Application of Fuzzy-Sliding Mode Control and Electronic Load Emulation to the Robust Control of Motor Drives

by

Zuhtu Hakan Akpolat

Thesis submitted to the University of Nottingham
for the degree of Doctor of Philosophy, April 1999

I would like to dedicate this work to my wife Ayse, my daughter Senem and my son Mirac for the continuous support they have given me. I would also like to dedicate it to my parents and my sisters.

Acknowledgements

I would like to express my most sincere gratitude to my supervisors, Dr. G.M. Asher and Dr. J.C. Clare, for their guidance and support over the course of this project.

I would like to especially thank Dr. J. Cilia for helping me through the first year in the setting up of the experimental rig. I would also like to thank Dr. M. Sumner for helping me in the transputer network implementation.

Finally I would like to thank my friends and my colleagues, especially Mr. C.S. Staines, Mr. C. Hewson, Mr. D. Butt, Mr. L. Empiringham, Mr. N. Teske, and Mr. R. Magill and the other research members of the Power Electronics, Machines and Control (PEMC) Group for their help during the course of this work.

Contents

Abstract

This thesis is concerned with the experimental investigation of robust speed control strategies for the industrial motor drive systems. The first objective of the thesis is to implement a high performance programmable dynamometer which can provide desired linear and non-linear mechanical loads for the experimental validation of the robust control methods. The discrete time implementation of the conventional dynamometer control strategy (the inverse model approach) is analysed and it is shown that this method suffers from the stability and noise problems. A new dynamometer control strategy, based on speed tracking and torque feed-forward compensation, is developed and successfully implemented in the experimental system. The emulation is placed in a closed loop speed control system and the experimental results are compared with the corresponding ideal simulated results for the validation of the dynamometer control strategy. The comparisons show excellent agreement for a variety of linear and non-linear mechanical load models and such a high performance experimental load emulation results are reported for the first time in research literature.

The second objective of the project is to investigate the Fuzzy Logic Control (FLC) and the Sliding Mode Control (SMC) approaches in order to develop a simple, algorithmic and practical robust control design procedure for industrial speed drive control systems. The Reaching Law Control (RLC) method, which is an approach to SMC design, and the FLC are used together in order to develop a practical robust speed control strategy. The robustness of the proposed control approach is tested for a variety of linear and non-linear mechanical loads provided by the dynamometer. Using the new robust control method, good output responses are obtained for large parameter variations and external disturbances.

Chapter 1

Introduction

1.1 Robust Control of Motor Drives and Project Motivations

Electrical drives, especially induction motor drives, are widely used in today's industries. In many applications, speed control is required, either to hold a speed under unknown disturbances (speed regulation) or to change the speed according to a reference profile (speed servos). The output of a speed controller is a torque demand and electric drives are called upon to have good torque control performance. In the past, separately excited DC machines were used in most high performance speed control applications because torque and motor flux could be controlled easily and independently. With the development of the vector control theory [1-3], often called Field Oriented Control (FOC), decoupled torque and flux control of induction motors is now possible and induction machine drives are beginning to replace DC machine drives for speed control applications (due to their robust, cheap construction and lack of sliding contacts). Fast torque control for permanent magnet and (switched) reluctance machines have also been developed. This thesis concerns the area of speed control and is thus applicable to all machine drive types having good torque and flux control properties. Vector controlled induction motors are used in the project of this thesis.

In most speed drive control systems, conventional PI or PID controllers are used due to their simplicity of design and implementation. However, the speed control performance obtained using a PI/PID controller is sensitive to the frequently seen uncertainties such as plant parameter variations, external load disturbances and unmodelled and non-linear dynamics of the plant. Therefore, a *robust* controller would be attractive in most industrial applications. From the control point of view, robustness is the property that the dynamic response (including stability of course) is satisfactory not only for the nominal plant transfer function used for the design but also for the entire class of plants (including disturbances) that express all the

uncertainties of the dynamic environment in which the real controller is expected to operate [4,5]. A controller is said to be robust if it gives satisfactory dynamic responses in the presence of parameter variations, external disturbances and unmodelled or non-linear dynamics of the plant. The problem of designing robust controllers is thus called robust control.

For the robust control of motor drives, a variety of approaches (e.g., sliding mode, fuzzy, two-degree-of-freedom, torque feed-forward and adaptive control methods) have been investigated [6-19] and are still under investigation by many researchers. There are two interesting observations about many, if not most, published methods. Firstly, none of them seem to have been implemented in industrial drives currently on the market. Secondly, most robust methods are compared with conventional PI control for fixed loads; they are not compared over a range of load parameters that an industrial drive is likely to meet in practice.

Considering the first observation, recent robust drive control methods [6-19] are either theoretically complex and difficult to implement (containing algorithmic parameters even more difficult to set than PI coefficients!), or simple but not very effective in the case of large parameter variations. Among the many control techniques used for the robust control of electrical drives, the most popular approaches are Sliding Mode Control (SMC) and Fuzzy Logic Control (FLC) methods due to their simplicity and supposed effectiveness. These methods will be briefly discussed in Section 1.3. In some recent researches [13-16], the SMC method (also called Variable Structure Control (VSC)) has been supported by model following and adaptive control techniques to increase the effectiveness of the controller. However, these additional mechanisms increase the complexity of the controller (even more parameters to set) and make the practical implementation difficult. A simple, effective and practical robust motor drive control method is desirable in many industrial applications. It is one of the motivations of this thesis to investigate simple robust algorithms that are effective and can be transferred to industry.

Considering the second observation, due to the lack of providing a variety of linear and non-linear mechanical loads, the experimental validations of the robust control methods are usually not very satisfactory or convincing. This is simply because it is not easy to have different kinds of linear and non-linear mechanical loads in a laboratory. A desirable solution is to *emulate* mechanical loads using a *programmable dynamometer* (load machine). This allows realistic testing of electrical drives and also provides desired linear and non-linear loads for the experimental validation of the control strategies. At the outset of the project, it was thought that

implementing a programmable dynamometer would be straightforward. It was not. It turned out to be a very interesting problem and a substantial part of the project work addresses it.

1.2 Electronic Emulation of Mechanical Loads

The use of static dynamometers is very common in the testing of electrical machines [20-22]. In these applications, the electrical machines are tested in steady state or slowly changing conditions. Emulation of dynamic load models have been considered in some recent researches [23-27]. The general approach is to use the *inverse mechanical dynamics*, in which the shaft speed is measured and used to drive the reference torque for the dynamometer. In these papers, only simulation results of the continuous system are presented. However, in practice, the emulation strategy will be implemented on a microprocessor. We need therefore to consider sampling effects and noise problems. When such control is implemented in a real time control system, the output of the inverse model (i.e., the torque reference for the dynamometer) may be very noisy due to the derivative of measured noisy speed signal (note that inverse dynamics usually contains at least one first order derivative term). Furthermore, the emulation fails and becomes unstable if the emulated inertia is much larger than the actual drive inertia. This will be analysed and discussed in details in Chapter 3.

In the previous studies [23-27], the control structures are not designed to achieve exact dynamic matching; rather, the purpose is to achieve an acceptable time response matching for *open loop* emulation (i.e., the emulation is not a part of a *closed loop* control system). Note that if filtering or other measures are introduced to counteract the stability problems, an acceptable open loop performance may be obtained. However, due to the violated pole-zero structure of the desired mechanical load (open loop system), such emulation can not be used in a closed loop control system.

In [24,25] a model-reference approach is presented in which it is implied that the shaft speed or position could be used as a tracking variable and so avoid the inverse dynamics. However, this is not clear since the authors present results in which the shaft torque is the tracked variable; neither is the preservation of the mechanical dynamics (pole-zero structure) addressed in [24,25]. In [28], an integrator back-stepping design technique is presented which claims to emulate a dynamic load under closed loop conditions. However, the desired torque trajectory is still derived from an

inverse mechanical model and only simulation results are given. In fact, all the researchers in [23-28] present only simulation results. This is also discussed in Chapter 3.

Thus, a new dynamometer control strategy, which will preserve the dynamics of a desired load model when the emulation is placed in a closed loop speed control system, is required. The dynamometer should of course be able to emulate both linear and non-linear loads accurately.

1.3 Fuzzy Logic and Sliding Mode Control

Fuzzy set theory, introduced by Zadeh in 1965 [29], has found wide applications in many practical control systems as well as in the control of electrical drives. FLC has emerged as a practical alternative to the conventional control techniques since it provides a *decision making mechanism* which allows the control law to be conveniently changed in order to deal with parameter variations and external disturbances if the input variables (criteria for the decision making) to the Fuzzy Controller (FC) are chosen properly.

The classical linear controllers (e.g., PI, PD, PI+lead, etc.,) are still the most widely used controllers in the industrial applications due to their simplicity of design and microprocessor implementations. Although the research in the field of FLC is growing rapidly, the exact equivalence between classical controllers and FCs has not been well established in order to provide an easy transition between the control methods. It may be thought that there is no point in implementing a linear control law by a FC; however, the equivalence can be used as a preliminary step for designing robust FCs, since it provides a method of designing different classical control laws at different operating conditions and then to use FC as an interpolator. In addition, although there are many successful fuzzy speed and position control applications, usually these controllers are designed by trial and error methods [17,30]. In most cases, no formal approach is used to choose the number and shape of the membership functions. The derivation of the fuzzy equivalence of a linear controller actually generates an automatic design procedure for the FCs. Finally, from another aspect, the equivalence principle may also help to obtain a fair comparison between fuzzy and linear controllers. There are many research papers presenting such performance comparisons between fuzzy and linear controllers [17,31-34]. Some of these papers result in a doubt about the fairness of the comparisons. It is reasonable to assume that in order to have a fair comparison, the controllers under evaluation should give exactly or very similar closed loop output responses to the same input references for same

nominal condition [31]. Hence, when the parameters of the plant are changed or an external disturbance is applied, one can easily see which method gives the more robust control performance. Using the equivalence principle, the FC under evaluation may be designed to satisfy this comparison criteria.

In this thesis, the equivalence principle is investigated and shown to be a powerful technique for deterministic non-linear systems. For non-deterministic systems (e.g., a linear system with unknown parameters), this approach can not used directly since the decision making mechanism of the FC requires information about system non-linearity and/or parameter variations to chose an appropriate control action. For this reason, the robust investigations moved on to considering the SMC (with and without fuzzy implementation) and its variants.

The SMC approach to the robust control of motor drives [9-16] is probably the most popular method since it is actually developed for the control of uncertain and non-linear systems [35]. The main disadvantage of the SMC method is the assumption that the control signal can be switched from one value to another at infinite rate. In practical systems, however, it is impossible to manage this since the microprocessor implementation of the control strategy requires a finite sampling time. Direct microprocessor application of the SMC method results in a high frequency oscillation (chattering) about the desired equilibrium point. This is generally undesirable since chattering excites the unmodelled high frequency dynamics of the systems. A significant research effort has been directed at eliminating or reducing the chattering [35].

A new SMC design technique called Reaching Law Control (RLC) has been introduced by Gao and Hung in [36]. This approach not only establishes a reaching condition to the sliding surface directly but also specifies the dynamic characteristics of the system during the reaching phase. Additional merits of the RLC approach include simplification of the solution for SMC and providing a measure for the reduction of chattering. Since the RLC approach is quite new and the classical SMC is a well known technique, there are only a few practical applications of the RLC approach to motor drive control systems [69,70]. Neither contains a mathematical description of their implementation or any comparative studies, and it is felt that more practical researches are needed in order to clarify the effectiveness of the RLC approach in the control of electrical drives.

The relation between SMC and FLC, and the use of both control methods in a single controller are also new and attractive research areas [16,37-44]. Obviously, the researchers combines

these methods to use the advantages of both methods and thus to improve the effectiveness of the controllers. However, the combinations of the methods usually introduce complexities in the design and results in difficulties in microprocessor implementations. As far as the industrial applications are concerned, the combination of the SMC and FLC methods will be more acceptable if the resultant control structure demands the absolute minimum (ideally none at all) of control parameters to be set for a given application.

1.4 Objectives of the Thesis

The first objective of the thesis is to develop and implement a high performance load emulation strategy using a dynamometer which will provide desired linear and non-linear mechanical loads for the experimental validations of the drive control strategies. The emulation should preserve the dynamics of the desired load model when it is placed in a closed loop control system because providing accurate load dynamics in a closed loop system is quite important for the experimental validations of the robust control strategies.

The second objective of the thesis is to investigate the FLC and SMC (including the RLC method) approaches in order to develop a simple, algorithmic and practical robust control design procedure for industrial drive control systems. The target is to combine the FLC and SMC methods in a common framework by keeping the complexity of the control structure as small as possible. This is a requirement for many industrial applications.

1.5 Thesis Overview

The material covered in each chapter is briefly given below :

Chapter 2 explains the experimental system implemented for the realisation of the project objectives. The experimental system consists of two 0.55kW vector controlled induction machine drives. The real time control is achieved by a parallel processor network composed of transputers and some interfaces. The chapter describes the task of each transputer. A brief description of the hardware interface circuits is also included in the chapter.

Chapter 3 is dedicated to the emulation of mechanical loads. The conventional inverse model approach is analysed and the problems of this approach are discussed. New approaches based on speed tracking and torque feed-forward compensation are developed and implemented in the experimental system. For the validation of the emulation strategy, the experimental results are compared with the corresponding *ideal* simulation results obtained using SIMULINK/MATLAB software package. The linear and non-linear mechanical loads are accurately emulated when the emulation is placed in a closed loop speed control system.

In Chapter 4, the equivalence of the fuzzy and classical controllers are initially derived for a second order controller which represents the most widely used classical controllers such as PI, PD, PID, PI+lead, etc. The equivalence is also derived for a general controller transfer function with m-zeros and n-poles. Hierarchical fuzzy control structures are employed in order to reduce the number of rules in the FC. This method is then used to design a robust FC for a class of deterministic non-linear systems. Chapter 4 argues that for non-deterministic systems (e.g., a speed control system with unknown inertia and friction), the equivalence design approach can not be used directly since the decision making mechanism of the FC requires information about the system parameter variations in order to calculate an appropriate control action. For speed control applications, the speed error and the change of error, which are the usual inputs to the FC, do not provide required information about the parameter variations to compensate the effects of the varying parameters. Thus, in order to calculate a proper control action in the case of the parameter variations, a method is required to provide the necessary information for the decision making mechanism of the FC.

