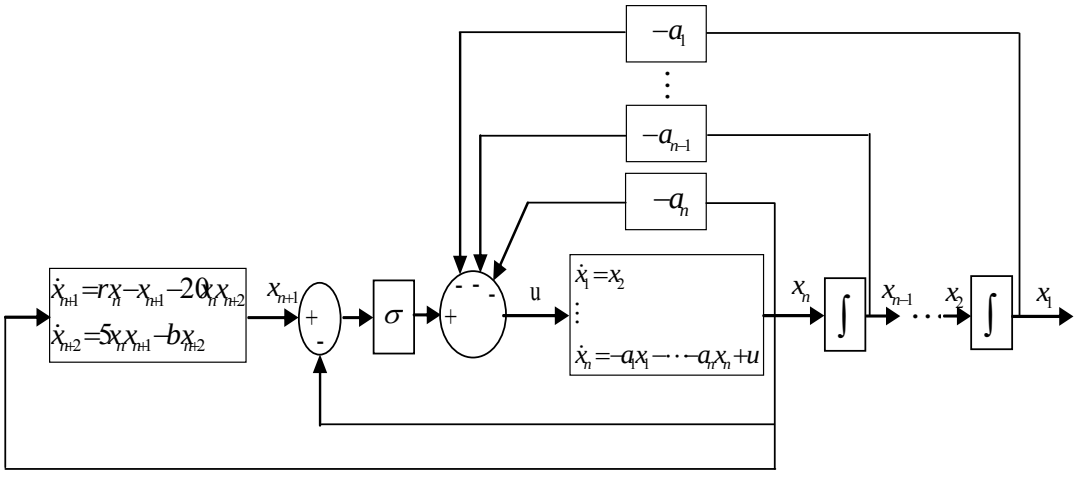


Figure 5.11 Block diagram of the proposed chaotification method based on Lorenz system for linear, time-invariant systems defined in the controllable canonical form

5.6.1 Implementation of Chaotified DC Motor System by Lorenz Chaotic System

To demonstrate the applicability and real life engineering application potentials of the introduced simple dynamical state feedback method, a permanent magnet DC motor is considered as the system to be chaotified due to the its wide spread engineering applications and due to its prominent status as having the features of the reliance of its working principles on one of the four fundamental forces of the nature, i.e. electromagnetic interaction.

In the experiments, an rf-310ta series DC motor which is easy to use and can be found easily with very low cost is used, so any one who is interested in the applications of the proposed chaotification method on a real DC motor may attempt to reproduce the experiments and observe how a chaotified DC motor runs. Just for simplicity, a first order simplified model is chosen for the permanent magnet DC motor: $\dot{x} = -a_m x + b_m u$. Where x stands for the angular velocity of the motor, i.e. the rpm and u stands for the armature input voltage. The first order system parameters of the used rf-310ta series DC motor were identified as $a_m = 2$ and $b_m = 5600$ by measuring its response due to the step input.

By the application of the proposed chaotification method to the DC motor for the considered simplified model and for the Lorenz chaotic system reference, the augmented system becomes a third order nonlinear system which is identical to the Lorenz system since the original system, i.e. the DC motor has the first dynamical model. The resulting system is given in Equation 5.17 where the original symbols of the Lorenz state variables, so that the first one is taken equal to the DC motor state, are preserved.

$$\begin{aligned} \dot{x} &= -a_m x - (\sigma - a_m)x + \sigma y = -a_m x + b_m u \\ \dot{y} &= rx - y - 20xz \\ \dot{z} &= 5xy - bz \end{aligned} \tag{5.17}$$

Herein, the control input is chosen as

$$u = \frac{1}{b_m}\left[-(\sigma - a_m)x + \sigma y\right] \tag{5.18}$$

The obtained chaotified DC motor system is depicted in Figure 2.

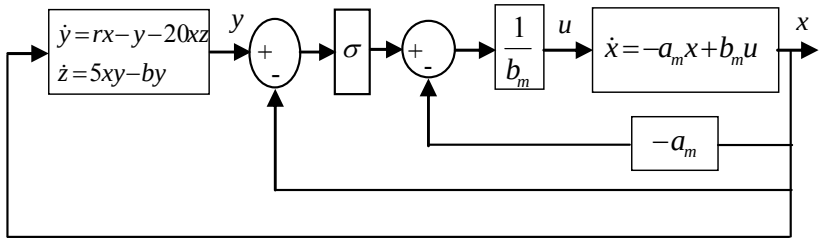


Figure 5.12 Diagram of the chaotified DC motor by Lorenz System

The whole DC motor chaotification system depicted in Figure 5.12 was implemented as a fully analog system in a modular fashion. The plant is the rf-310ta series DC motor which indeed implements the first state equation in Equation 5.17. The second and third state equations of the Lorenz system Equation 5.17 which constitute the nonlinear dynamical controller part were realized with the circuit configuration in Figure 5.13. Where two AD633 analog multipliers realize nonlinear terms, two LF353 opamps together with eleven resistors R1=R2=R5=100kΩ, R3=R4=1kΩ, R6=R8=10kΩ, R7=1MΩ, RV1=RV3=100kΩ, RV2=470kΩ and two capacitors C1=C2=10nF realize linear weighted summations and integrations. The motor speed, which is the state variable to be feedback, is measured by a tacho-generator which is again an rf-310ta series DC motor-generator connected to the shaft of the DC motor. The tacho-generator output voltage, considered here as proportional to the motor speed, constitutes the feedback signal which is again analog. The other part of the hardware realization comprised of two LF353 opamps realize the summers and amplifications in Figure 5.13 and provides the control input deriving the DC motor. The experimental set-up of the proposed chaotification method is given in Figure 5.14.

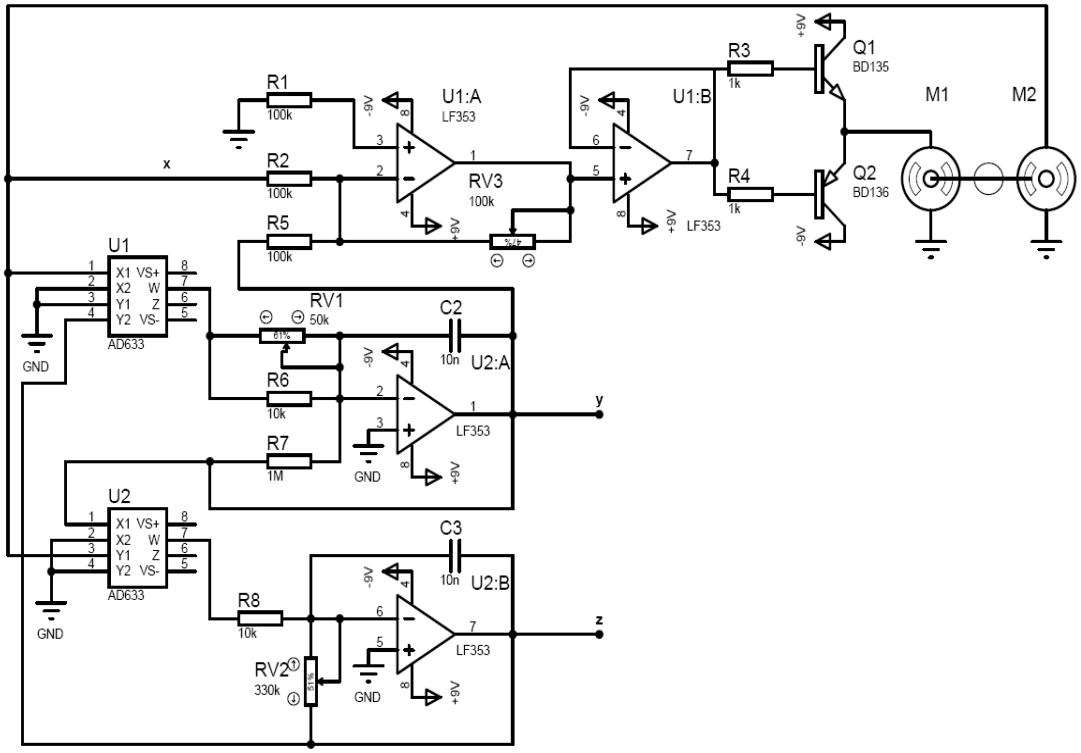


Figure 5.13 Fully analog realization of the chaotified DC motor system based on the first order DC motor model and Lorenz chaotic system reference

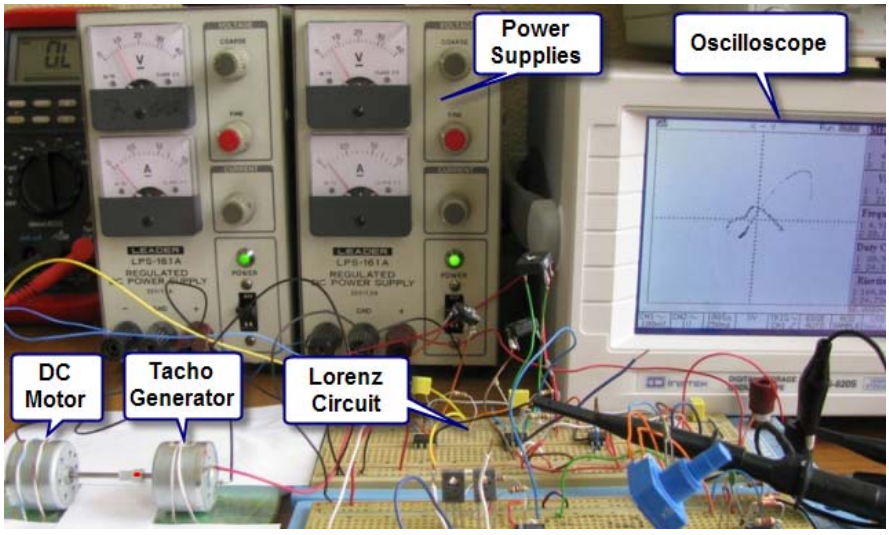
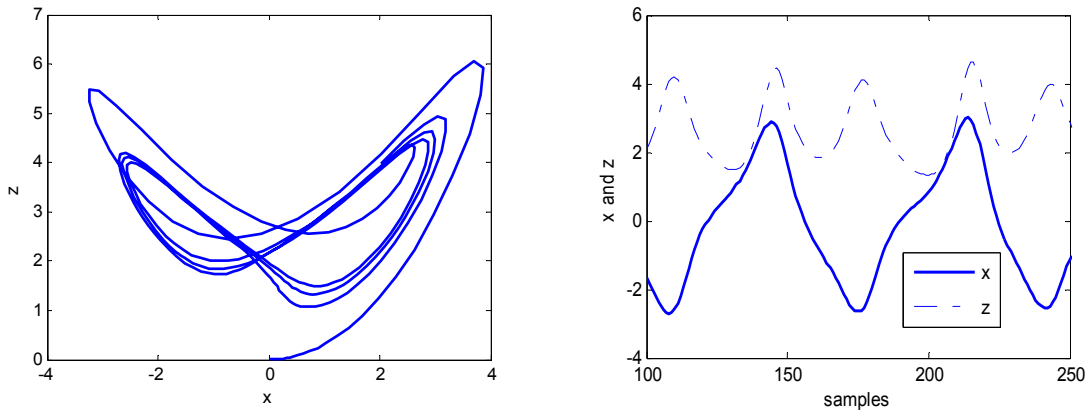


Figure 5.14 Experimental set-up for chaotification of DC motor

5.6.2 Simulations of the Proposed Chaotification System and its Bifurcation Diagrams Observed by Analog Circuit

The simulations of the chaotified DC Motor by Lorenz system are depicted in Figure 5.15 (a) and (b) where phase portrait for the x and z states of the simulated Lorenz chaotic system and time-waveforms of that are simulated, respectively. To achieve for the simulation program of the chaotified DC Motor by Lorenz system, an example program is given as a MATLAB m.file

```
am=2; bm=5600; sigma=10; r=56.6; b=5.02;
xm=[0.01, 0.01, 0.01]; tempxm=xm; delta_t=0.01;
for k=1:1000
xm(2)=tempxm(2)+(r*tempxm(1)-tempxm(2)-20*tempxm(1)*tempxm(3))*delta_t;
xm(3)=tempxm(3)+(5*tempxm(1)*tempxm(2)-b*tempxm(3))*delta_t;
u=(1/bm)*(-(sigma-am)*tempxm(1)+sigma*(tempxm(2)));
xm(1)=tempxm(1)+(-am*tempxm(1)+ bm*u)*delta_t;
end;
figure; plot(x,z); xlabel('x'); ylabel('x');
figure; plot(x,'-'); hold on; plot(z,'-.'); xlabel('samples'); ylabel('x and z');
```



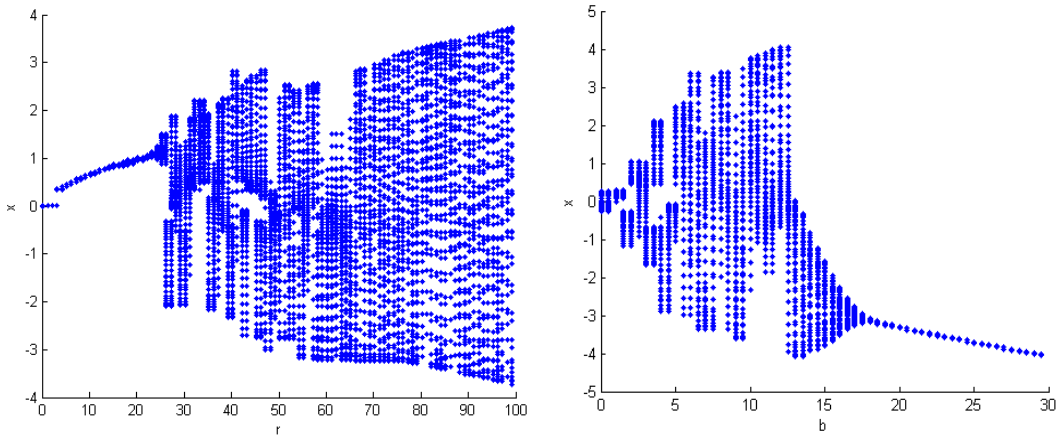
(a) (b)

Figure 5.15 (a) Phase portrait for the x and z states of the simulated Lorenz chaotic system (b) Time-waveforms of the x and z states of the Lorenz chaotic system which is simulated.

On the other hand, simulations program of bifurcation diagram for parameters *r* and *b* might be written as a MATLAB m.file. For example, bifurcation diagram for parameters *r* is given as a simulation program to the following

```
am=2; bm=5600; sigma=10; r=0.2; b=5.02; x=zeros(250,1);
xm=[0.01, 0.01, 0.01]; tempxm=xm; delta_t=0.01;  hold on;
while r<100
  for k=1:250,
  u=(1/bm)*(-(sigma-am)*tempxm(1) + sigma*(tempxm(2)));
  xm(1)=tempxm(1)+(-am*tempxm(1)+ bm*u)*delta_t;
  xm(2)=tempxm(2)+(r*tempxm(1)-tempxm(2) 20*tempxm(1)*tempxm(3))*delta_t
  xm(3)=tempxm(3)+(5*tempxm(1)*tempxm(2) -b*tempxm(3))*delta_t;
  tempxm=xm; x(k)=tempxm(1); y(k)=tempxm(2);    z(k)=tempxm(3);
  plot(r,x(200:250),'b.');
  end; r=r+1;
end; xlabel('r'); ylabel('x');
```

The simulated bifurcation diagrams of the chaotified DC Motor by Lorenz system are depicted in Figure 5.16 (a) and (b) where parameters *r* and *b*, respectively, of the simulated Lorenz chaotic system are simulated.



(a) (b)

Figure 5.16 The simulations of bifurcation diagrams of the Lorenz chaotic system (a) for *r* (b) for *b*.

On the other hand, the bifurcation diagrams of the chaotified DC Motor by Lorenz system might be given as snapshots of X-Y modes of the oscilloscope. For example, the snapshots of the X-Y mode of the oscilloscope for *b* parameter of the Lorenz chaotic system (implemented by RV2 potentiometer) are given in Figure 5.17 (a), (b), and (c).

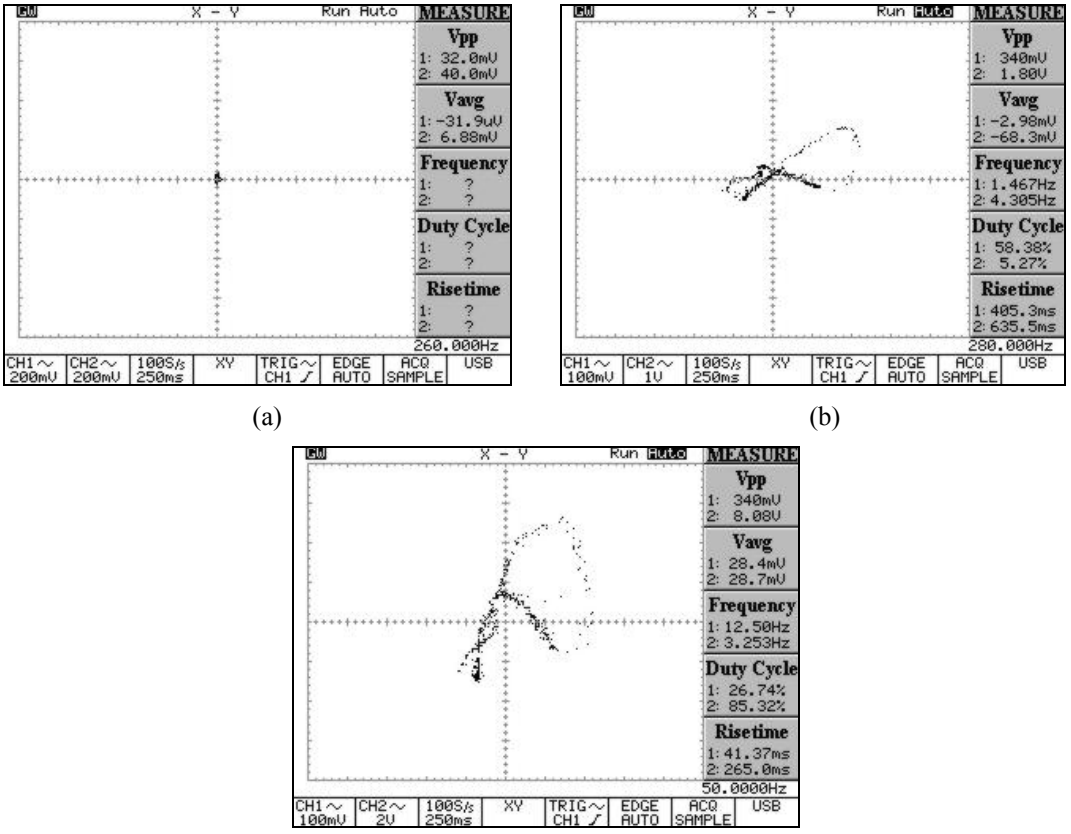


Figure 5.17 A snapshot on the X-Y mode of the oscilloscope where the signals in the x and z states of Lorenz circuits are fed, respectively. While RV1 20kOhm and RV3 50kOhm are constant, RV2 value is changed (a) 1kOhm, (b) 50kOhm and (c) 350kOhm.