Chapter 5 describes a practical robust speed controller design procedure developed using the SMC (RLC) and FLC control approaches. The chapter discusses the discrete time implementation of the SMC strategy and the chattering problem. The RLC approach is applied for the design of the speed controller and also found more appropriate for the speed control applications in which the torque demand limitation is required to protect the drive power electronics and the machine. This is also discussed in Chapter 5. In order to get useful information about the system parameter variations and external disturbances, a reference switching function trajectory is derived using the *reaching law* designed for the nominal parameters of the system. This reference trajectory is then compared with the real switching trajectory and the error is interpreted by the decision making mechanism of the FC which changes the control actions appropriately in the case of parameter variations and disturbances. The proposed robust control design procedure is thus based on the RLC and FLC approaches.

The noise problem (speed encoder resolutions) is taken into account in the design of the final controller in order to have a realistic controller for the practical systems. The new control strategy is implemented in the experimental system and validated against some linear and non-linear loads and external disturbances provided by the programmable dynamometer.

Finally, Chapter 6 gives an overview of what has been achieved and discusses further work which may be a base for future projects.

Chapter 2

Experimental System

2.1 Introduction

This chapter describes the details of the experimental system used in this research project. The experimental system consists of two vector-controlled induction motor drives, a microprocessor system (composed of transputers, A/D converters and an I/O module installed in a PC) and some interface circuits. Fig. 2.1 show a photograph of the experimental implementation and the block diagram of the experimental system is illustrated in Fig.2.2, where '1' and '2' refer to the drive machine (motor) and the load machine (dynamometer) respectively. The drive machine and its inverter provide the target system for research into motion control strategies. The load machine is controlled so that the mechanical rig dynamics, defined as the speed response to a given drive torque, is equivalent to that of a desired linear or non-linear mechanical load dynamics. The details of the load emulation strategy will be explained in Chapter 3.

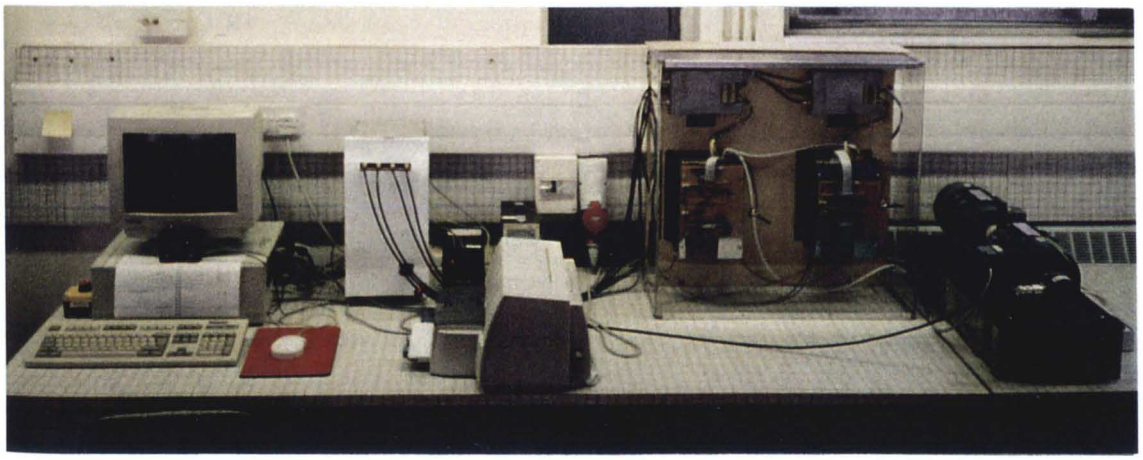


Figure 2.1 Experimental system

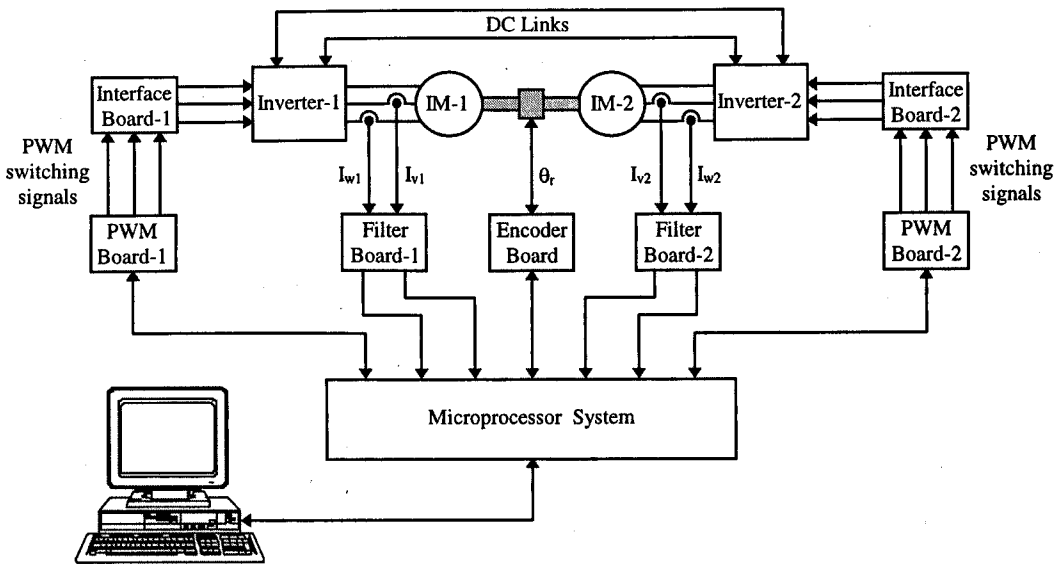


Figure 2.2 Block diagram of the experimental system

The DC links of the inverters are connected to each other to circulate the energy so that a dynamic braking unit is not required for the dynamometer. In this case, the mains supplies only the drive losses. The vector control is based on the Indirect Rotor Flux Orientation (IRFO) [1] which will be briefly summarised in Section 2.2. The individual components of the experimental system will be explained in the following sections in more detail.

2.2 Control of Induction Machines using Indirect Rotor Flux Orientation

The Field Oriented Control (FOC) or Vector Control is a standard control method for induction motors in adjustable speed drive applications. Vector control effectively "transforms" the AC machine into a "DC machine equivalent" in which a torque producing and field producing current may be defined. Torque and flux can thus be independently controlled as in a DC machine. In order to transform the machine into the "DC machine equivalent", the rotor flux angle (defining the position of the rotor flux vector in space) must be known at all times. This angle defines also the position of a rotating d-q axis frame. When d-axis of the frame is aligned with the rotor flux, the system is said to be Field Oriented. The dynamic equations of an induction machine are given in Appendix-A together with the derivation of the equations in field orientated form. There are a number of vector control strategies which are either termed

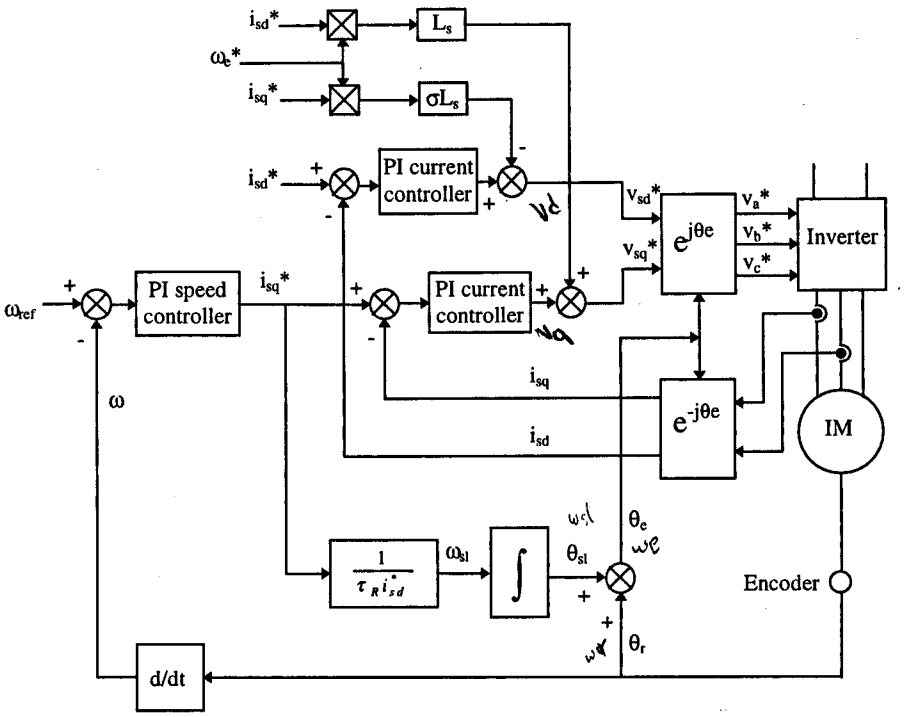


Figure 2.3 Speed control of an induction machine using IRFO method

direct or *indirect* and utilise either *impressed voltages* or *impressed currents* [45]. In the direct vector control, the rotor flux angle is either measured or derived via an observer at every sample instant. In the indirect vector control [1-3], the rotor flux angle θ_e is not calculated explicitly, rather it is derived using (assuming constant motor flux)

$$\omega_{sl} = \frac{i_{sq}^*}{\tau_R \cdot i_{sd}^*} \tag{2.1a}$$

$$\theta_e = \theta_r + \int \omega_{sl} dt \tag{2.1b}$$

where i_{sq}^* and i_{sd}^* are the reference q and d axis currents in the d-q field oriented frame, τ_R is the rotor time constant equal to L_R / R_R (L_R and R_R are the rotor self inductance and the rotor resistance per phase) and θ_r is the measured rotor position. The field orientation is achieved by imposing a slip frequency ω_{sl} according to (2.1a). If the rotor time constant τ_R is accurately matched to the actual machine value, the d axis will be inherently aligned to the rotor flux axis. This type of vector control is called Indirect Rotor Flux Orientation (IRFO) and is illustrated in Fig.2.3. In the figure, L_s is the stator self inductance, σ is the leakage coefficient and $e^{j\theta_e}$ and

$e^{-j\theta_e}$ denote the co-ordinate transformations (see Appendix-A). A more detailed description of the theory of this type vector control can be found in [1-3].

If the estimate of τ_R is incorrect (e.g., because of rotor heating), the calculated flux angle will be incorrect. From a control point of view, the effect of this is to reduce the real torque response in the machine (the response of i_{sq} is the same but of course i_{sq} is no-longer a pure torque producing current). However, even a degraded torque response (providing the error in the flux angle is not too large) is much faster than the response of the outer speed loop. In practice therefore, it has been found the errors in τ_R of 20-30% have a scarcely noticeable effect on speed transient and disturbance rejection capability. This can be seen in the figures of Chapter 3 comparing the experimental and simulated results and showing excellent agreements. Note that the simulated systems are the ideal or target systems that we want to implement on the experimental system. However, it was found that after about one hour continuous operation, due to the rotor heating, the effect of the incorrect τ_R became more noticeable in that the agreement of experimental and simulated results started to deteriorate. Therefore, all the experimental results presented in the thesis have been taken within the first hour of the operations.

2.3 The Microprocessor System

Transputers are specially developed processors for implementing parallel processing systems [46,47]. In the experimental system shown in Fig.2.1 and 2.2, the data acquisition of the input signals, processing and the generation of the control signals are achieved using a transputer network illustrated in Fig.2.4. Transputers are chosen for this project due to their availability in the department and the capability of implementing parallel processing systems. They also offer a good flexibility in implementing different types of algorithms [46-50].

The transputer network consists of five T805 floating point transputers, two A/D converter trams (Sunnyside ADT102) and one digital I/O tram (Sunnyside IOT332). All the transputers and the trams are located on two TMB08 transputer mother boards (one is slaved to the other) which are installed in a PC. A Pascal graphical interface program [48] running on the PC provides the user interface with the transputer network, allowing for on-line data storage/display and on-line change of parameters and set points.

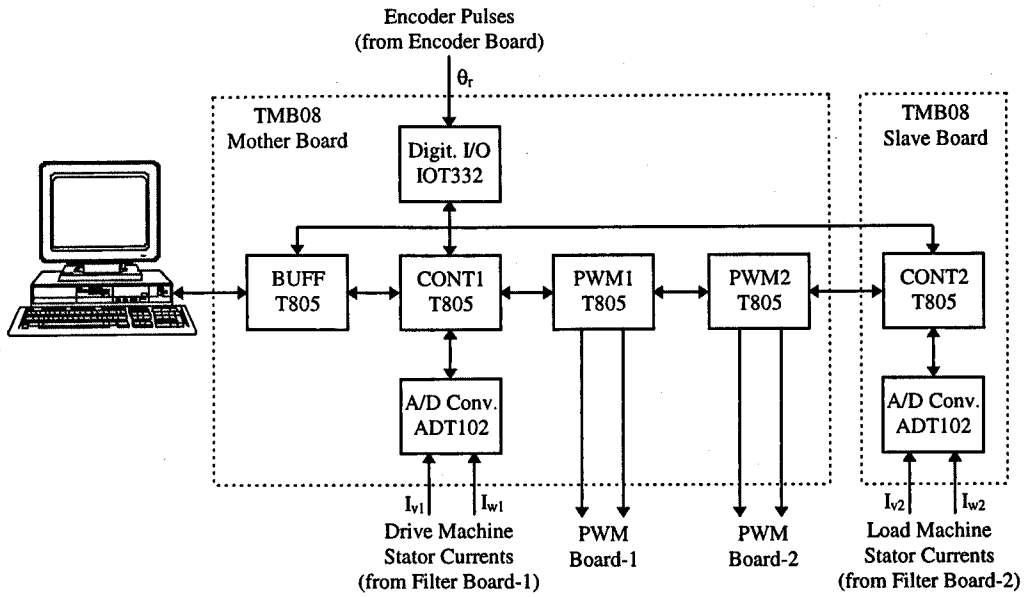


Figure 2.4 Block diagram of the transputer network

The transputers are programmed in a high level language called OCCAM which is specially developed to allow parallel programming. A program is written for each transputer and each program is separately compiled, linked and then loaded into the transputers on the network. Since the transputers have a parallel processing capability, when one process is waiting for a communication with another transputer or a peripheral, the second process is executed. This property has been used in the experimental implementation to establish the asynchronous communication between the PC and the transputer network.

The processes allocated to the transputers implementing different tasks, are synchronised by a communication protocol. No processes is allowed to take longer than the *basic sample period* (T_{bs}) which is actually the current loop sampling period for the vector control implementations. It should be remembered that the main objectives of this project are to develop a dynamometer control strategy for the emulation of mechanical loads and to use the emulated loads for the experimental investigation of the robust control methods such as fuzzy and sliding mode control. Therefore, in order to avoid any unnecessary burden in the software development, a value of $\approx 500\mu s$ is found suitable for the basic sample period T_{bs} for this project. On the other hand, T_{bs} must be an integer multiple of the PWM switching period [49]. A value of 6kHz switching frequency is considered adequate for the purpose of this research. Hence, T_{bs} has been set to $504\mu s$ which is the integer multiple of the PWM switching period ($168\mu s$). A value

of ≈ 504μs is also felt to be a maximum time considering the fast current loop bandwidths required in a vector drive. Note that every 504μs, the transputer labelled PWM1 communicates with CONT1 and PWM2 in order to establish the synchronisation between the main processes running on the transputers. This will be discussed later in this section in more detail.

The main tasks assigned to the transputers shown in Fig.2.4 are as follows :

(i) BUFF Transputer

The transputer labelled BUFF communicates with the PC and the rest of the network. It basically provides the necessary buffering for data capture facilities, decodes the user command (e.g., set point, on-line change of parameters and monitored variables) and transfers the command to the transputers labelled CONT1 and CONT2. The BUFF transputer also receives the electrical torque reference value of the drive machine (T_e) and the rotor angle (θ_r) from the transputer CONT1 and sends them to the CONT2 transputer in which the load emulation strategy is implemented. This is because they are required in the load machine control strategy which will be explained in Chapter 3.

There are two main processes running in parallel on the transputer BUFF. A high priority process carries out the synchronous communication (every 504μs) with the transputers CONT1 and CONT2. When this process is idle, waiting for communication with CONT1 and CONT2, a second lower priority process communicates with the host PC. Both processes have access to a common block of memory which is filled up with data by the high priority process while the other reads this data and transfers it to the host. In this way the transputer network can write to the buffer synchronously every 504μs and the host can read from this memory asynchronously without disturbing the operation of the transputer network.

(ii) CONT1 Transputer

Vector control of the drive machine, reading the digital data from the encoder interface board, measurement of the drive machine line currents (I_{v1}, I_{w1}) and the speed control strategy under evaluation such as fuzzy and sliding mode control techniques are implemented on the transputer labelled CONT1. The rotor angle is calculated using the digital data received from the encoder interface board though the I/O tram (IOT332). This angle is used to provide speed feedback for the vector controller and it is also transferred to the CONT2 transputer via the BUFF transputer.

The A/D conversion of two line currents of the drive machine are carried out by the ADT102 tram. The vector orientation algorithm and the current control loops are executed once every 504μs. Since this project concerns the area of speed control, a high speed sampling frequency is considered in order to allow a high speed control bandwidth [5]. However, the inner current control loops and their delay effects should be taken into consideration in the selection of the speed sampling period. Therefore, a value of 2.5ms sampling time is found adequate for the speed control loops in this project. The electrical torque reference of the drive machine, which is directly proportional to the output of the speed controller, is also transferred to the CONT2 transputer for use in the load emulation strategy. Once the vector control calculations are ready, the CONT1 transputer waits for the synchronising pulse from the PWM1 transputer to send the new calculated voltage vector in terms of v_d, v_q and θ_e to the PWM1 transputer.

(iii) PWM1 Transputer

The PWM1 transputer generates the switching pattern required by the PWM Generation Board which drives the power MOSFET inverter via the Inverter Interface Board (see Fig.2.2). As stated before, a switching frequency of 6kHz is considered adequate for the purpose of this research. In order to obtain a switching frequency of 6kHz (corresponding to a switching period of 168μs) the same voltage reference is used for three consecutive switching cycles [49]. The voltage reference, at the desired electrical frequency, consists of two quadrature voltages v_d and v_q and the flux-angle, θ_e. The transputer calculates the timing signals using regular symmetric PWM. However, due to the PWM Generation Board which will be discussed in the following section, two switching patterns must be calculated for each switching period. The PWM1 transputer therefore calculates the adequate switching patterns every 84μs and sends them via two transputer links to the PWM Generation Board. Once every six calculations which corresponds to 504μs, the PWM1 transputer is updated by the CONT1 transputer and this communication is used to synchronise the processes on the whole network. Note that the PWM1 transputer sends a synchronisation pulse to the PWM2 transputer to synchronise the processes on the PWM2 and CONT2 transputers as well. The synchronisation of the whole network is required in order to avoid any dead-lock due to communications between the transputers.

(iv) PWM2 Transputer

The transputer labelled PWM2 generates the switching pattern for the load machine. It implements exactly same task that the PWM1 transputer does. The only difference is that it receives the voltage reference from the CONT2 transputer to calculate the PWM timing values for the load machine.

(v) CONT2 Transputer

Vector control of the load machine, measurement of its currents (I_{v2}, I_{w2}) and the load emulation strategy are implemented in this transputer. An A/D converter tram (ADT102) is used again to sample the filtered currents. The CONT2 transputer receives the rotor angle and the electrical torque reference of the drive machine from the BUFF transputer which transfers them from the CONT1 transputer. These variables are used in the load emulation strategy. Similar to the control of drive machine, once the vector control calculations are ready, the CONT2 transputer waits for the communication with the PWM2 transputer to send the new calculated voltage vector in terms of v_d, v_q and θ_e. This communication occurs every 504μs synchronous with the rest of the processes running on the other transputers. Note that the software structures in the network are designed in such a way that any desired variable in CONT1 and CONT2 can be recorded and monitored.

2.4 The Interface Circuits

The transputer network communicates with the outside world by using some interface circuits as shown in Fig.2.2. Each transputer has four serial bi-directional links which can be connected to another transputer or to a link adapter. The link adapters can convert the serial data from the link into parallel format suitable for use by the hardware. The PWM transputers generate switching times which should be converted to appropriate PWM patterns by some hardware interfaces. The analogue signals should be measured and low pass filtered against noise and aliasing before the A/D conversion stage. An encoder interface circuit is also required to convert the encoder pulses to an appropriate information for the calculation of the rotor shaft angle and the speed. In addition, two inverter interface circuits are needed to use the generated PWM signals instead of the internal PWM signals of the commercial inverters. The circuit diagrams of all the interface boards are given in Appendix-B.

Most of the interface circuits used in this study have been employed in some previous projects [48-50] with some small differences and they have been described extensively. However, they are briefly discussed below as well:

(i) PWM Generation (Counter) Circuits

The PMW transputers generate the switching times of each inverter leg. However, these switching times need to be converted to the appropriate PWM patterns before they can be sent to the Inverter Interface Boards. Therefore, two PWM Generation (Counter) Circuits shown in Fig.B.1 are built around a 8254 counter/timer. The 8254 provides three separate counters which is very convenient for generating the three phase PWM patterns on just one chip. The input to the timer is a 8-bit parallel bus, since the transputer links use serial communication, a CO11 link adapter has to be used in order to convert the serial data into parallel data. Two of these adapters have to be used, one to provide the address to access the different counters while the other, to carry the digital count for the appropriate timing signal. The 8254 is used in monostable mode, i.e. the output of each counter is normally high. When it is triggered, the output will become low for the duration of the count value. Three different counting values, one for each phase, are sent by the PWM transputer every 84μs. Normally the three counters will be triggered at the same time. Extra circuitry is needed in order to provide high to low pulses, as well as the low to high pulses required by the gate drivers. The extra circuitry consists of three XOR gates and three flip-flops, which work as programmable inverters. Typical waveforms for one phase are shown in Fig.2.5, where t_1, t_2 and t_3 correspond to the timing values calculated by the PWM transputer. The clock frequency used for the 8254 is 5MHz. The 5MHz oscillator is also used to provide an appropriate clock signal for the link adapters and the *dead-lock protection circuit* which will be explained next.

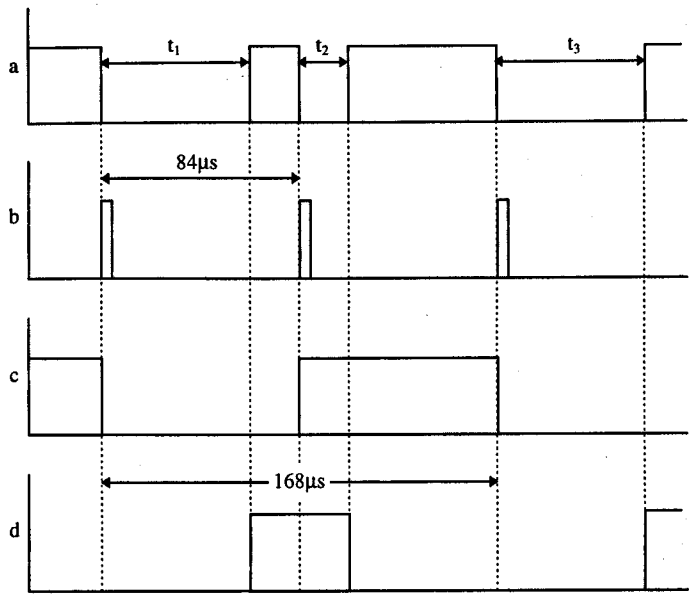


Figure 2.5 Typical waveforms of the PWM generation circuits
(a) 8254 counter output (b) Trigger pulses
(c) Inverting signals at XOR gate input (d) PWM output

(ii) Dead-lock Protection Circuits

These circuits are built on the PWM Generation Boards (thus, not shown in Fig.2.2) additionally to protect the inverters when a dead-lock occurs. If any one of the transputers fails to send or receive a message to/from a channel (a link), then a dead-lock occurs. This failure can be caused either by a hardware problem or a software error. Hardware faults usually arise from electromagnetic interference on the transputer links.

Deadlock leads to immediate loss of the PWM signal. When this happens, the power MOSFETs will remain in the last switching pattern received before dead-lock. This can cause the full DC link voltage to appear on the machine terminals which will cause a large increase in current due to the relatively small stator resistance. Although an overcurrent fault should turn all the MOSFETs off with no equipment damage, a dead-lock protection has been designed to ensure a higher degree of safety.

This protection circuit (Fig.B.2) uses the transputer generated pulses that enables the timer on the PWM board together with an eight bit-binary counter. When the transputer network is operating under normal conditions, the timer counter enable signal will be sent to the PWM

board every 84μs. The binary counter (TTL 54590) uses a 2.5MHz external clock which is obtained by dividing the counter circuit clock by two. During normal operation the enable signal will reset the counter before it reaches its final count (256 pulses equivalent to 102.4μs). If the reset signal is not received from the transputer, indicating that the synchronism has been lost due to a dead-lock, the counter will reach its final count and the RCO (see Fig.B.2) will generate a pulse. This pulse is latched with a flip-flop and a fault signal is generated with a maximum delay of 18.4μs after the synchronism is lost.

(iii) Inverter Interface Circuits

Two inverter interface circuits are built in order to use the PWM signals generated by the PWM Counter Boards. These circuits basically cancel the internal PWM signals in the inverters and connect the ones generated by the counter boards to the inverter gate drivers. Also, they provide an extra manual control to stop the inverters when it is required. The circuit diagram of the inverter interface board is illustrated in Fig.B.3. Note that these interface circuits are between the logic and the power boards of the inverters. Thus, in order to avoid any inconvenience in the operation of the inverters, the interface circuits are designed to allow all the signals to carry on between the logic and the power boards except the internal PWM signals.

(iv) Current Measurement and Filter Circuits

The circuit diagram of the current measurement and the filter boards are shown in Fig.B.4 and B.5 respectively. Hall effect current transducers are used in the current measurement circuits. The analogue signals from the current measurement boards are filtered to avoid aliasing and noise problems in the A/D conversion stage. These signals, which are used for vector control, are sampled at 2kHz. The low pass filters used are second order Butterworth filters (see Fig.B.5) with a cut off frequency of 600Hz which provides sufficient attenuation of frequencies above 1kHz.

(v) Encoder Interface Circuit

The motor shaft position and the speed are derived using an encoder mounted directly on the common shaft of the induction motors. This encoder provides six channels (three complementary pair lines) A, B and C with 5V CMOS differential line driver outputs. A and B

are quadrature signals which provide information on the speed and direction of the shaft rotation. Z acts as a zero position reference.

The interface circuit shown in Fig.B.6 employs the HCTL-2016 decoder chip which transforms the three signals A, B and C to absolute shaft position measurement. The encoder signals are fed to differential line receivers (DS88C20) which increase noise immunity. Twisting the encoder cable around a ferrite core is also an effective technique to reduce the high frequency noise. The overall system resolution is 10000 pulses per revolution corresponding to 0.036° per bit. The 16 bit output data from the decoder is fed as two bytes directly to a parallel input/output TRAM (IOT332). In order to read the position every sample period (i.e. 504μs), the CONT1 transputer sends two consecutive enable signals (OE) together with the appropriate select signal (SEL). The latter signal selects which byte (upper or lower) will be read first.

2.5 The Induction Machines and the Inverters

The AC machines used in the experimental system are Brook Hansen 3-phase induction machines rated at 0.55kW. The parameters of the machines are given in Table 2.1, where the nominal mechanical parameters J_n and B_n are the total rig inertia (including coupling) and the total average viscous friction of both machines.