Moreover, the snapshots of the X-Y mode of the oscilloscope for RV1 potentiometer of the implemented Lorenz chaotic system are given in Figure 5.18 (a), (b), and (c).

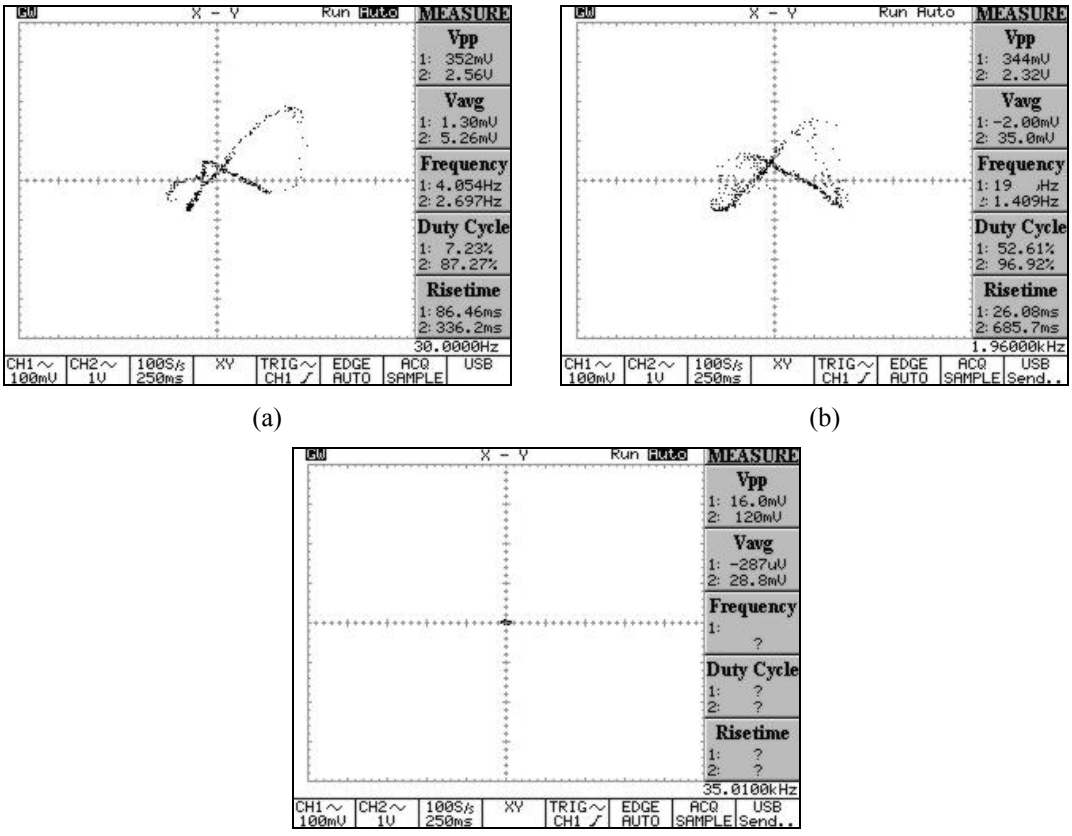


Figure 5.18 A snapshot on the X-Y mode of the oscilloscope where the signals in the x and z states of Lorenz circuits are fed, respectively. While RV2 and RV3 are 50kOhm, RV1 value is changed (a) 10kOhm, (b) 30kOhm and (c) 50-100kOhm.

CHAPTER SIX

CONCLUSION

A new real-time simulation-emulation design, test and redesign platform for controller design, a new input-output data based nonlinear dynamical adaptive controller design method, and a new chaotification method based on dynamical state feedback are developed in this thesis. The conclusions and future directions related to these three contributions are outlined in the sequel.

Controller-Design-Test-Redesign-Platform (CDTRP): The developed CDTRP enables to be embedded, via its hardware peripheral unit, real disturbances and any part or accessories of a real plant such as actuators and sensors which can be implemented in the laboratory conditions while the other parts are implemented in the simulator or emulator of the platform, so enabling users to approximate to the real plants and their real environments as much as they desire. The platform still provides the possibility of modifying and tuning the controller candidates based on their performances in a flexible way without causing any damage on the real plant or any other risky situation.

CDTRP can be operated in 24 different real-time operating modes where the controller, plant and peripheral unit are implemented either in the simulator or in the emulator or realized as an external analog or digital hardware depending on the application and on the memory-time and also other implementation requirements. The operating modes, a subset of which are introduced in this thesis as contributing to the real time simulation literature and another subset corresponds to the well-known simulation modes in the literature, are described in a novel taxonomy and also categorized based on their suitability to the design, test and redesign stages of the proposed controller design process.

As observed by the investigation conducted in this thesis, the abilities of the CDTRP platform in implementing the controllers and plants are limited by the

hardware and software realizations used for its simulator and emulator units, i.e. the MATLAB environment used for the PC simulator and the emulator implemented by using PIC microcontroller and also their communication abilities with each other and also with external hardware units. However, the operating frequency range, which was determined by employing the coupled oscillators in the experimental investigation conducted for examining the validity of the implementations of control systems dynamics in the CDTRP, can be enlarged by using a different simulation environment and an emulator with a more advanced hardware such as a digital signal processor rather than a microcontroller. Such software and hardware improvements on the CDTRP platform enlarge also the classes of plants and controllers which are reliably implementable in the developed platform. The implemented 24 real time simulation modes can be extended by using additional hardware such as an analog interface for the simulator (in PC) and/or by employing a different implementation unit for each component of the peripheral unit.

It can be further concluded that the proposed design, test and redesign procedure can be used in any simulation platform providing a hierarchy of operating modes from the flexible ones to the ones close to the reality and open to be developed in more precise manner in some focused control applications.

Learning algorithms for adaptive nonlinear dynamical controller design: The developed adaptive control algorithm, which employs ARMA and NARMA input-output models both for plant and the closed-loop system consisting of plant and controller, is suitable to run online based on measurement data. In the linear case, it can be viewed as an algorithm solving Diophantine equation in real-time using data measured from the plant not a model of the plant. The proposed learning algorithm for adaptive control has the possibility of implementing it as an Artificial Neural Network (ANN) choosing appropriate basis functions in NARMA models. As opposed to the inverse system based ANN controllers, it attempts to find a closed loop system to possess a desired behavior rather than attempting to find an inverse of the plant yielding a unity closed loop system.

The developed adaptive control scheme defines a kind of Model Reference Adaptive Control (MRAC) when the desired output of the plant is provided by a stable reference model and the controller parameters are updated directly based on the measured plant outputs in real-time without taking into account previous measurements. On the other hand, it defines a self-tuning adaptive control when the measured plant input-outputs within a chosen time window are first used for identifying a plant model and then for updating the controller parameters not at each time but at the sampled times with sampling period not less than the window length used for identification.

Stability analyses of the closed loop systems obtained for different implementations of the introduced adaptive control method can be realized as future works. The adaptive control learning algorithm can be improved by introducing some robustifying mechanism into the controller learning algorithm and/or plant parameter identification subroutines. The developed adaptive control scheme is open to be applied on handling nonlinearities and model/parameter uncertainties in specific control problems.

Dynamical state feedbak chaotification of input state linearizable systems: The developed chaotification method, which is based on matching a part of the system to a part of a chosen reference chaotic system, employs a dynamical state feedback. As compared to the known chaotification methods, the method is more advantageous due to its generality allowing to be applied into the whole class of input-state linearizable systems. It has also the possibility of exploiting the hardware implementations (and also theoretical analysis results.) already available in the literature without any modification or only with a linear change of variables for the realization of the developed chaotifying dynamical state feedback controller.

The potential of the proposed chaotification in real world applications is demonstrated via its application on chaotic liquid mixing system actuated by chaotified permanent magnet DC motor. The developed chaotic liquid mixing system is observed to outperform the other available chaotic liquid mixing systems as well

as the constant and periodical speed liquid mixing in terms of the time spent under the same or smaller power consumptions.

Possible extensions of the chaotification research realized in the thesis are two fold, i.e. of theoretical and of practical. In the theoretical direction, i) one can develop other dynamical state feedback chaotification methods which are not based on a reference chaotic system but on some available theoretical results for ensuring the chaotic behavior of the closed loop dynamics, ii) one can enhance the proposed method by adding novel features in order to meet some specifications on desired closed loop chaotic dynamics, and iii) one can enlarge the class of systems for which dynamical state feedback chaotification method is applied. In the practical direction, the application areas of the developed chaotification method can be enlarged and, in a focused application, accurate model/parameters can be obtained by employing suitable identification methods for a real system under consideration.

REFERENCES

Abdallah, C., Dawson, D., Dorato, P., & Jamshidi, M. (1991). Survey of Robust Control for Rigid Robots. *IEEE Control Systems Magazine,* 11(2), 24-30.

Abdollahi, F., Talebi H. A., & Patel R. V. (2006). A Stable Neural Network-Based Observer with Application to Flexible-Joint Manipulators. *IEEE Transactions on Neural Networks*, 17(1), 118-129.

Ackermann, J., & Blue, P. (2002). *Robust Control: the Parameter Space Approach* (2nd ed.). London: Springer.