Table 2.1 The parameters of the induction machines (per phase)

Frame reference : D80A4R	Stator resistance (R_s) = 18.5 ohm
Rated speed : 1400 rpm	Stator self inductance (L_s) = 0.88 H
Number of poles : 4	Rotor resistance (R_R) = 12 ohm
Number of stator slots : 36	Rotor self inductance (L_R) = 0.883 H
Rated i_{sd} = 0.864 A	Mutual inductance (L_0) = 0.843 H
Rated i_{sq} = 1.085 A	Nominal rig inertia (J_n) = $3.5*10^{-3}$ kgm^2
Torque at rated i_{sq} = 4.534 Nm	Nominal viscous friction (B_n) = $7*10^{-4}$Nms

The AC drives are both identical and they are HEENAN (Model HS110) power MOSFET voltage fed inverters rated at 1.1kW. The input supply is single phase and the output is three phase with maximum 5A current (continuous) limit.

Since the DC links are connected together, the mains power only supplies the system losses. The only energy which can not be circulated between the two inverter drives is the kinetic energy of the rig: this energy will be dissipated in the rig (mostly in the motor resistances and frictions) or else dumped in the DC link capacitors. Measurements show that the DC link voltage increase is about 5-6V for a full speed to zero transient. This is negligible and no action was taken to limit this increase.

2.6 Conclusions

In this chapter, the main hardware and software components of the experimental system have been described. The experimental system enables the user to control both induction machines separately and thus provides a practical test system for research into dynamometer and motor control strategies. Using the experimental system, a new dynamometer control strategy will be proposed in Chapter 3. The dynamometer will be used to provide linear and non-linear mechanical loads for the experimental validation of the robust speed control method developed in Chapter 5. However, any linear, non-linear, robust or adaptive control method can be experimentally tested using the rig described in this chapter.

A transputer network has been employed for the real time control of the experimental system. Although a transputer implementation is not suitable for a commercial product, it is very attractive for a research implementation. This is simply because the transputer implementation is extremely flexible and imposes almost no constraint in processing power. Another transputer can always be added to the network if more processing power is required. However, a high performance DSP may be used to control the overall experimental system.

Chapter 3

Emulation of Mechanical Load Models

3.1 Introduction

The use of torque-controlled load dynamometers is common in engine test-beds or in the testing of electrical machines [20-22]. In these applications, the engine or electrical machine is normally tested under steady state or slowly changing conditions. Recent research, aimed at emulating loads having faster dynamics [23-27], has resulted in simulated load emulation under *open-loop* conditions i.e. the emulated load is not part of a closed loop speed or position control system. Dynamic load emulation under closed-loop conditions is desirable for evaluating motor drive controllers. Researchers reporting motor control methods generally validate results using either a load machine connected to a resistor bank (emulating viscous friction) or a torque-controlled load machine emulating gravitational loads or general torque disturbances. However, adaptive and robust control schemes are attracting considerable attention. To verify the effectiveness of these, it is desirable to provide a dynamometer load in which mechanical parameters (such as inertia and friction) can either be pre-programmed or else vary with speed or position (e.g. winding applications, robot arms). In such cases, it is very desirable that the emulation preserves the model mechanical dynamics. This chapter addresses this problem.

In addition to machine/engine testing, another application of mechanical load emulation (either in open or closed loop) is to provide off-site testing of converter drives driving real industrial applications. Many of these provide challenges for the application or commissioning engineer. Examples include high-stiction loads (e.g. reciprocating pumps, escalators), period impact loads (large washing machines, compressors), the catching of spinning loads (after power interrupt) and many underhauling/overhauling applications. If the parameters of such loads are even only approximately known, the ability to evaluate and test such applications off-site would be advantageous.

As described in Chapter 2, the experimental system used in this project consists of two vector controlled induction machines on a common shaft which are controlled using a microprocessor system. The drive machine and its inverter provide the target system for research into motion control strategies. The main objective of this chapter is to develop a control strategy for the load machine (dynamometer) so that the mechanical rig dynamics, defined as the speed response to a given drive torque, is equivalent to that of a desired linear or non-linear mechanical load dynamics. In this way, the emulation preserves the physical causality of a mechanical system in which the motion variables are the output responses to a driving force or torque. In this chapter, we concentrate on the control of the load machine to achieve this objective.

Previous dynamic emulation research [23-28] is based on the principle of *inverse mechanical dynamics* in which the shaft speed is measured and used to derive the desired torque for the dynamometer. In Section 3.2, we analyse and discuss this principle, showing that discretization effects can severely affect the emulation. In [24],[25] a model-reference approach is presented in which it is implied that the shaft speed or position could be used as a tracking variable and so avoid the inverse dynamics. However, this is not clear since the authors present results in which the shaft torque is the tracked variable; neither is the preservation of the mechanical dynamics (pole-zero structure) addressed in [24],[25]. In [28], an integrator back-stepping design technique is presented which claims to emulate a dynamic load under closed loop conditions. However, the desired torque trajectory is still derived from an inverse mechanical model and only simulation results are given. In fact, all the researchers in [23-28] present only simulation results. This is also discussed in Section 3.2. In this chapter, the results are experimental, simulation results being provided only for comparison.

3.2 Dynamic Emulation of Mechanical Loads and Conventional Inverse Model Approach

Consider a basic first order mechanical dynamics given by

$$T_e = J\frac{d\omega}{dt} + B\omega \tag{3.1}$$

where T_e is the electrical torque, J is the moment of inertia, B is the viscous-friction coefficient and ω is the mechanical angular speed. This equation describes the dynamic behaviour of a mechanical drive load with constant inertia and friction during the transients and steady state condition. The Open Loop Transfer Function (OLTF) becomes

$$\frac{\omega(s)}{T_e(s)} = \frac{1}{Js + B} \tag{3.2}$$

In load emulation, the objective is to produce an OLTF which may be any desired linear or non-linear relation between input T_e and output ω. It is possibly best to start with the simplest load emulation which can be expressed by

$$\frac{\omega(s)}{T_e(s)} = \frac{1}{J_{em}s + B_{em}} \tag{3.3}$$

where J_{em} and B_{em} are the emulated inertia and friction respectively. In other words, J_{em} and B_{em} will be defined or given by the user. Now, the aim is to produce a load machine control structure so that the relation between the shaft speed ω and the electrical driving machine (motor) torque T_e is given by equation (3.3).

In order to emulate the mechanical drive load of equation (3.3), perhaps the simplest method is to use the *inverse model* as shown in Fig.3.1.

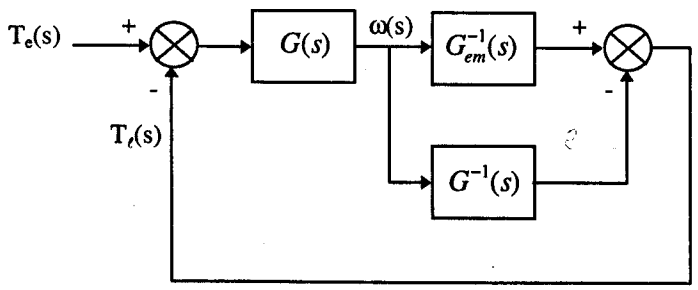


Figure 3.1 The inverse model approach for load emulation

In Fig.3.1, $G(s) = 1/(Js+B)$ is the total dynamics of the drive and load machine (including connecting shaft), $G_{em}^{-1}(s) = J_{em}s + B_{em}$ is the inverse of the emulated load transfer function, T_ℓ is the electrical load machine (dynamometer) torque, T_e is the electrical drive machine (motor) torque. The speed is measured and the inverse dynamics $G_{em}^{-1}(s)$ yields T_ℓ after compensation for the drive and load machine dynamics. Then it is easily shown that

$$\frac{\omega(s)}{T_e(s)} = G_{em}(s) = \frac{1}{J_{em}s + B_{em}} \tag{3.4}$$

as required. However, in practice, the inverse dynamics will be implemented on a μP and sampling effects need to be considered. Using a backward difference approximation [5] :

$$\dot{\omega}(k) = \frac{\omega(k) - \omega(k-1)}{T_s} \qquad \text{and} \qquad \dot{\omega}(z) = \frac{(z-1)}{T_s z}\omega(z) \tag{3.5}$$

where T_s is the sampling time, yields the sampled-data system of Fig.3.2.

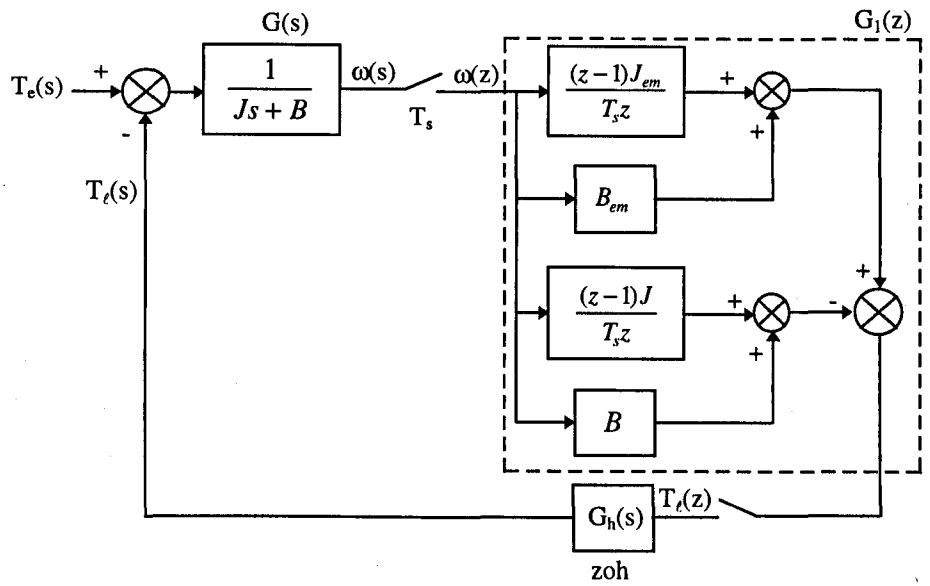


Figure 3.2 The sampled-data system of the inverse model

From Fig.3.2, where $G_h(s) = (1 - e^{-T_s s})/s$ (zero-order-hold), ω(z) can be derived as

$$\omega(z) = \frac{T_e G(z)}{1 + G_h G(z) G_1(z)} \tag{3.6}$$

Defining $\Delta J = J_{em} - J$, $\Delta B = B_{em} - B$ and $A = \exp(-BT_s / J)$, then $G_1(z)$ and $G_hG(z)$ of (3.6) can be expressed as

$$G_1(z) = \frac{(\Delta J + T_s \Delta B)z - \Delta J}{T_s z} \quad \text{and} \quad G_h G(z) = Z\{G_h(s)G(s)\} = \frac{(1-A)}{B(z-A)}.$$

$G_hG(z)$ is the zero order hold equivalent [5] or the step invariance discretization [52] of G(s). Usually, G_h is dropped and it is denoted as G(z) for simplicity of notation. It should be noted that numerator is $T_eG(z)$, not $T_e(z)G(z)$, so that the zeros of (3.6) depend on T_e. The characteristic equation $1 + G_h G(z) G_1(z) = 0$ of (3.6) can be derived as:

$$z^2 + \beta_1 z + \beta_2 = 0 \tag{3.7}$$

where $\beta_1 = \frac{(\Delta J + T_s \Delta B)(1-A)}{BT_s} - A$ and $\beta_2 = -\frac{\Delta J(1-A)}{BT_s}$.

In general, B will be small so that $A \cong 1 - (BT_s / J)$ and β_1, β_2 become

$$\beta_1 = \frac{J_{em} - 2J + B_{em}T_s}{J} \quad \text{and} \quad \beta_2 = -\frac{\Delta J}{J}.$$

If B_{em} is zero or small (corresponding to the emulation of an inertial load) then the roots are $z_1 = 1$ and $z_2 = -\Delta J/J$ which means the system is unstable if $\Delta J/J > 1$ or $J_{em} > 2*J$. Alternatively, if $\Delta J = 0$ and B_{em} is not zero or small then the roots can be shown to be $z_1 = 0$ and $z_2 = 1 - (B_{em}T_s/J)$. This means the system is unstable when $(B_{em}T_s/J) > 2$. However, the system can be stabilised using a digital filter of the form z(1-a)/(z-a) such that the second order system exhibits a dominant root close to $\exp(-T_sB_{em}/J_{em})$; this ensures a degree of matching for the dominant pole (i.e. the emulation has a dominant pole close to the actual mechanical pole). Such a filter pole can be provided by the filter normally included to smooth the noise on T_e since differentiating the shaft speed is noisy. Near-matching of the dominant pole may yield acceptable *open-loop* emulation providing that unstable or ringing poles [5] are kept in check. However, the overall pole-zero

structure of the desired mechanical load is not preserved and the emulation gives totally erroneous results if used in a closed loop control system.

The above analysis can also be applied to the particular torque control schemes of [23-27]. In these papers, the signal $T_\ell(s)$ at the output summer of Fig.3.1 becomes a shaft torque demand for a shaft torque control loop; the shaft torque being either measured or derived from the drive (or load) machine torque and the drive (or load) machine dynamics (it is noted however, that $G^{-1}(s)$ is not used as the compensating factor in these papers). The error is processed by a shaft torque controller to derive T_ℓ. The shaft torque controller can in fact fulfil the function of stabilising the system in a similar manner to the filter above. In the algorithms of [23-27], the structures are not designed to achieve exact dynamic matching as represented by (3.4); rather, the emphasis is on retaining stability [28] and achieving an acceptable time response matching for open loop emulation. Again, the overall pole-zero structure of the desired mechanical load is violated and the emulation cannot be used in a closed loop control system.

It is noted that simulations of inverse-dynamic structures such as Fig.3.1 are often successful. This is because $G_{em}^{-1}(s)$ is usually made proper by adjacent elements or filtering. Numerical solution in packages such as SIMULINK/MATLAB then proceeds by predictor-corrector methods with small time steps to yield near continuous or "analogue" simulations. However, in practice, noise considerations prohibit the use of small time steps for the computation of the inverse dynamics and discretization effects lead to the problems outlined above. In conclusion, we feel that inverse dynamics (and the need to compute accelerations) should be avoided. Further, it may not always be possible to derive the inverse dynamics of some non-linear loads.