Ahmed, M. S. (2000). Neural-Net-Based Direct Adaptive Control for a Class of Nonlinear Plants. *IEEE Transactions on Automatic Control*, 45(1), 119-124.

Alvarez-Hernández, M. M., Shinbrot, T., Zalc, J., & Muzzio, F. J. (2002). Practical Chaotic Mixing. *Chemical Engineering Science*, 57(1), 3749-3753.

Ascanio, G., Brito-Bazán, M., Brito-De La Fuente, E., Carreau, P. J., & Tanguy, P. A. (2002). Unconventional Configuration Studies to Improve Mixing Times in Stirred Tanks. *The Canadian Journal of Chemical Engineering*, 80(1), 558-565.

Aseltine, J. A., Mancini, A. R., & Sarture, C. W. (1958). A Survey of Adaptive Control Systems. *IRE Transactions on Automatic Control*, 6(1), 102–108.

Astolfi, A., Karagiannis, D., & Ortega, R. (2008). *Nonlinear and Adaptive Control Design with Applications*. Springer-Verlag.

Astrom, K. J. (1987). Adaptive Feedback Control. *Proceedings of the IEEE*, 75(2), 185-217.

Astrom, K. J., & Hagglund, T. (1995). *PID Controllers: Theory, Design, and Tuning* (2nd ed.). Research Triangle Park, NC: Instrument Society of America.

Astrom, K. J., & Hagglund, T. (2004). Revisiting The Ziegler–Nichols Step Response Method for PID Control, *Journal of Process Control*, 14, 635-650.

Astrom, K. J., & Murray, R. M. (2008). *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton, NJ: Princeton Univ. Press.

Astrom, K. J., & Wittenmark, B. (1980). Self-Tuning Controllers Based on Pole-Zero Placement, *IEE Proceedings.-D: Control Theory Applications,* 127(5), 120-130.

Astrom, K. J., & Wittenmark, B. (1994). *Adaptive Control*: Addison-Wesley, Longman Publishing Co., Inc. Boston, MA, USA.

Astrom, K. J., & Wittenmark, B. (1997). *Computer-Controlled Systems Theory and Design* (3rd ed.). Prentice Hall.

Bacic, M. (2005). On Hardware-in-the-Loop Simulation. *44th IEEE Conference on Decision and Control European Control Conference,* 1, 3194-3198.

Baek, S-M, & Kuc, T-Y. (1997). An Adaptive PID Learning Control of DC Motors. *IEEE International Conference on Systems, Man, and Cybernetics*, 3(1), 2877-2882.

Balcı, O. (2003). Verification, Validation, and Certification of Modeling and Simulation Applications. *Proceedings of the 35th Conference on Winter Simulation: Driving Innovation,* 1, 150-158.

Barreto G. A., & Araujo A. F. R. (2004). Identification and Control of Dynamical Systems Using the Self-Organizing Map. *IEEE Transactions on Neural Networks*, 15(5), 1244-1259.

Benaskeur A., & Desbiens A. (2002). Backstepping-Based Adaptive PID Control. *IEE Proceedings Control Theory and Applications,* 149(1), 54-59.

Betin, F., Sivert, A., Yazidi, A. & Capolino G. A. (2007). Determination of Scaling Factors for Fuzzy Logic Control Using The Sliding-Mode Approach: Application to Control of a DC Machine Drive. *IEEE Transactions on Industrial Electronics*, 54(1), 296-309.

Bishop, R. H. (2008). *The Mechatronics Handbook* (2nd ed.). CRC Press.

Blanchini, F. (2000). The Gain Scheduling and the Robust State Feedback Stabilization Problems. *IEEE Transactions on Automatic Control*, 45(11), 2061–2070.

Blanchini, F. Parisini, T. Pellegrino, F. A. & Pin, G. (2009). High-Gain Adaptive Control: A Derivative-Based Approach. *IEEE Transactions on Automatic Control*, 54(9), 2164-2169.

Bolton, W. (2004). *Instrumentation and Control Systems*. Oxford, UK: Elsevier Publishing Co.

Boyd, S. P., & Barratt C. H. (1991). *Linear Controller Design: Limits of Performance* Englewood Cliffs, NJ: Prentice-Hall.

Brockett, R. W. (1973). Lie Theory And Control Systems Defined on Spheres. *SIAM Journal on Applied Mathematics*, 25(2), 213-225.

Brown, R. G. (2004). *Smoothing, Forecasting and Prediction of Discrete Time Series*, Prentice-Hall, Englewood Cliffs, NJ.

Burghelea, T., Segre, E., Bar-Joseph, I., Groisman, A., & Steinberg, V. (2004). Chaotic Flow and Efficient Mixing in a Microchannel with a Polymer Solution. *Physics Review E,* 69(6), 66305-(1-8).

Cabrera, J. B. D., & Narendra, K. S. (1999). Issues in the Application of Neural Networks for Tracking Based on Inverse Control. *IEEE Transactions on Automatic Control*, 44(11), 2007–2027.

Cao, R., Hou, Z., & Zhang, W. (2008). The Model-Free Learning Enhanced Motion Control of DC Motor. *International Conference on Electrical Machines and Systems,* 1, 1222-1226.

Cao, Y., Sun, Y., & Mao, W. (1998). A New Necessary and Sufficient Condition for Static Output Feedback Stabilizability and Comments on Stabilization via Static Output Feedback. *IEEE Transactions on Automatic Control*, 43(8), 1110-1111.

Cassandras, C., & Lafortune, S. (2008). *Introduction to Discrete Event Systems* (2nd ed.). Springer Science Business Media, LLC.

Chen, C. T. (1984). *Linear System Theory and Design* (3rd ed.). Holt, Rinehart and Winston, Inc., Publishers, Orlando, Florida.

Chen, G. (1999). *Controlling Chaos and Bifurcations in Engineering Systems*. CRC Press, Boca Raton, FL.

Chen, G., & Dong, X. (1993). On Feedback Control of Chaotic Continuous-Time Systems. *IEEE Transactions on Circuits and Systems I,* 40(9), 591-601.

Chen, L. J., & Narendra, K. S. (2001). Nonlinear Adaptive Control Using Neural Networks and Multiple Models. *Automatica,* 37(8), 1245-1255.

Chen, L. J., & Narendra, K. S. (2003). Intelligent Contro1 Using Multiple Neural Networks. *International Journal of Adaptive Control and Signal Processing,* 17(6), 417-430.

Chen, L., & Narendra, K. S. (2004). Identification and Control of a Nonlinear Discrete-Time System Based on its Linearization: A Unified Framework. *IEEE Transaction on Neural Networks*, 15(3), 663-673.

Chen, S., & Billings, S. A. (1989). Modelling and Analysis of Non-linear Time Series. *International Journal of Control*, 50(6), 2151-2171.

Cheng, D., Tarn, T. J., & Isidori, A. (1985). Global External Linearization of Nonlinear Systems via Feedback. *IEEE Transactions on Automatic Control*, 30(8), 808-811.

Chien, K. L., Hrones, J. A., & Reswick, J. B. (1952). On the Automatic Control of Generalized Passive Systems. *ASME Transactions*, 74(1), 175-185.

Chua, L. O., Wu, C. W., Huang, A., & Zhong, G. Q. (1993). A Universal Circuit for Studying and Generating Chaos-Part I: Routes to Chaos, *IEEE Transactions on Circuits and Systems-I,* 40(10), 732-761.

Conte, G., Moog, C. H., & Perdon, A. M. (1999). *Nonlinear Control Systems (an Algebraic Setting)*. New York: Lecture Notes in Control and Information Sciences, Springer.

Craig, J. J., (1986). *Introduction to Robotics: Mechanics and Control*. Reading, Mass.: Addison-Wesley.

Cuomo, K. M., & Oppenheim, A. V. (1993). Circuit Implementation of Synchronized Chaos with Applications to Communications. *Physics Review Letters*, 71(1), 65-68.

Cuomo, K. M., Oppenheim, A. V., & Strogatz, S. H. (1993). Synchronization of Lorenz-Based Chaotic Circuits with Applications to Communications. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing,* 40(10), 626-633.

Cybenko, G. (1989). Approximation by Superpositions of a Sigmoidal Function. *Mathematics of Control, Signals, and Systems,* 2(4), 303-314.

Data, A., & Ioannou, P. A. (1994). Performance Analysis and Improvement in Model Reference Adaptive Control. *IEEE Transactions on Automatic Control*, 39(12), 2370-2387.

Ding, Z. (1999). Almost Disturbance Decoupling of Uncertain Nonlinear Output Feedback Systems. *IEE Proceedings Control Theory and Applications*, 146(2), 220-226.

Ding, Z. (2002). Adaptive Tracking with Complete Compensation of Unknown Disturbances for Nonlinear Output Feedback Systems. *IEE Proceedings Control Theory and Applications,* 149(6), 533-539.

Ditto, W. L., Rauseo, S. N., & Spano, M. L. (1990). Experimental Control of Chaos. *Physics Review E,* 65(26), 3211-3214.

Dorf, R. (1989). *Modern Control Systems* (5th ed.). Addison–Wesley, Reading, Mass.

Dorf, R. C., & Bishop, R. H. (2008). *Modern Control Systems* (11th ed.). Prentice Hall.

Doyle, F. J., Allgower, F., & Morari, M. (1996). A Normal Form Approach to Approximate Input-Output Linearization for Maximum Phase Nonlinear SISO Systems. *IEEE Transactions on Automatic Control*, 41(2), 305-309.

Doyle, J. C. (1983). Synthesis of Robust Controllers and Filters. *Proceedings of the IEEE Conference on Decision and Control San Antonio, TX*., 1, 109-114.