3.3 Proposed New Emulation Strategies

In this section, two new load emulation strategies are developed to overcome the problems mentioned in the previous section. In these methods, basically, the real shaft speed is forced to follow a model-reference speed (the desired shaft speed) which is obtained by applying the drive machine torque to the desired emulated load dynamics. The details are given in the following subsections.

3.3.1 Emulation using Model Reference Speed Tracking

In this approach, the load to be emulated is used to derive an ideal speed demand as shown in Fig.3.3. ω is the real shaft speed and compared with ω_{em} which is the ideal speed demand or the desired speed at which we wish to go if T_e is applied to the load being emulated. The error is fed through a controller $G_t(s)$ to derive the load torque T_ℓ for the load machine.

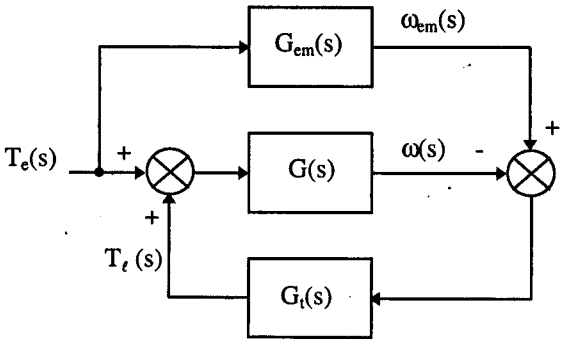


Figure 3.3 Load emulation using the model reference speed tracking approach

From Fig.3.3, the relation between the real speed $\omega(s)$ and the motor torque $T_e(s)$ can be derived as follows :

$$\frac{\omega(s)}{T_e(s)} = G_{em}(s)G_{comp}^{-1}(s) \tag{3.8}$$

where

$$G_{comp}(s) = \frac{(1/G(s)) + G_t(s)}{(1/G_{em}(s)) + G_t(s)} \tag{3.9}$$

If the term $G_{comp}(s)$ is added to the system in series as shown in Fig.3.4a, the required relation

$$\frac{\omega(s)}{T_e(s)} = G_{comp}(s)G_{em}(s)G_{comp}^{-1}(s) = G_{em}(s)$$

is obtained and thus the load's pole zero structure is retained.

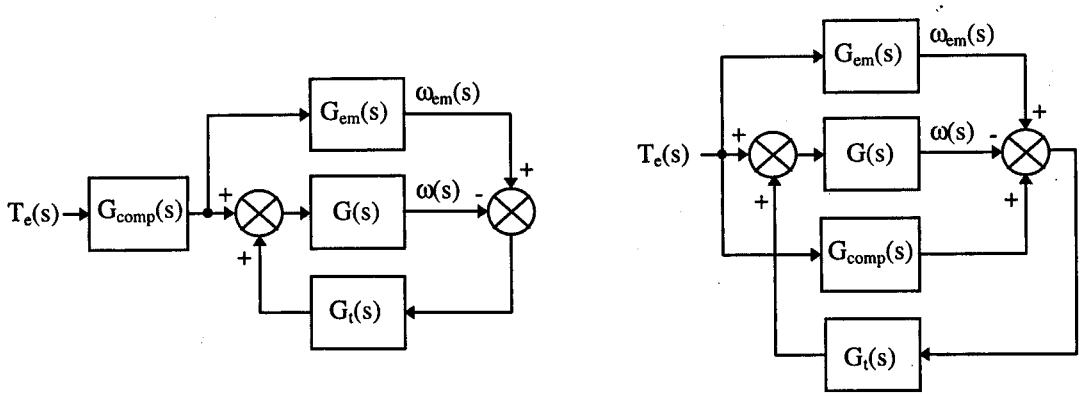


Figure 3.4 Compensated systems
(a) Series compensation
(b) Parallel compensation

Another way of the compensation is to employ a parallel compensation term as shown in Fig.3.4b from which the transfer function between ω(s) and $T_e(s)$ can be derived as

$$\frac{\omega(s)}{T_e(s)} = G_{em}(s) + \frac{G(s)\left(1+G_t(s)G_{em}(s)\right) - G_{em}(s)\left(1+G_t(s)G(s)\right)}{1+G_t(s)G(s)} + \frac{G_t(s)G(s)}{1+G_t(s)G(s)} G_{comp}(s) \qquad (3.10)$$

which implies that if $G_{comp}(s)$ is chosen as

$$G_{comp}(s) = \frac{G_{em}(s)\left(1+G_t(s)G(s)\right) - G(s)\left(1+G_t(s)G_{em}(s)\right)}{G_t(s)G(s)} \qquad (3.11)$$

then the require relation between ω(s) and $T_e(s)$ is obtained. However, both methods suffer from the fact that the compensation term $G_{comp}(s)$ includes emulated load parameters. This may be acceptable for linear emulated loads but it is undesirable for non-linear loads. The emulated load parameters should not be involved in the compensation term. A solution is given in the following section.

3.3.2 Emulation using Model Reference Speed Tracking with Feed-forward Torque Compensation

In Fig.3.3, a load machine speed controller is used to ensure that the shaft speed tracks an ideal speed ω_{em}. The control can be implemented by adding a feed-forward term obtained by using the inverse dynamics, not operating on the real speed as in Fig.3.1, but on the ideal emulated speed ω_{em}. This new structure is shown in Fig.3.5.

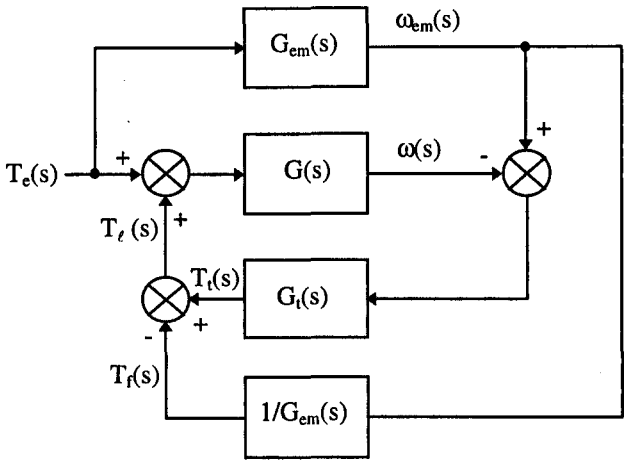


Figure 3.5 Load emulation using model reference speed tracking with feed-forward torque compensation

As seen in Fig.3.5, The feed-forward torque T_f becomes T_e via the inverse dynamics of the emulated load. Thus, the load machine cancels the motor torque T_e on the shaft and the net motor-generator torque is just enough to produce the required speed. Note that the path transmittance between T_f and T_e is unity so that the inverse load dynamics are not explicitly required. The system shown in Fig.3.5 reduces to the system shown in Fig.3.6. As seen in Fig.3.6, the driving machine torque, T_e, is inserted into a load model to derive ω_{em}. This speed is then forced onto the system via the load machine feedback loop as shown. Note that in practice, the negative of the driving machine torque is applied to the load machine. The load and driving machine torque are forced to cancel each other and thus T_t is the net residual torque on the shaft. From Fig.3.6, the transfer function becomes

$$\frac{\omega(s)}{T_e(s)} = G_{em}(s)\frac{G(s)G_t(s)}{1+G(s)G_t(s)} = G_{em}(s)G_{comp}^{-1}(s) \tag{3.12}$$

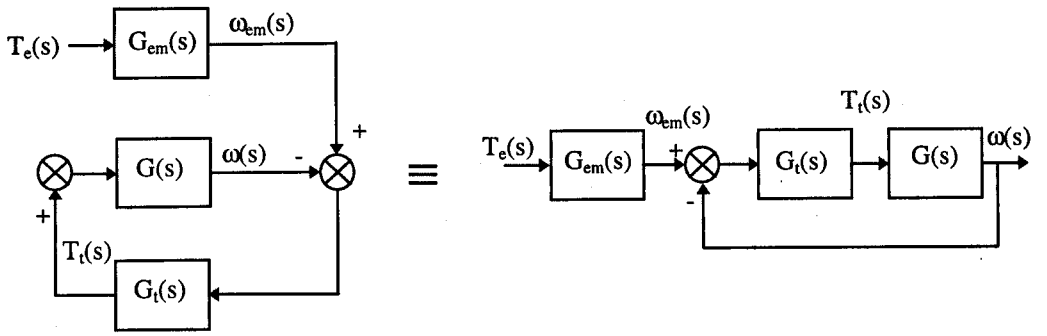


Figure 3.6 Reduced system

From (3.12), $G_{comp}(s)$ can be written as

$$G_{comp}(s) = \frac{1 + G(s)G_t(s)}{G(s)G_t(s)} \tag{3.13}$$

which should be added to the system in series to obtain the required relation between ω and T_e as indicated in Fig.3.7. The compensation term is no longer a function of $G_{em}(s)$ as required. It is assumed here that $G_{em}(s)$ must take into account the drive motor's rotor inertia; this allows the use of the electrical drive torque as the input forcing variable. The minimum inertia that can be emulated is thus equal to the drive motor's rotor inertia which accords with reality.

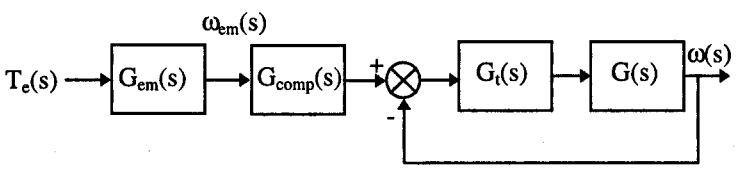


Figure 3.7 The compensated load emulation system satisfying the required relation between T_e and ω

Thus, as required, the resultant transfer function becomes

$$\frac{\omega(s)}{T_e(s)} = G_{em}(s)G_{comp}(s)G_{comp}^{-1}(s) = G_{em}(s) \tag{3.14}$$

However, since the speed tracking loop controller $G_t(s)$ is nominally a PI, $G_{comp}(s)$ becomes improper (i.e. the degree of numerator is higher than the degree of denominator). This can be

solved by considering the sampled-data representation for a microprocessor implementation which will be discussed in the following section.

3.4 Digital Implementation of the Emulation Strategy with Feed-forward Torque Compensation

The sampled-data system block diagram of the emulation strategy with feed-forward torque compensation is shown in Fig.3.8.

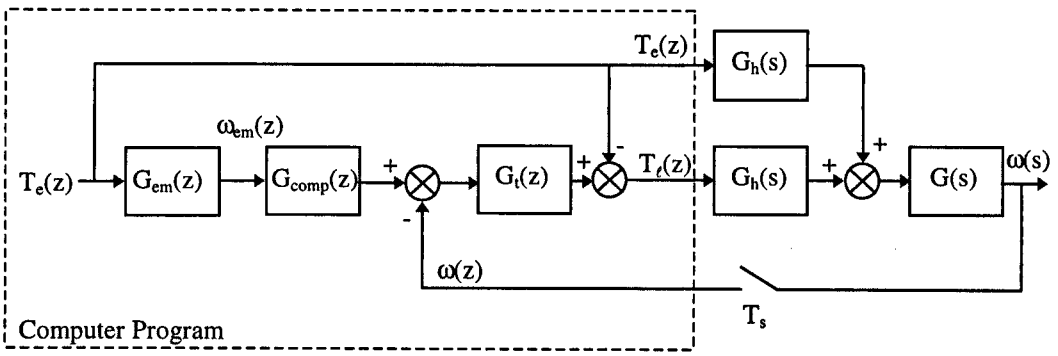


Figure 3.8 Open loop sampled-data system of the emulation strategy with feed-forward torque compensation

The reason why it is called 'open loop' system is that the transfer function ω/T_e is the open loop system to be controlled. Once the load emulation is achieved (which means the required open loop poles and zeros of the emulated load are obtained), then the loop can be closed for the speed control. The system in Fig.3.8 reduces to the system shown in Fig.3.9.

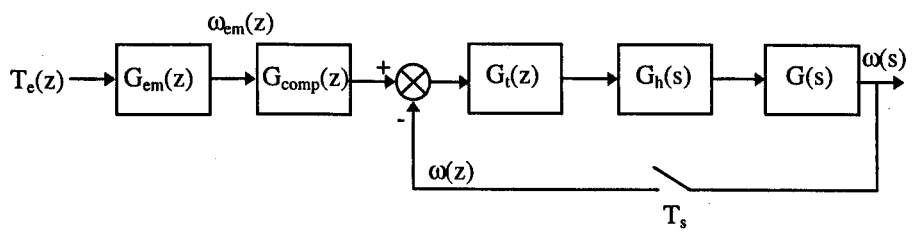


Figure 3.9 Reduced digital open loop load emulation system

Obviously, the function of $G_{comp}(z)$ is to cancel all parts of the open loop except the emulated load transfer function, so that the open loop z transfer function, (OLZTF) $\omega(z)/T_e(z)$, becomes equal to some discretized equivalent of $G_{em}(s)$. Therefore, $G_{comp}(z)$ should be

$$G_{comp}(z) = \frac{1 + G_t(z)G(z)}{G_t(z)G(z)} \tag{3.15}$$

where $G(z) = Z\{G_h(s)G(s)\}$ (the zero order hold equivalent of G(s)) and the speed tracking loop controller $G_t(z)$ is nominally a discrete PI controller. However, $G_{comp}(z)$ of equation (3.15) is improper (i.e. the degree of the numerator is higher than the degree of the denominator). It can be made proper by introducing a single delay:

$$G_{comp}(z) = \frac{1 + G_t(z)G(z)}{G_t(z)G(z)} \frac{1}{z} \tag{3.16}$$

This delay can itself be compensated by setting $G_{em}(z)$ as:

$$G_{em}(z) = z \cdot Z\{G_h(s)G_{em}(s)\} \tag{3.17}$$

For a linear $G_{em}(s)$, (3.17) corresponds to discretization by pole-zero matching in which the zero at infinity (s plane) is mapped onto z = 0 [51]. This is quite elegant since the z-transfer function $\omega(z)/T_e(z)$ reduces to:

$$\frac{\omega(z)}{T_e(z)} = G_{em}(z)G_{comp}(z)\frac{G_t(z)G(z)}{1 + G_t(z)G(z)} = Z\{G_h(s)G_{em}(s)\} \tag{3.18}$$

so that the emulation is equivalent to the step invariance discretization [52] (or the zero order hold equivalent) of $G_{em}(s)$. For a non-linear load, the advance operator z in (3.17) corresponds to using the latest value of T_e in the non-linear difference equations (see Section 3.6).