Doyle, J. C., Francis, B. A., & Tannenbaum, A. R., (1992). *Feedback Control Theory.* New York: Macmillan, 1992.

Doyle, J. C., Glover, K., Khargonekar, P. P., & Francis, B. A. (1989). State-Space Solutions to Standard H_2 and H_∞ Problems. *IEEE Transactions on Automatic Control*, 34(8), 831–847.

Dufour, C., Ishikawa, T., Abourida, S., & Belanger, J. (2007). Modern Hardware-In-the-Loop Simulation Technology for Fuel Cell Hybrid Electric Vehicles. *IEEE Vehicle Power and Propulsion Conference VPPC 2007*, 1, 432-439.

Dullerud, G. E., & Paganini, F. (2000). *A Course in Robust Control Theory: A Convex Approach*. New York: Springer.

Ezal, K., Kokotovic, P. V., Tee1, A. R., & Başer, T. (1999). Disturbance Attenuating Output-Feedback Control of Nonlinear Systems with Local Optimality. *Proceedings of the American Control Conference,* 1, 2578-2583.

Facchinetti, A., & Mauri, M. (2009). Hardware-in-the-Loop Overhead Line Emulator for Active Pantograph Testing. *IEEE Transactions on Industrial Electronics*, 56(10), 4071-4078.

Filatov, N. M., & Unbehauen, H. (2000). Survey of Adaptive Dual Control Methods. *IEE Proceedings of the Control Theory and Applications*, 147(1), 118-128.

Fliess, M. (1990). Generalized Controller Canonical Forms for Linear and Nonlinear Dynamics. *IEEE Transactions on Automatic Control,* 35(9), 994-1001.

Fradkov, A. L. (1994). Nonlinear Adaptive Control: Regulation-Tracking-Oscillations. *Proceeding IFAC Workshop New Trends in Design of Control Systems Smolenice*, 1, 426-431.

Fradkov, A. L., & Evans, R. (2005). Control of Chaos: Methods and Applications in Engineering. *Annual Reviews in Control,* 29(1), 33-56.

Fradkov, A. L., & Pogromsky, A. Y. (1999). *Introduction to Control of Oscillations and Chaos*. World Scientific, Singapore.

Franklin, G. F., Powell, J. D., & Emami–Naeini, A. (1991). *Feedback Control of Dynamics Systems* (2nd ed.). Addison–Wesley, Reading, Mass.

Franklin, G. F., Powell, J. D., & Workman, M. (2000). *Digital Control of Dynamic Systems* (3rd ed.). Addison Wesley, CA.

Fukuda, T., & Shibata, T. (1992). Theory and Applications of Neural Networks for Industrial Control Systems. *IEEE Transactions on Industrial Electronics*, 39(6), 472-491.

Galotto, L., Pinto, J. O. P., Filho, J. A. B., & Lambert-Torres, G. (2007). Recursive Least Square and Genetic Algorithm Based Tool for PID Controllers Tuning. *International Conference on Intelligent Systems Applications to Power Systems*, 1, 1-6.

Ge, S. S., & Wang C. (2002). Direct Adaptive NN Control of a Class of Nonlinear Systems, *IEEE Transactions on Neural Networks*, 13(1), 214-221.

Ge, S. S., Hang, C. C., & Zhang, T. (1999). Adaptive Neural Network Control of Nonlinear Systems by State and Output Feedback. *IEEE Transactions on Systems, Man and Cybernetics-Part B Cybernetics*, 29(6), 818-828.

Ghorbel, F., Hung, J. Y., & Spong, M. W. (1989). Adaptive Control of Flexible Joint Manipulators. *IEEE Control System Magazine*, 9(7), 9-13.

Goodwin, G. C., & Sin, K. S. (1984). *Adaptive Filtering Prediction and Control*, Prentice Hall, NJ.

Goodwin, G. C., Graebe, S. F., & Salgado, M. E. (2001). *Control System Design*. Englewood Cliffs, NJ: Prentice Hall.

Grizzle, J. W., & Kokotovic, P. V. (1988). Feedback Linearization of Sampled Data Systems. *IEEE Transactions on Automatic Control*, 33(9), 857-859.

Guo, L., & Chen, H. (1991). The Astrom–Wittenmark Self-Tuning Regulator Revisited and ELS-Based Adaptive Trackers. *IEEE Transactions on Automatic Control*, 36(7), 802–812.

Güvenç, B. A. & Güvenç, L., (2002). Robust Two Degree-of-Freedom Add-on Controller Design for Automatic Steering. *IEEE Transactions on Control Systems Technology*, 10(1), 137-148.

Hanselmann, H. (1996). Hardware-in-the-Loop Simulation Testing and its Integration into a CACSD Toolset. *IEEE International Symposium on Computer-Aided Control System Design,* 1, 152-156.

Harnby, N., Edwards, M. F., & Nienow, A. W. (1992). *Mixing in the Process Industries.* Oxford, U.K.: Butterworth-Heinemann.

Hatwell, M. S., Oderkerk, B. J., Sacher, C. A., & Inbar, G. F. (1991). The Development of a Model Reference Adaptive Controller to Control the Knee Joint of Paraplegics. *IEEE Transactions on Automatic Control*, 36(6), 683-691.

Hauser, J., Sastry, S., & Kokotovic, P. (1992). Nonlinear Control via Approximate Input-Output Linearization: The Ball and Beam Example. *IEEE Transactions on Automatic Control*, 37(3), 392-398.

Hedrick, J. K., & Girard, A. (2005). *Control of Nonlinear Dynamic Systems: Theory and Applications*. 07.04.2009, http://www.me.berkeley.edu/ME237/notes.html.

Hellerstein, J. L., Diao, Y., Parekh, S., & Tilbury, D.M. (2004). *Feedback Control of Computing Systems*. John Wiley & Sons.

Hermann, R., & Krener, A. J. (1977). Nonlinear Controllability and Observability. *IEEE Transactions on Automatic Control*, 22(5), 728-740.

Horowitz, P. (2009). *Build a Lorenz Attractor*. 10 August 2009, http://frank.harvard.edu/~paulh/misc/lorenz.htm

Hsia, T. C. S. (1989). A New Technique for Robust Control of Servo Systems. *IEEE Transactions on Industrial Electronics*, 36(1), 1-7.

Huang, S., Tan, K., Lee, T., & Putra, A. (2007). Adaptive Control of Mechanical Systems Using Neural Networks. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 37(5), 897-903.

Hunt, L.R., Su, R. & Meyer, G. (1983). Global Transformations of Nonlinear Systems. *IEEE Transactions on Automatic Control,* 28(1), 24-31.

Hyvärinen, A., & Oja, E. (1997). A Fast Fixed-Point Algorithm for Independent Component Analysis. *Neural Computation,* 9(7), 1483-1492.

İbrahim, D. (2006). *Microcontroller Based Applied Digital Control.* John Wiley & Sons, Ltd., West Sussex.

Ioannou, P. A., & Sun, J. (1996). *Robust Adaptive Control.* Prentice-Hall, Englewood Cliffs, NJ.

Isermann, R., Schaffnit, J., & Sinsel, S. (1999). Hardware-in-the-Loop Simulation for the Design and Testing Of Engine-Control Systems. *Control Engineering Practice,* 7(5), 643-653.

Isidori, A. (1985). The Matching of a Prescribed Linear Input-Output Behavior in a Nonlinear System. *IEEE Transactions on Automatic Control*, 30(3), 258-265.

Isidori, A. (1995). *Nonlinear Control Systems* (3rd ed.). Springer Verlag, London.

Isidori, A., & Byrnes, C. I. (1990). Output Regulations of Nonlinear Systems. *IEEE Transaction on Automatic Control*, 35(2), 131-140.

Isidori, A., Krener, A. J., Gori-Giorgi, C., & Monaco, S. (1981). Nonlinear Decoupling via Feedback: A Differential Geometric Approach. *IEEE Transactions on Automatic Control,* 26(2), 331-345.

Jurdjevic, V., & Sussmann, H. J. (1972). Control Systems on Lie Groups. *Journal of Differential Equations*, 12(2), 313-329.

Kalman, R. E. (1960). Contributions to the Theory of Optimal Control. *Boletin de la Sociedad Matematica Mexicana*, 5(1), 102-119.

Kalman, R. E. (1961). On the General Theory of Control Systems. *IRE Transactions on Automatic Control*, 1(1), 481–493.

Karacan, S., Hapoğlu, H., Cabbar, Y., & Alpbaz, M. (1997). Pole Placement Self Tuning Control for Packed Distillation Column. *Chemical Engineering and Processing,* 36(1), 309–315.

Kavaslar, F., & Güzeliş, C. (1995). A Computer-Assisted Investigation of a 2-D Array of Chua's Circuits. *IEEE Transactions on Circuits and Systems I,* 42(10), 721-735.

Kawato, M., Furukawa, K. & Suzuki, R. (1987). A Hierarchical Neural-Network Model for Control and Learning of Voluntary Movement. *Biological Cybernetics*, 57(1), 169–185.

Keel, L., Rego, J. & Bhattacharyya, S. (2003). A New Approach to Digital PID Controller Design. *IEEE Transactions on Automatic Control,* 48(4), 687–692.

Khalil, H. (1996). *Nonlinear Systems* (2nd ed.). Upper Saddle River, NJ: Prentice Hall.

Kocarev, L., & Parlitz, U. (1995). General Approach for Chaotic Synchronization with Application to Communication. *Physical Review Letters,* 74(25), 5028–5031.