Since the desired open loop (ω/T_e) load emulation is obtained, the speed loop of the driving machine can be closed to control the speed of the emulated load. Fig.3.10 shows the sampled-data closed loop speed control of the emulated load. The system shown in Fig.3.10 actually corresponds to the experimental implementation of the motor-dynamometer system.

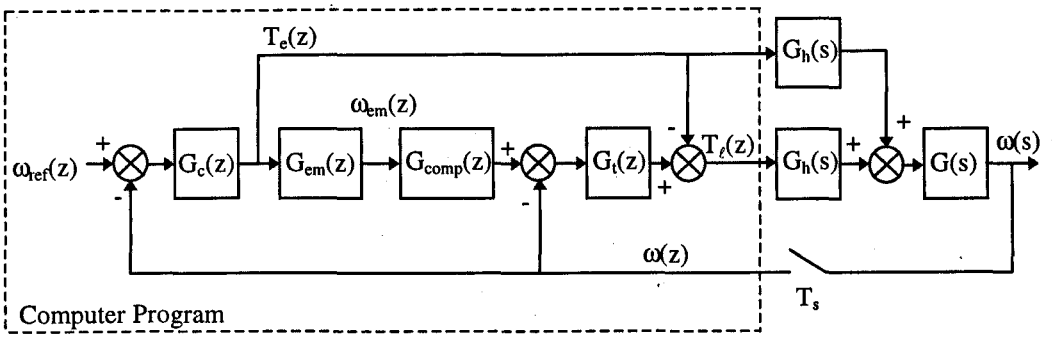


Figure 3.10 Sampled-data closed loop speed control of the emulated load

The controller $G_c(z)$ shown in Fig.3.10 is the speed controller of the emulated load. In other words, it is the controller under evaluation and robust, adaptive, linear or non-linear control methods will be implemented in this controller for the experimental investigation and validation. However, the main objective of this chapter is to develop a dynamometer control strategy, not the experimental investigation of the control methods. Therefore, $G_c(z)$ is chosen as a simple PI controller for the test of the developed dynamometer control strategy. In Chapter 5, a new robust speed controller design procedure will be described and it will be implemented in the controller $G_c(z)$ to evaluate its robustness against some linear and non-linear loads which are provided by the dynamometer described in this chapter.

If the speed tracking loop controller $G_t(z)$ is chosen as a PI given by

$$G_t(z) = \frac{K_t(z - A_t)}{z - 1}$$

then, $G_{comp}(z)$ (the delayed inverse of the speed tracking closed loop transfer function) given by (3.16) can be expressed as

$$G_{comp}(z) = \frac{1 + G_t(z)G(z)}{G_t(z)G(z)} \frac{1}{z} = \frac{a_2 z^2 + a_1 z + a_0}{b_2 z^2 + b_1 z + b_0} \tag{3.19}$$

where the coefficients are given in Table 3.1.

Table 3.1 The coefficients in the Equation (3.19)

$a_0 = \frac{JP}{K_t T_s} - A_t$	$b_0 = 0$
$a_1 = -(\frac{J(P+1)}{K_t T_s} - 1)$	$b_1 = -A_t$
$a_2 = \frac{J}{K_t T_s}$	$b_2 = 1$

In Table 3.1, $P = e^{-(T_s B/J)} \cong 1-(BT_s/J)$, T_s is the sampling time, J and B are the total inertia and friction of the drive and load machines including coupling.

$G_{em}(z)$ is the discretized emulated load dynamics and can be any linear or non-linear relation between ω and T_e (non-linear relations will be illustrated in Section 3.6). As stated before, in order to cancel the delay term in $G_{comp}(z)$, $G_{em}(z)$ should contain a unit advance term. If the emulated load is $G_{em}(s) = 1/(J_{em}s + B_{em})$ then

$$G_{em}(z) = z \cdot Z\{G_h(s)G_{em}(s)\} = \frac{C_{em}z}{z - P_{em}} \tag{3.20}$$

where C_{em} and P_{em} are constants. In fact (3.20) is the discretization of $G_{em}(s)$ using pole-zero matching [51] in which the zero at infinity (s plane) is mapped onto z = 0. This zero will cancel the $G_{comp}(z)$ delay pole to yield the system in Fig.3.11 which is equivalent to the system shown in Fig.3.10.

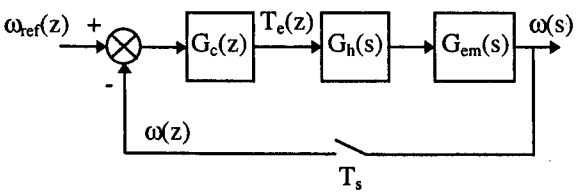


Figure 3.11 Reduced closed loop speed control system

Fig.3.11 actually shows the *ideal system* that we want to implement in the experimental rig. Thus, the system shown in Fig.3.11 will be simulated and the experimental results will be compared with the simulation results for the validation of the emulation strategy. The

simulation assumes that the vector controlled motor provides ideal torque control, and this is simplified to a unity gain within the simulation.

3.5 Experimental Validation of the Emulation Strategy

The experimental implementation of the emulation strategy on the vector controlled induction motor-dynamometer set is shown in Fig.3.12. The vector control is based on the IRFO which is briefly described in Appendix-A. PI speed controller-1 and controller-2 correspond to $G_c(z)$ and $G_t(z)$ respectively. K_T is the torque constant (e.g., $T_e = K_T \cdot i_{sq1}^*$) and the q-axis current demand of the load machine (i_{sq2}^*) corresponds to the load machine electrical torque demand T_ℓ (i.e., $T_\ell = K_T \cdot i_{sq2}^*$). The load to be emulated is implemented in the block G_{em} of Fig.3.12 and the compensation block G_{comp} is given by (3.19). Note that G_{em} may be a set of linear or non-linear difference equations. The design of the speed and current controllers shown in Fig.3.12 are discussed in the following subsection.

3.5.1 Design of the Controllers

The PI current controllers (d and q axis) of both machines are designed to yield a bandwidth of about 200Hz and kept constant. An attempt at a faster response results in undesirable closed loop response due to the ringing pole [71]. This limits the emulated mechanical dynamics to frequencies up to ≈50Hz. Note that high frequency vibrational modes (including backlash at modest to high speeds) will always remain beyond dynamometer emulation. Current loop delays can be included in the speed tracking loop in series with $G_t(z)$ and thus will appear in $G_{comp}(z)$. They are difficult to model accurately due to converter delays (although a model fit can be obtained from frequency response tests). However, if the frequencies of emulated mechanical dynamics are kept within reasonable limits mentioned above, the neglect of the current loops has little effect on the quality of emulation as shown in Section 3.5.2 and 3.6.

The PI speed controller-2 (the speed tracking loop controller) is designed to give a closed loop natural frequency for the speed tracking of 20rad/s and it is kept constant for all of the emulated loads in the thesis. The modest response of this loop is of no concern since the closed loop dynamics are compensated for by $G_{comp}(z)$ as explained. However, it is vital that PI speed controller-2 does not saturate otherwise the compensation will not be valid.

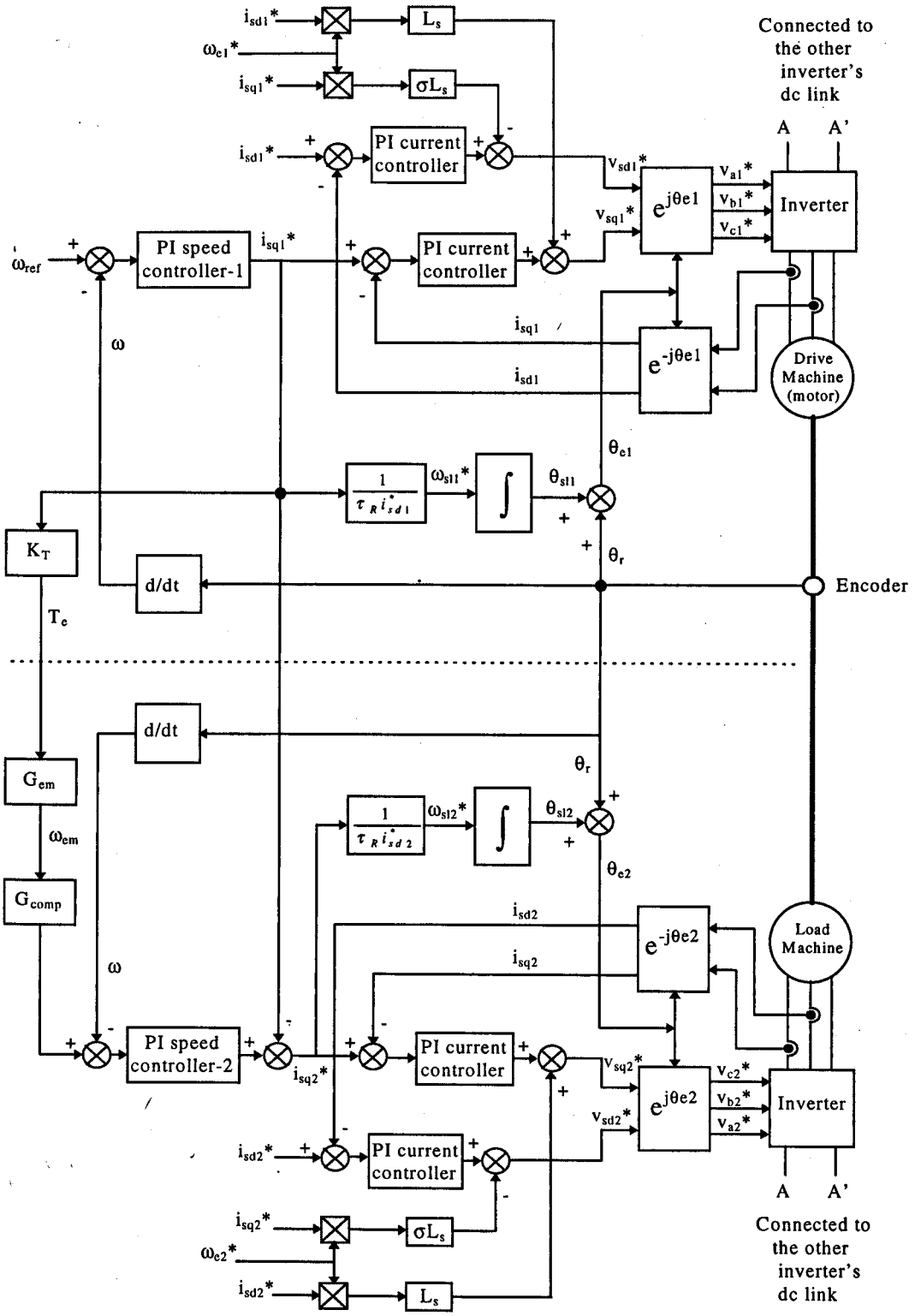


Figure 3.12 Experimental system

As mentioned in Section 3.4, $G_c(z)$ (corresponds to PI speed controller-1 of Fig.3.12) is the controller under evaluation. However, the main objective of this chapter is to validate the proposed emulation strategy, not to evaluate the controller. Hence, it is chosen as a PI controller and designed to give a closed loop natural frequency (ω_n) of 70rad/s and a damping ratio (ζ) of 0.7 for $J_{em} = J$, $B_{em} = B$ ($J = 0.0035 kgm^2$ and $B = 0.0007$Nms are the nominal inertia and friction of the rig, see Table 2.1).

3.5.2 Emulation of Linear Loads

In this section, the experimental results will be shown for the emulation of the linear load $G_{em}(s) = 1/(J_{em}s + B_{em})$ in a closed loop control system. The results will be compared with the simulation (using the SIMULINK / MATLAB package) of the closed loop system shown in Fig.3.11. In practice, the torque demand of the drive machine (i_{sq1}^*) should be limited to protect the inverter. Therefore, an anti-wind-up mechanism [1] is included in the PI speed controller-1. The anti-wind-up mechanism is also implemented in the controller of the simulation in order that an equivalence with the experimental system is obtained.

The experimental closed loop speed responses for three different loads of $J_{em} = J$, $J_{em} = 4J$, $J_{em} = 10J$ (B_{em} is kept constant and equal to B) and the corresponding simulation responses are shown in Fig.3.13a. $G_c(z)$ is the same in all three cases. The reference input is a step demand (100rad/s). Also shown is the *experimental torque measure* $k \cdot i_{sd1} \cdot i_{sq1}$ (where k is a constant equal to K_T/i_{sd1}^* and i_{sq1}, i_{sd1} are the *measured* stator currents of the drive machine) in comparison with the simulated T_e of Fig.3.11. Fig.3.13b shows the shaded area of Fig.3.13a in order to have a better view of the experimental and simulated speed responses. The experimental and simulated responses for $J_{em} = 0.5J$ and $J_{em} = J$ ($B_{em} = B$) are illustrated in Fig.3.13c. In order to avoid excessive overshoot and the effects of the fast current control loop poles in the experimental system, the gain of the speed controller $G_c(z)$ is halved for the case of $J_{em} = 0.5J$ (note that the plant gain is doubled when J_{em} is set to 0.5J).