Kokotovic, P. V. (1992). The Joy of Feedback: Nonlinear and Adaptive. *IEEE Control Systems Magazine,* 12(3), 7-17.

Kosmatopoulos, E. B. (2008). Adaptive Control Design Based on Adaptive Optimization Principles. *IEEE Transactions on Automatic Control,* 53(11), 2680-2685.

Krener, A. J. (2003). *A Brief Tutorial on Linear and Nonlinear Control Theory*. 07.04.2009, http://www.samsi.info/200 304/multi/akrener.pdf

Krstic, M., Kanellakopoulos, I., & Kokotovic, P. V. (1992). Adaptive Nonlinear Control without Overparameterization. *System and Control Letters*, 19(1), 177-185.

Krstic, M., Kanellakopoulos, I., & Kokotovic, P. V. (1994). Nonlinear Design of Adaptive Controllers for Linear Systems. *IEEE Transactions on Automatic Control*, 39(4), 738-752.

Ku, C. C., & Lee, K. Y. (1995). Diagonal Recurrent Neural Networks for Dynamic Systems Control. *IEEE Transactions on Neural Networks*, 6(1), 144–156.

Kucera, J. (1967). Solution in Large of Control Problem: x = (Au + Bu)x. *Ibid*, 17(92), 91-96.

Kuo, B.C. (1995). *Automatic Control Systems* (7th ed.). Prentice–Hall, Englewood Cliffs, N.J.

Kwan, C., Lewis, F.L., & Dawson, D.M. (1998). Robust Neural-Network Control of Rigid-Link Electrically Driven Robots. *IEEE Transactions on Neural Networks,* 9(4), 581-588.

Landau, I. D., & Zito, G. (2006). *Digital Control Systems: Design, Identification and Implementation-(Communications and Control Engineering).* Springer-Verlag London.

Landau, I. D., Lozano, R., & M'Saad, M. (1997). *Adaptive Control*. Springer, London, UK.

Lavretsky, E. (2009). Combined/Composite Model Reference Adaptive Control. *IEEE Transactions on Automatic Control,* 54(11), 2692-2697.

Lee, K. Y., Ko, H. S., Kim, H. C., Lee, J. H., & Park Y. M. (2001). A Free-Model Based Controller Design and Its Application to Power System Stabilization. *Proceedings of the American Control Conference Arlington/VA*, 1, 805-810.

Leith, D. J., & Leithead, W. E. (2000). Survey of Gain-Scheduling Analysis and Design. *International Journal of Control*, 73(11), 1001-1025.

Leung, T. P., & Qin, H.-S. (2001). *Advanced Topics in Nonlinear Control Systems*. World Scientific Pub. Co, Singapore.

Levin, A. U., & Narendra, K. S. (1993). Control of Nonlinear Dynamical Systems Using Neural Networks: Controllability and Stabilization. *IEEE Transaction on Neural Networks*, 4(2), 192–206.

Levin, A. U., & Narendra, K. S. (1996). Control of Nonlinear Dynamical Systems Using Neural Networks—Part II: Observability, Identification, and Control. *IEEE Transaction on Neural Networks*, 7(1), 30–42.

Lewis, F. L. (1996). Neural-Network Control of Robot Manipulators. *IEEE Expert*, 11(3), 64-75.

Lewis, F. L., Dawson D. M., & Abdallah C. T. (2004). *Robot Manipulator Control: Theory and Practice.* New York, NY: Marcel Dekker, Inc.

Lewis, F. L., Jaganathan S., & Yeşildirek A. (1999). *Neural Network Control of Robot Manipulators and Nonlinear Systems.* New York: Taylor and Francis.

Lewis, F.L., Liu, K. & Yeşildirek, A. (1995). Neural Net Robot Controller with Guaranteed Tracking Performance. *IEEE Transactions on Neural Networks,* 6(3), 703-715.

Li, H., Steurer, M., Shi, K., Woodruff, S., & Zhang, D. (2006). Development of a unified design, test, and research platform for wind energy systems based on hardware-in-the-loop real-time simulation. *IEEE Transactions on Industrial Electronics*, 53(4), 1144–1151.

Lightbody, G., & Irwin, G.W. (1995). Direct Neural Model Reference Adaptive Control. *IEE Proceeding Control Theory Applications*, 142(1), 31-43.

Limanond, S., & Si, J. (1998). Neural-Network-Based Control Design: An LMI Approach. *IEEE Transactions on Neural Networks*, 9(6), 1422-1429.

Lin, F. (2007). *Robust Control Design: An Optimal Control Approach*. John Wiley & Sons Ltd, England.

Lin, F. J. (1997). Real-Time IP Position Controller Design with Torque Feedforward Control for PM Synchronous Motor. *IEEE Transactions on Industrial Electronics*, 44(1), 398-407.

Lin, F. J., & Shen, P. H. (2006). Adaptive Fuzzy-Neural-Network Control for a DSP-Based Permanent Magnet Linear Synchronous Motor Servo Drive. *IEEE Transactions on Fuzzy Systems,* 14(4), 481–495.

Liu, G. P., & Daley, S. (2000). Optimal-Tuning Nonlinear PID Control of Hydraulic Systems. *Control Engineering Practice,* 8(9), 1045-1053.

Ljung, L. (1999). *System Identification Theory for the User*, (2nd ed.). Upper Saddle River: Prentice Hall.

Loh, R. N. K., & Lu, M. (2002). Design of One-Step-Ahead Prediction Observers for System Parameter Identification. *7th International Conference on Control, Automation, Robotics and Vision ICARCV*, 3, 1502-1507.

Lorenz, E. N. (1963). Deterministic Non-Periodic Flows, *Journal of the Atmospheric Sciences,* 20(1), 130-141.

Lu, B., Wu, X., Figueroa, H., & Monti, A. (2007). A Low-Cost Real-Time Hardware-in-The-Loop Testing Approach of Power Electronics Controls. *IEEE Transactions on Industrial Electronics*, 54(2), 919-931.

Lü, J. H., & Chen, G. R. (2002). A New Chaotic Attractor Coined. *International Journal of Bifurcation and Chaos,* 12(3), 659-666.

Maclay, D. (1997). Simulation Gets into the Loop. *IEEE Review*, 43(3), 109-112.

Mandal, A. K., (2006). *Introduction to Control Engineering Modeling, Analysis and Design.* New Age International (P) Ltd., Publishers.

Mao, K. Z. (2002). RBF Neural Network Center Selection Based on Fisher Ration Class Separability Measure. *IEEE Transactions on Neural Networks,* 13(5), 1211-1217.

Marino, R., & Tomei, P. (1993). Global Adaptive Output-Feedback Control of Nonlinear Systems. Parts I. *IEEE Transactions on Automatic Control,* 38(1), 17-32.

Mehta, S., & Chiasson, J. (1998). Nonlinear Control of a Series DC Motor: Theory and Experiment. *IEEE Transactions on Industrial Electronics*, 45(1), 134-141.

Meireles, M. R. G., Almeida, P. E. M., & Simões, M. G. (2003). A Comprehensive Review for Industrial Applicability of Artificial Neural Networks. *IEEE Transactions on Industrial Electronics*, 50(3), 585-601.

Micera, S., Sabatini, A. M., & Dario, P. (1999). Adaptive fuzzy control of electrically stimulated muscles for arms movements. *Medical and Biological Engineering and Computing*, 37(6), 680-685.

Middleton, R. H., Goodwin, G. C., Hill, D. J. & Mayne, D. Q. (1988). Design Issues in Adaptive Control. *IEEE Transactions on Automatic Control*, 33(1), 50-58.

Morgül, Ö. (2003). A Model-Based Scheme for Anticontrol of Some Chaotic Systems. *International Journal of Bifurcation and Chaos*, 13(11), 3449-3457.

Morgül, Ö., & Solak, E. (1996). Observer Based Synchronization of Chaotic Systems. *Physics Review E,* 54(5), 4803-4811.

Nagrath, I. J., & Gopal, M. (2006). *Control Systems Engineering.* New Age International (P) Ltd. Publishers.

Narendra, K. S. (1991). Intelligent Control. *IEEE Control Systems*, 11(1), 39-40.

Narendra, K. S. (1993). Hierarchical Neural Network Models for Identification and Control. *Engineering in Medicine and Biology Society Proceedings of the 15th Annual International Conference of the IEEE*, 287-287.

Narendra, K. S. (1996). Neural Networks for Control: Theory and Practice. *Proceedings of the IEEE*, 84(10), 1385-1406.

Narendra, K. S., & Annaswamy, A. M. (1989). *Stable Adaptive Systems*. Englewood Cliffs, NJ: Prentice-Hall, 1989.

Narendra, K. S., & Balakrishnan, J. (1994). Intelligent Control Using Fixed and Adaptive Models. *Proceedings of the 33rd Conference Decision and Control*, 2, 1680-1685.

Narendra, K. S., & Balakrishnan, J. (1997). Adaptive Control Using Multiple Models. *IEEE Transactions on Automatic Control*, 42(2), 171-187.

Narendra, K. S., & Kudva, P. (1974). Stable Adaptive Schemes for System Identification and Control-Part I. *IEEE Transactions on Systems, Man and Cybernetics,* 4(6), 542–551.

Narendra, K. S., & Lewis, F. L. (2001). Introduction to the Special Issue on Neural Network Feedback Control. *Automatica*, 37(8), 1147-1148.

Narendra, K. S., & Mukhopadhyay, S. (1997). Adaptive Control Using Neural Networks and Approximate Models. *IEEE Transaction on Neural Networks*, 8(3), 475-485.