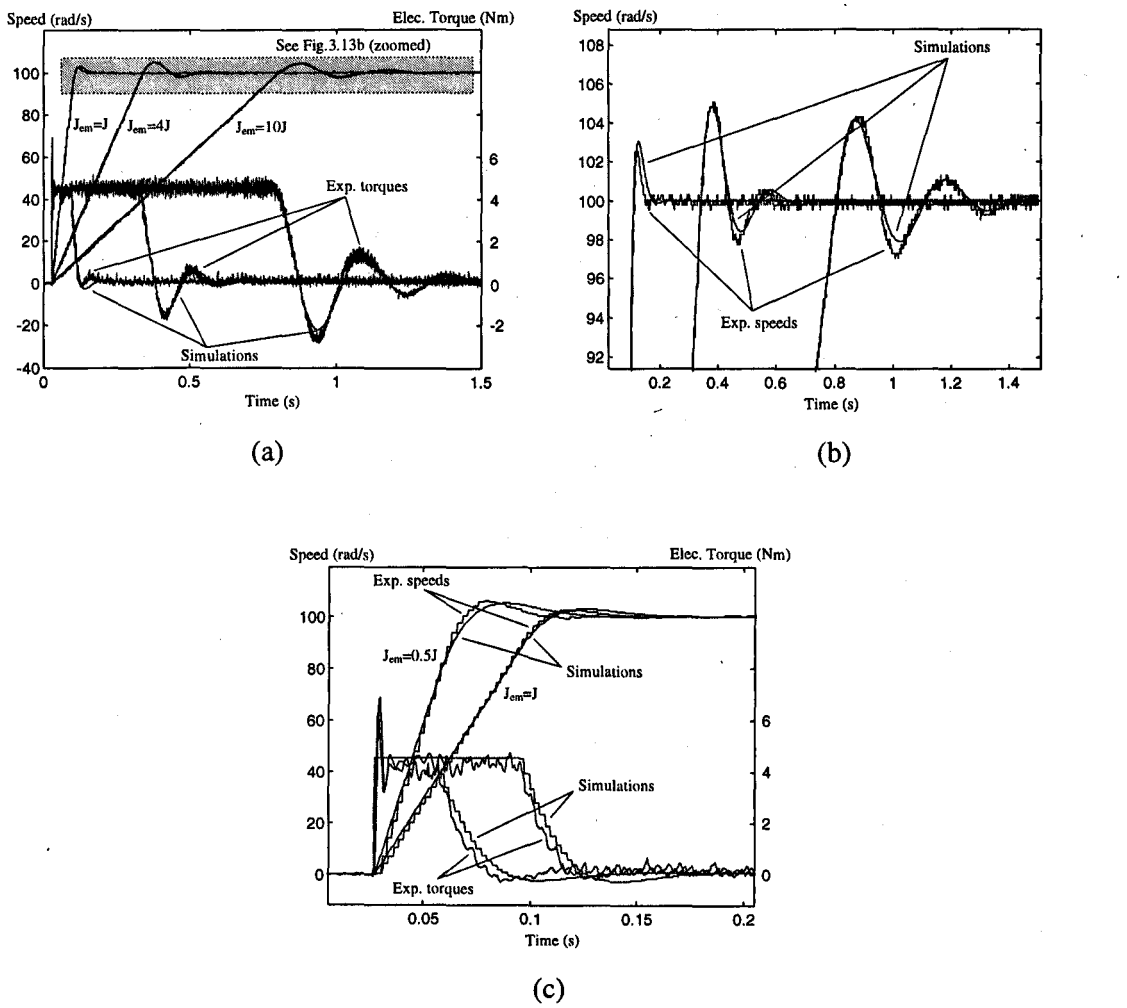


Figure 3.13

(a) The experimental and simulated speed and electrical torque responses for $J_{em} = J$, $J_{em} = 4J$, $J_{em} = 10J$ (B_{em} is kept constant and equal to B)

(b) Shaded area of Fig.3.13a

(c) Responses for $J_{em} = 0.5J$ and $J_{em} = J$

The comparison is felt to be very good and shows that the load dynamics are preserved in the emulation. The small differences between the experimental and simulated responses are thought to be due to the current control loops that are ignored in the simulation model. As mentioned before, it is difficult to model the exact current control dynamics for the closed loop. An approximate model could be included, however because of the good agreement between the experimental and simulated responses of Fig.3.13, it is felt that including the current loop delays in the simulation is unnecessary.

Fig.3.14a and b shows the experimental and simulated speed and electrical torque responses to an emulated load disturbance T_{ext} (50% of the rated torque) added before G_{em} in Fig.3.12 (J_{em} = 2J, B_{em} = 10B). Emulation of external torque disturbances present no problem.

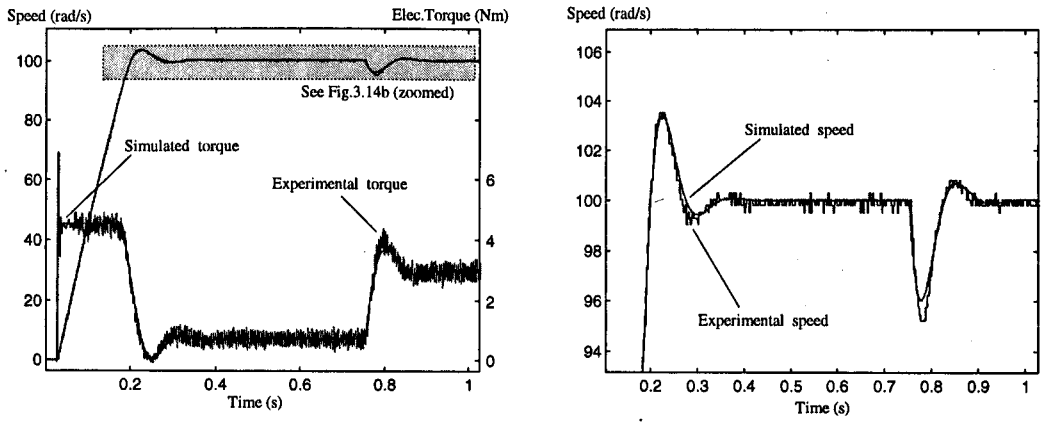


Figure 3.14

(a) The experimental and simulated speed and electrical torque responses for a step external load torque (50% of the rated torque)

(b) Shaded area of Fig.3.14a

3.6 Emulation of Non-linear Loads

In this section, experimental results will be shown for the emulation of some non-linear load models in comparison with the corresponding simulation results. For the non-linear loads, $G_{em}(z)$ becomes a set of non-linear difference equation with T_e as the input and ω as the output.

3.6.1 Aerodynamic Friction

The fans introduce a non-linear aerodynamic friction which can be included in the torque-speed equation as

$$T_e = J_{em}\frac{d\omega}{dt} + B_{em}\omega + B_a\omega^2 \tag{3.21}$$

where B_a is the aerodynamic friction (or windage) coefficient in Nms^2. Equation (3.21) can be rewritten in the form of

$$\dot{\omega} = \frac{T_e}{J_{em}} - \frac{B_{em}\omega + \mathrm{sgn}(\omega)B_a\omega^2}{J_{em}} \tag{3.22}$$

where sgn(ω) function is required to incorporate the direction of the speed; if ω becomes negative, $B_a\omega^2$ term will lose the sign and this term will cause a wrong effect. A discretization method is needed to implement (3.22) on a μP. One of the simplest methods is the backward difference method (Euler method) which can be expressed as

$$\dot{x} = f(x,t) \qquad \Rightarrow \qquad x_k = x_{k-1} + T_s f(x_{k-1}, t_{k-1}) \tag{3.23}$$

Thus, (3.22) can be discretized as

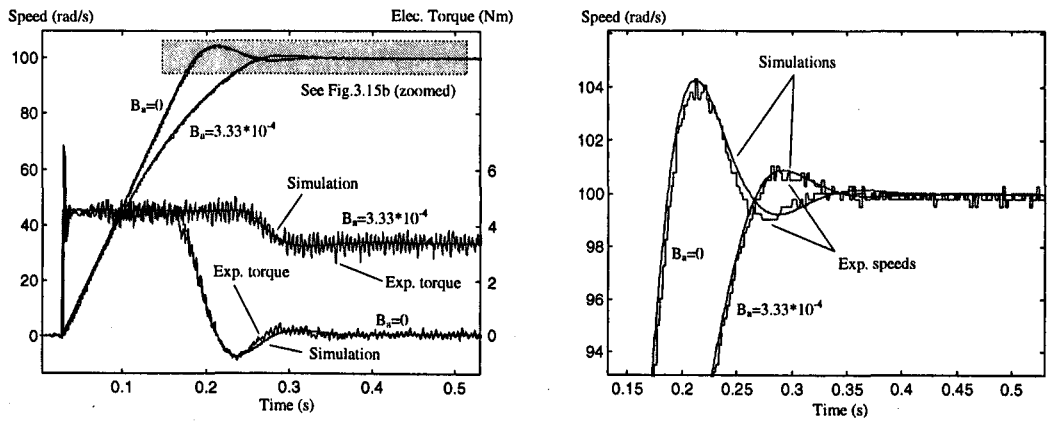
$$\frac{\omega(k) - \omega(k-1)}{T_s} = \frac{T_e(k)}{J_{em}} - \frac{B_{em}\omega(k-1) + \mathrm{sgn}(\omega(k-1))B_a\omega^2(k-1)}{J_{em}} \tag{3.24}$$

and ω(k) can be derived as

$$\omega(k) = (1 - \frac{T_s B_{em}}{J_{em}})\omega(k-1) + \frac{T_s}{J_{em}} T_e(k) - \frac{\mathrm{sgn}(\omega(k-1)) T_s B_a \omega^2(k-1)}{J_{em}} \qquad (3.25)$$

Note that in (3.25), instead of $T_e(k-1)$, $T_e(k)$ is used to introduce a unit advance term in order to cancel the delay term in $G_{comp}(z)$. In the experimental system, (3.25) is implemented in the G_{em} block of Fig.3.12. In the simulated system, the original non-linear differential equation (3.22) is implemented in the G_{em} block of Fig.3.11.

Fig.3.15a shows the experimental and simulated closed loop speed and electrical torque responses of the non-linear load model given by (3.21). The reference input is a step (100 rad/s) function. The figure shows the responses for $B_a = 0$ (corresponding to a linear load with J_{em} and B_{em} only) and $B_a = 3.33*10^{-4}$ (chosen so that the steady state value of the driving machine electrical torque, T_e, becomes 75% of the rated torque) in order to illustrate the effect of the aerodynamic friction. The emulated inertia and friction are chosen as $J_{em} = 2J$, $B_{em} = B$. Note that the PI speed controller-1 of Fig.3.12 (and $G_c(z)$ of Fig.3.11) is the same controller designed in Section 3.5.1 and kept constant for both values of B_a. Fig.3.15b shows the shaded area of Fig.3.15a to give a clearer view.



(a) (b)

Figure 3.15

(a) The experimental and simulated speed and electrical torque responses of the aerodynamic frictional load

(b) Shaded area of Fig.3.15a

3.6.2 Speed Dependent Inertial Load

As a further simplistic test of the performance of the load emulation, an extra inertial non-linearity is introduced in the torque speed equation as

$$T_e = J_t(\omega)\frac{d\omega}{dt} + B_{em}\omega \tag{3.26}$$

where

$$J_t(\omega) = J_{em} + K_j\omega^2 \tag{3.27}$$

so that the effective inertia is a function of the speed. Note that such a speed-dependent inertia may not be mechanically realistic. The inertial function is included here for assessment purposes; a realistic mechanical system involving a speed-dependent inertia is considered in Section 3.6.5. Equation (3.26) can be rewritten in the form of

$$\dot{\omega} = \frac{T_e}{J_{em} + K_j\omega^2} - \frac{B_{em}\omega}{J_{em} + K_j\omega^2} \tag{3.28}$$

which is discretized using the backward difference method (as shown in Section 3.6.1) to use in the experimental implementation. In the simulated system shown in Fig.3.11, (3.28) is directly implemented in the G_{em} block.

Fig.3.16a shows the experimental and simulated closed loop speed and electrical torque responses of the non-linear load model represented by (3.26). The reference speed is a step (100 rad/s) demand. In order to illustrate the effect of K_j, the speed and electrical torque responses are shown for $K_j = 0$ and $K_j = 2*10^{-6}$ (chosen so that the total effective inertia, $J_{em} + K_j\omega^2$, becomes approximately 8J when the speed reaches the steady state value of 100 rad/s). The emulated inertia, friction and the speed controller ($G_c(z)$ and The PI speed controller-1) are the same with the ones used in Section 3.6.1. Note that the speed controller is kept constant for both values of K_j. Fig.3.16b shows the shaded area of Fig.3.16a to have a better view of the experimental and simulated speeds.