Narendra, K. S., & Parthasarathy, K. (1990). Identification and Control of Dynamical Systems Using Neural Networks. *IEEE Transactions on Neural Networks*, 1(1), 4–27.

Narendra, K. S., & Venkataraman, S. T. (1995). Adaptation and Learning in Robotics and Automation, *Proceedings of IEEE International Conference on Robotics and Automation Nagoya*, 1, 1244-1249.

Nguyen, D. H., & Widrow, B. (1990). Neural Networks for Self-Learning Control Systems. *IEEE Control Systems Magazine,* 10(3), 18-23.

Noriega, J. R., & Wang, H. (1998). A Direct Adaptive Neural-Network Control for Unknown Nonlinear Systems and Its Application. *IEEE Transactions on Neural Networks*, 9(1), 27-34.

Ogata, K. (1994). *Discrete Time Control Systems* (2nd ed.). Prentice Hall.

Ogata, K. (1997). *Modern Control Engineering* (3rd ed.). Englewood Cliffs, NJ: Prentice-Hall.

Omatu, S., & Yoshioka, M. (1997). Self-Tuning Neuro-PID Control and Applications. *IEEE International Conference on Systems, Man, and Cybernetics,* 3, 1985-1989.

Omidvar, O. M., & Elliott, D. L. (1997). *Neural Systems for Control*. Elsevier Science & Technology Books.

Ott, E. (1993). *Chaos in Dynamical Systems*. Cambridge University Press, UK.

Ott, E., Grebogi, C., & Yorke, J. A. (1990). Controlling Chaos. *Physics Review Letter,* 64(11), 1196-1199.

Ottino, J. M., & Wiggins, S. (2004). Foundations of Chaotic Mixing. *Philosophical Transactions of the Royal Society of London*, A 362, 937-970.

Özer, M., İşler, Y., & Özer, H. (2004). A Computer Software for Simulating Single-Compartmental Model of Neurons. *Computer Methods and Programs in Biomedicine*, 75(1), 51-57.

Pan, Z., & Başar, T. (1998). Adaptive controller design for tracking and disturbance attenuation in parametric strict-feedback nonlinear systems. *IEEE Transactions on Automatic Control,* 43(8), 1066-1084.

Park, J.-W., Harley, R. G., & Venayagamoorthy, G. K. (2002). Comparison of MLP and RBF Neural Networks Using Deviation Signals for On-line Identification of a Synchronous Generator, *IEEE in Power Engineering Society Winter Meeting*, 1, 274-279.

Parker, T. S., & Chua, L. O. (1987a). Chaos–A Tutorial for Engineers. *Proceedings of The IEEE,* 75(8), 982–1008.

Parker, T. S., & Chua, L. O. (1987b). INSITE–A Software Toolkit for the Analysis of Nonlinear Dynamical Systems. *Proceedings of The IEEE,* 75(8), 1081–1089.

Paul, E. L., Atiemo-Obeng, V., & Kresta, S. M. (2004). *Handbook of Industrial Mixing: Science and Practice.* Wiley-Interscience, New York.

Pecora, L. M., & Carroll, T. L. (1990). Synchronization in Chaotic Systems. *Physical Review Letters,* 64, 821-824.

Peitgen, H. O., Jürgens, H., & Saupe, D. (1992). *Chaos and Fractals: New Frontiers of Science*. Springer-Verlag New York, Inc.

Pellegrinetti, G., & Bentsman, J., (1996). Nonlinear Control Oriented Boiler Modeling—A Benchmark Problem for Controller Design. *IEEE Transactions on Control Systems Technology*, 4(1), 57-63.

Phillips, C. L., & Troy, N. H. (1997). *Digital Control System Analysis and Design* (3rd ed.). Prentice Hall, NJ.

Pimpalkhare, A., & Bandyopadhyay, B. (1994). Comments on Stabilization via Static Output Feedback. *IEEE Transactions on Automatic Control,* 39(5), 1148-1148.

Pintelon, R., & Schoukens, J. (2001). *System Identification: A Frequency Domain Approach*. Piscataway, NJ: IEEE Press.

Poggio, T. & Girosi, F. (1990a). Networks for Approximation and Learning. *Proceedings of the IEEE* , 78(9), 1481-1497.

Poggio, T. & Girosi, F. (1990b). Networks and the Best Approximation Property. *Biological Cybernetics*, 63(1), 169-176.

Psaltis, D., Sideris A. & Yamamura, A. A. (1988). A Multilayered Neural Network Controller. *IEEE Control Systems Magazine,* 8(2), 17-21.

Rank, E. (2003). Application of Bayesian Trained RBF Networks to Nonlinear Time-Series Modeling. *Signal Processing*, 83(7), 1393-1410.

Rodriguez, F., & Emadi, A. (2007). A Novel Digital Control Technique for Brushless DC Motor Drives. *IEEE Transactions on Industrial Electronics*, 54(5), 2365-2373.

Rojas, C. R., Welsh, J. S., Goodwin, G. C., & Feuer, A. (2007). Robust optimal experiment design for system identification, *Automatica,* 43(6), 993–1008.

Rosenblum, M. G., Pikovsky, A. S., & Kurths, J. (1996). Phase Synchronization of Chaotic Oscillators. *Physical Review Letters,* 76(11), 1804-1807.

Rössler, O. E. (1979). An Equation for Hyperchaos. *Physics Letters A,* 71(2-3), 155-157.

Rugh, W. J. (1996). *Linear System Theory.* New Jersey, Upper Saddle River, Prentice Hall.

Salomonsson, D., Soder, L., & Sannino, A. (2008). An Adaptive Control System for a DC Microgrid for Data Centers. *IEEE Transactions on Industrial Applications*, 44(6), 1910-1917.

Sanner, R. M. & Slotine, J. J. E. (1992). Gaussian Networks for Direct Adaptive Control. *IEEE Transactions on Neural Networks*, 3(6), 837-863.

Sargent, R. G. (2004). Validation and Verification of Simulation Models. *Proceedings of the 36th conference on Winter simulation*, 1, 17-28.

Sastry, S. (1999). *Nonlinear Systems Analysis, Stability and Control.* Springer-Verlag NY, Inc.

Sastry, S. S., & Isidori, A. (1989). Adaptive Control of Linearizable Systems. *IEEE Transactions on Automatic Control*, 34(11), 1123-1131.

Sastry, S., & Bodson, M. (1989). *Adaptive Control Stability, Convergence, and Robustness*. Prentice Hall.

Savaci, F. A., Yalçın, M. E., & Güzeliş, C. (2003). Steady-State Analysis of Nonlinearly Coupled Chua's Circuits with Periodic Input. *International Journal of Bifurcation and Chaos*, 13(11), 3395-3407.

Selver, M. A., & Güzeliş, C. (2009). Semiautomatic Transfer Function Initialization for Abdominal Visualization Using Self-Generating Hierarchical Radial Basis Function Networks. *IEEE Transactions on Visualization and Computer Graphics*, 15(3), 395-409.

Sen, M., & A. Pena. (1997). Pole Placement in Discrete Plants with Unstable Zeros with Extensions to Adaptive Control. *Proceedings of the American Control Conference*, 5, 2687-2691.

Seong, C. Y., & Widrow, B. (2001a). Neural Dynamic Optimization for Control Systems-Part I: Background. *IEEE Transactions on Systems, Man, And Cybernetics-Part B: Cybernetics*, 31(4), 482-489.

Seong, C. Y., & Widrow, B. (2001b). Neural Dynamic Optimization for Control Systems-Part III: Applications. *IEEE Transactions on Systems, Man, And Cybernetics—Part B: Cybernetics*, 31(4), 502-513.

Sepulchre, R., Jankovic, M., & Kokotovic, P. (1997). *Constructive Nonlinear Control*. Springer-Verlag. New York.

Seto, D., Annaswamy, A. M. & Baillieul, J. (1994). Adaptive Control of Nonlinear Systems with Triangular Structure. *IEEE Transactions on Automatic Control,* 39(7), 1411–1428.

Sinha, S. C., Henrichs, J. T., & Ravindra, B. (2000). A General Approach in the Design of Active Controllers for Nonlinear Systems Exhibiting Chaos. *International Journal of Bifurcation & Chaos*, 10(1), 165-179.

Sjoberg, J., Zhang, Q., Ljung, L., Benveniste, A., Delyon, B., Glorennec, P., Hjalmarsson, H., & Juditsky, A. (1995). Nonlinear Black-Box Modeling in System Identification: A Unified Overview. *Automatica*, 31, 1691-1724.

Skoczowski, S., Domek, S., Pietrusewicz, K., & Broel-Plater, B. (2005). A Method for Improving the Robustness of PID Control. *IEEE Transactions on Industrial Electronics*, 52(6), 1669-1676.

Slotine, J. J. E. (1988). Putting Physics in Control the Example of Robotics. *IEEE Contr. Syst. Mag.*, vol. 8, 12-17.

Slotine, J. J. E., & Li, W. (1991). *Applied Nonlinear Control*. Prentice Hall.

Smith, R. S., & Doyle, J. C. (1992). Model Validation: A Connection Between Robust Control And Identification. *IEEE Transactions on Automatic Control*, 37(1), 942-952.

Sontag, E. D. (1990). *Mathematical Control Theory.* New York: Springer.

Soong, C. Y., & Huang, W. T. (2007). Triggering and Enhancing Chaos with a Prescribed Target Lyapunov Exponent Using Optimized Perturbations of Minimum Power. *Physics Review E,* 75(3), 36206-(1-7).

Spong, M. W., & Vidyasagar, M. (1987). Robust Linear Compensator Design for Nonlinear Robotic Control. *IEEE Journal of Robotics and Automation*, 3(4), 345-350.

Spong, M., Khorasani, K., & Kokotovic, P. V. (1987). An Integral Manifold Approach to the Feedback Control of Flexible Joint Robots. *IEEE Journal of Robotics and Automation,* 3(4), 291–300.

Spooner, J. T., Maggiore, M., Ordonez, R., & Passino, K. M. (2002). *Stable Adaptive Control and Estimation for Nonlinear Systems: Neural and Fuzzy Approximator Techniques,* New York: Wiley.

Sprott, J. C. (2000). A New Class of Chaotic Circuit, *Physics Letters A,* 266(1), 19-23.

Steurer, M., Edrington, C., Sloderbeck, M., Ren W., & Langston, J. (2009). A Megawatt-Scale Power Hardware-in-the-Loop Simulation Setup for Motor

Drives. *IEEE Transactions on Industrial Electronics*, in press, 1-1, Digital Object Identifier 10.1109/TIE.2009.2036639.

Sundareshan, M. K., & Condarcure, T. A. (1998). Recurrent Neural-Network Training by a Learning Automaton Approach for Trajectory Learning and Control System Design. *IEEE Transactions on Neural Networks*, 9(3), 354–368.

Suykens, J. A. K., De Moor, B., & Vandewalle, J. (2000). Robust Local Stability of Multilayer Recurrent Neural Networks. *IEEE Transactions on Neural Networks*, 11(1), 222–229.

Suykens, J. A. K., Vandewalle, J., & De Moor, B. (1996). *Artificial Neural Networks for Modeling and Control of Non-Linear Systems.* Boston, MA: Kluwer.

Syrmos, V. L., Abdallah, C., & Dorato, P. (1994). Static Output Feedback: A Survey. *Proceedings of the 33rd Conference on Decision and Control,* 1, 837-842.

Şahin, S., İşler, Y., & Güzeliş, C. (2010). A Microcontroller Based Test Platform for Controller Design. *IEEE International Symposium on Industrial Electronics*, *Bari/Italy,* accepted.

Şahin, S., Ölmez, M., & İşler, Y. (2009). Microcontroller-Based Experimental Setup and Experiments for SCADA Education. *IEEE Transactions on Education*, in press, 1-1, Digital Object Identifier 10.1109/TE.2009.2026739.

Takigawa, T., Ohmura, N., Yagyuu, K., & Kataoka, K. (2000). Information Complexity of Laminar Chaotic Mixing Field Produced by Two Parallel, Rotating Cylinders. *Forma*, 15(1), 273-280.

Tang, H., Weng, L., Dong, Z. Y., & Yan, R. (2009). Adaptive and Learning Control for SI Engine Model with Uncertainties. *IEEE/ASME Transactions on Mechatronics,* 14(1), 93-104.

Tarte, Y., Chen, Y. Q., Ren, W. & Moore, K. L. (2006). Fractional Horsepower Dynamometer - A General Purpose Hardware-in-The-Loop Real-Time Simulation

Platform for Nonlinear Control Research And Education. *Proceedings of the IEEE Conference on Decision and Control*, 1, 3912 – 3917.

Toscano, R. (2005). A Simple Robust PI/PID Controller Design via Numerical Optimization Approach. *Journal of Process Control,* 15(1), 81-88.

Trofino-Neto, A., & Kucera, V. (1993). Stabilization via Static Output Feedback. *IEEE Transactions on Automatic Control*, 38(5), 764-765.

Uykan, Z., Güzeliş, C., Çelebi, M. E., & Koivo, H. N. (2000). Analysis of Input-Output Clustering for Determining Centers of RBFN. *IEEE Transactions on Neural Networks,* 11(4), 851-858.

Vanecek, A., & Celikovsky, S. (1994). Chaos Synthesis via Root Locus. *IEEE Transactions on Circuits and Systems I,* 41(1), 59-60.

Vidyasagar, M. (1993). *Nonlinear Systems Analysis.* Second Edition, Prentice Hall, Englewood Cliffs, New Jersey.

Wang, L. (2009). *Model Predictive Control System Design and Implementation Using MATLAB.* Springer London.

Wang, L. F., Tan, K. C., & Pralzlad, V. (2000). Developing Khepera Robot Applications in a Webots environment. *Proceedings of 2000 International Symposium on Micromechatronics and Human Science*, 1, 71-76.

Wang, L., Hu, W., & Tan, T. (2003). Recent Developments in Human Motion Analysis. *Pattern Recognition,* 36(3), 585-601.

Wang, W. J., & Lee, T. T. (1988). Poles-Zeros Placement and Decoupling in Discrete LQG Systems. *IEE Proceedings.-D: Control Theory Applications,* 134(6), 388-394.

Wang, X. F., & Chen, G. (2000). Chaotifying a Stable LTI System by Tiny Feedback Control. *IEEE Transactions on Circuits and Systems I,* 47(3), 410-415.

Wang, X., Chen, G., & Yu, X. (2000). Anticontrol of Chaos in Continuous-Time Systems via Time-Delayed Feedback. *Chaos,* 10(4), 771-779.

Werbos, P. (1991). An Overview of Neural Networks for Control. *IEEE Control Systems,* 11(1), 40-41.

Widrow, B., & Bilello, M. (1993). Adaptive Inverse Control. *Proceeding of the International Symposium on Intelligent Control*, 1, 1-6.

Widrow, B., & Plett, G. L. (1996). Adaptive Inverse Control Based on Linear and Non-Linear Adaptive Filtering. *Proceedings., International Workshop on Neural Networks for Identification, Control, Robotics, and Signal/Image Processing,* 1, 30-38.

Widrow, B., Lehr, M., Beaufays, F., Wan, Eric., & Bilello M. (1993). Learning Algorithms for Adaptive Signal Processing and Control. *IEEE International Conference*, 1, 1-8.

Wiggins, S. (1992). *Chaotic Transport in Dynamical Systems.* Springer, New York.

Wittenmark, B. (1995). Adaptive Dual Control Methods: an Overview. *In IFAC Symposium on Adaptive Systems in Control and Signal Proceeding Budapest,* 1, 67-72.

Wolf, A., Swifta, J. B., Swinney H. L., & Vastano, J. A. (1985). Determining Lyapunov exponents from a time series. *Physica D*, 16(3), 285-317.

Wu, H. X., Hu, J., & Xie, Y. C. (2007). Characteristic Model-Based All-Coefficient Adaptive Control Method and Its Applications, *IEEE Transactions on Systems Man and Cybernetics Part C – Applications and Reviews,* 37(2), 213-221.

Xu, H., & Ioannou, P. A. (2003). Robust Adaptive Control for a Class of MIMO Nonlinear Systems with Guaranteed Error Bounds. *IEEE Transactions on Automatic Control*, 48(5), 728-742.

Yamamoto, T., Takao, K., & Yamada, T. (2009). Design of a Data-Driven PID Controller. *IEEE Transactions on Control Systems Technology,* 17(1), 29-39.

Yang, Y., & Huang, K. (1992). Adaptive Lattice Estimation and Control of a Manipulator with One Flexible Forearm. *IEE Proceedings-D,* 139(3), 237-244.

Yaowen, L., Guangming, G., Hong, Z., & Yinghai, W. (2000). Synchronization of Hyperchaotic Harmonics in Time-Delay Systems and its Application to Secure Communication. *Physics Review E,* 62(6), 7898-7904.

Ye, S. & Chau, K. T. (2007). Chaoization of DC Motors for Industrial Mixing. *IEEE Transactions on Industrial Electronics,* 54(4), 2024-2032.

Yegnanarayana, B. (1981). Design of ARMA Digital Filters by Pole-Zero Decomposition. *IEEE Transactions on Acoustics, Speech and Signal Processing,* 29(3), 433–439.

Yeşildirek, A. & Lewis, F. L. (1994). Feedback Linearization Using Neural networks. *IEEE World Congress on Computational Intelligence,* 1, 2539-2544.

Zeigler, B. P., & Kim, J. (1993). Extending the DEVS-Scheme Knowledge-Based Simulation Environment for Real-Time Event-Based Control. *IEEE Transactions on Robotics and Automation,* 9(3), 351-356.

Zhang, J., Li, C., Zhang, H., & Yu, J. (2004). Chaos Synchronization Using Single Variable Feedback Based on Backstepping Method, *Chaos, Solitons & Fractals,* 21(5), 1183-1193.

Zhang, T., & Jay-Kuo, C. C. (2001). Audio Content Analysis for Online Audiovisual Data Segmentation and Classification. *IEEE Transactions on Speech and Audio Processing*, 9(4), 441-457.

Zhang, T., Ge, S. S., & Hang, C. C. (1999). Design and Performance Analysis of a Direct Adaptive Controller for Nonlinear Systems. *Automatica,* 35(11), 1809-1817.

Zhang, Y., Shieh, L. S., & Dunn, A. C. (2004). PID Controller Design for Disturbed Multivariable Systems. *IEE Proceeding Control Theory Applications*, 151(5), 567–576.

Zhang, Z., & Chen, G. (2005). Chaotic Motion Generation with Applications to Liquid Mixing. *Europe Conference Circuit Theory and Design, Cork/Ireland*, 1, 107-110.

Zhou, K., Doyle J. C., & Glover, K. (1996). *Robust and Optimal Control*, Prentice-Hall, Englewood Cliffs, NJ.

Ziegler, J. G., & Nichols, N. B. (1942). Optimum Setting for Automatic Controllers. *Transactions of the A.S.M.E.*, 759-768.