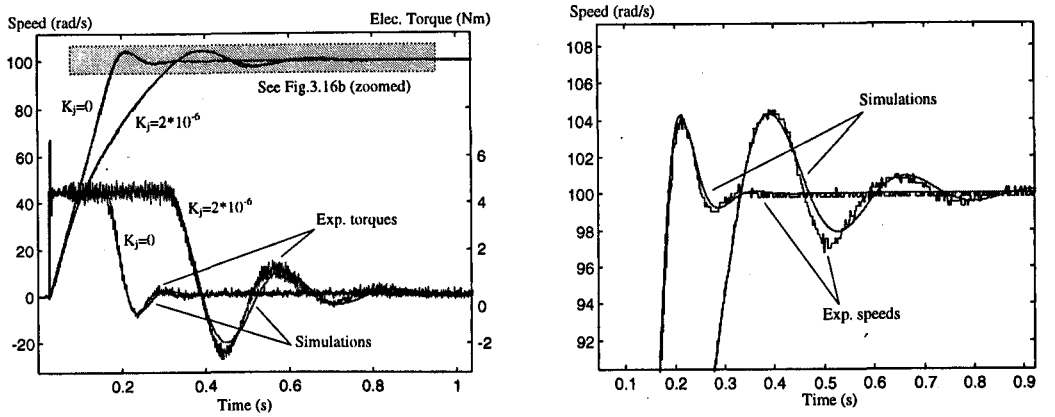


Figure 3.16

(a) The experimental and simulated speed and electrical torque responses of the speed dependent inertial load

(b) Shaded area of Fig.3.16a

3.6.3 Speed Dependent Inertial and Frictional Load

The third non-linear load characteristic to be emulated is

$$T_e = (J_{em} + J_p \sin \alpha\omega)\frac{d\omega}{dt} + (B_{em} + B_p \cos \beta\omega)\omega \qquad (3.29)$$

where J_p, B_p, α and β are constants. Note that the non-linearity introduced in (3.29) is for the assessment of the emulation strategy, they may not be mechanically realistic. For the simulated system shown in Fig.3.11, (3.29) is directly implemented in the G_{em} block and it is discretized using the backward differences for the experimental implementation. Fig.3.17a shows the closed loop experimental and simulated speed and electrical torque responses of the non-linear load given by (3.29). The reference speed is 100 rad/s and the parameters are $J_{em} = 4J$, $B_{em} = 10B$, $J_p = 3J$, $B_p = 5B$, $\alpha = \beta = 0.15$. Fig.3.17b is the shaded area of Fig.3.17a. The PI speed controller-1 of the experimental system (and $G_c(z)$ of the simulated system) is the same controller designed in Section 3.5.1

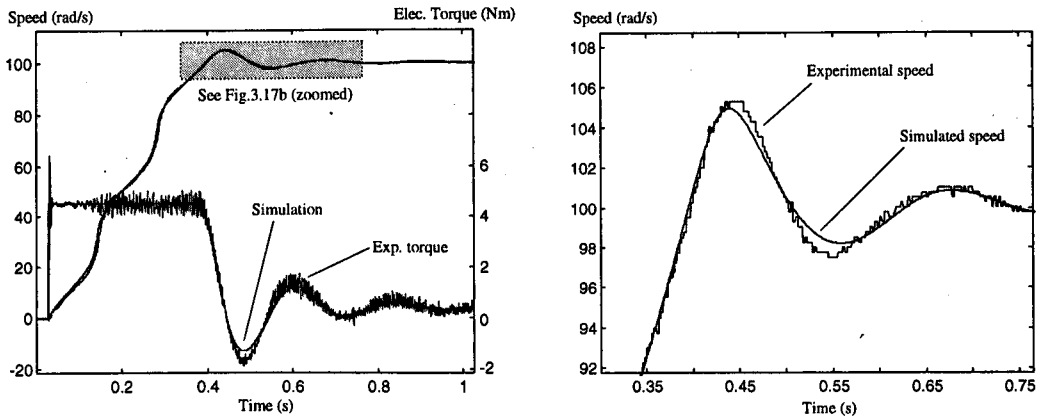


Figure 3.17

(a) The experimental and simulated speed and electrical torque responses of the speed dependent inertial and frictional load

(b) Shaded area of Fig.3.17a

3.6.4 Inertial and Frictional Load with Stiction

Stiction is a well known problem in the motion control systems. Mathematically, it is difficult to model an exact physical stiction because the stiction torque-speed characteristic tends towards a delta function at zero speed. However, an approximate model can be implemented for stiction emulation. Fig.3.18 shows a possible model function for combined viscous friction and stiction.

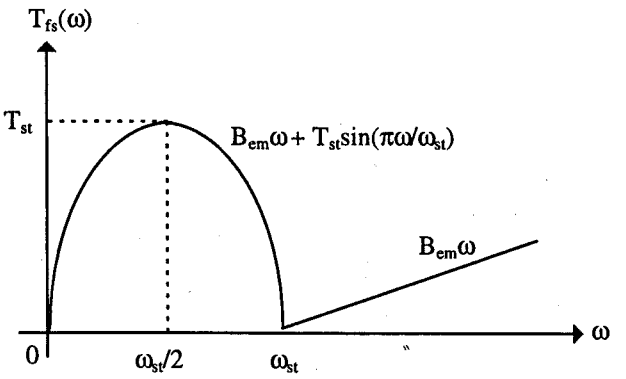


Figure 3.18 Torque-speed characteristic of an approximate stiction model

The graph is given only for positive speed; it is completely symmetrical for the negative speed with respect to the origin. The torque-speed equation can be written as

$$T_e = J_{em}\frac{d\omega}{dt} + T_{fs}(\omega) \tag{3.30}$$

where

$$T_{fs}(\omega) = \begin{cases} B_{em}\omega + T_{st}\sin(\frac{\pi}{\omega_{st}}\omega) & |\omega| \le \omega_{st} \\ B_{em}\omega & otherwise \end{cases} \tag{3.31}$$

which physically means that when the motor shaft starts to move, an extra torque (stiction torque) becomes active until the shaft speed reaches a certain value (ω_{st}). In other words, if the shaft speed is less than ω_{st}, there is a stiction torque additional to the friction torque. ω_{st} should be as small as possible to obtain a good stiction model. In the experimental implementation, the minimum value of ω_{st} is determined by the resolution of the shaft encoder which is 2.4rpm (0.25rad/s). Equation (3.30) can be discretized using backward differences to the difference equation:

$$\omega(k) = \begin{cases} (1-\frac{B_{em}T_s}{J_{em}})\omega(k-1) + \frac{T_s}{J_{em}}T_e(k) - \frac{T_{st}T_s}{J_{em}}\sin(\frac{\pi}{\omega_{st}}\omega(k-1)) & |\omega| \le \omega_{st} \\ (1-\frac{B_{em}T_s}{J_{em}})\omega(k-1) + \frac{T_s}{J_{em}}T_e(k) & |\omega| > \omega_{st} \end{cases} \tag{3.32}$$

The digital implementation will impose further restriction on T_{st} and ω_{st}. Numerical instability can occur for large slopes of T_{fs}-ω characteristic. The stability analysis of the stiction model is given below.

Stability analysis of the stiction model

In order to determine the stability limit, we should consider the maximum friction which occurs at the maximum slope of $T_{fs}(\omega)$ at $\omega = 0$ as seen in Fig.3.19.

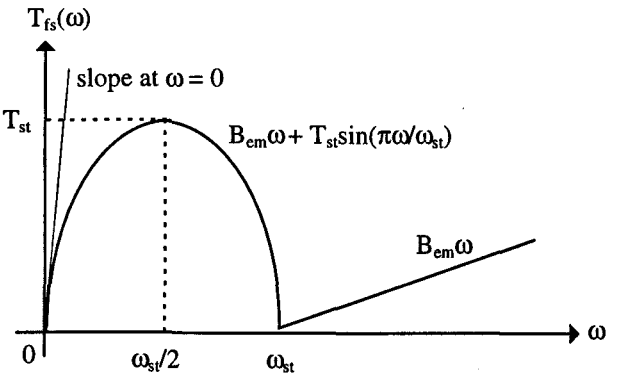


Figure 3.19 The maximum slope which occurs at $\omega = 0$

The maximum slope of $T_{fs}(\omega)$ at $\omega = 0$ is

$$\frac{dT_{fs}(\omega)}{d\omega}\Big|_{\omega=0} = B_{em} + \frac{\pi T_{st}}{\omega_{st}} \tag{3.33}$$

Thus we set $T_{fs}(\omega)$ as $(B_{em} + \frac{\pi T_{st}}{\omega_{st}})\omega$ to consider the worst case in terms of stability. For the stiction region $|\omega| \le \omega_{st}$, (3.32) is rewritten according to this consideration :

$$\omega(k) = (1-A)\omega(k-1) + \frac{T_s}{J_{em}} T_e(k) \tag{3.34}$$

where

$$A = \frac{T_s}{J_{em}}(B_{em} + \frac{\pi T_{st}}{\omega_{st}}) \tag{3.35}$$

Thus, the discrete domain transfer function becomes

$$\frac{\omega(z)}{T_e(z)} = \frac{(T_s / J_{em})z}{z-(1-A)} \tag{3.36}$$

Fig.3.20 shows the open loop pole locations z = (1-A) for the increasing value of A.

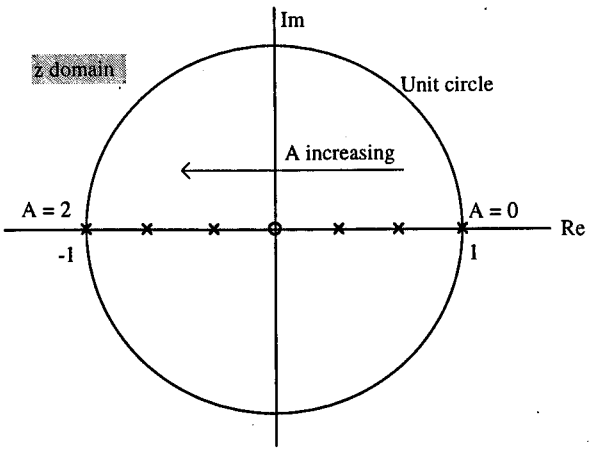


Figure 3.20 The open loop pole locations according to the value of A

From Fig.3.20, it can be seen that if A > 2, the open loop pole (1-A) will be outside the unit circle and system will be unstable. Therefore, the system is locally unstable in the stiction region when

$$A = \frac{T_s}{J_{em}}(B_{em} + \frac{\pi T_{st}}{\omega_{st}}) > 2 \tag{3.37}$$

The ratio of T_{st}/ω_{st} is thus limited; ω_{st} can not be very low if T_{st} is chosen high (assuming T_s, J_{em} and B_{em} are kept constant).

The open loop emulation of the stiction model

For the initial investigation, no outer speed controller (PI speed controller-1) is placed around the load. The closed loop speed control of this load model will be considered later in this section.

The stiction problem generally appears at or near zero speed. In order to see the stiction effects, the driving torque profile is chosen as indicated in Fig.3.21, where $T_e = 0$ for $t > 2T_p$.

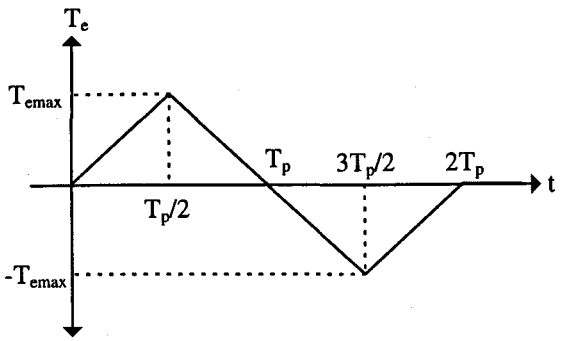


Figure 3.21 Applied electrical driving torque profile

The block diagram of the open loop experimental implementation is shown in Fig.3.22. Note that the comparison system (simulation model) is directly implemented using (3.30) and (3.31) in SIMULINK.

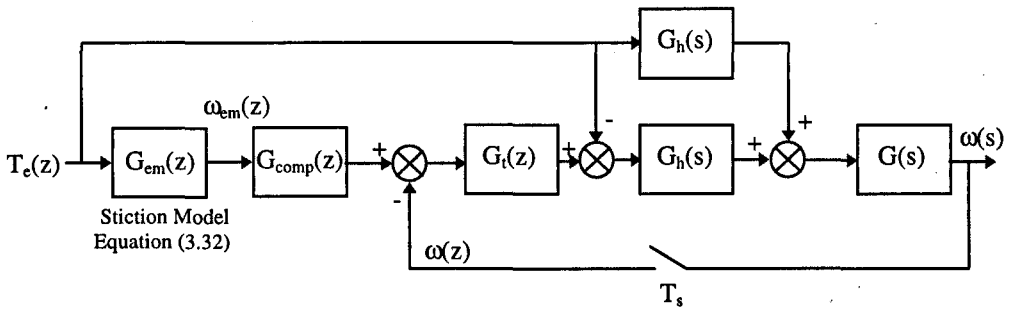


Figure 3.22 The block diagram of the experimental implementation of the stiction model

In order to see the stiction effect, all the parameters are kept constant except T_{st} which is increased gradually to monitor the increasing stiction effect upon the shaft speed. The parameters kept constant are given in Table 3.2.

Table 3.2 Parameters for the open loop stiction emulation

$J_{em} = 0.007 kgm^2$ (2J)	$T_p = 0.5s$	$\omega_{st} = 1rad/s$
$B_{em} = 0.01Nms$ (14.3B)	$T_{emax} = 1Nm$	$T_s = 2.5ms$

Note that ω_{st} is set to 4 times the resolution of the speed encoder. If ω_{st} is set to less than 2 times the resolution, the response is dominated by the resolution effects. The emulated friction is especially set to a large value (14.3B) to restrict the speed since the emulation is open loop.

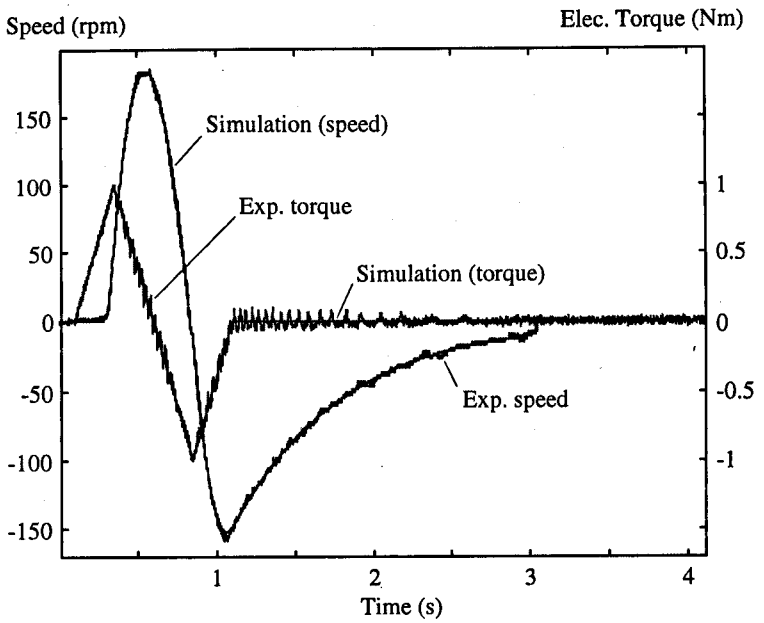


Figure 3.23 Experimental and simulated speed and electrical torque responses for $T_{st} = 0.7$Nm

Fig.3.23, 3.24 and 3.25 shows the experimental and simulated open loop speed and electrical torque responses for T_{st} = 0.7, 0.9375 and 0.95 respectively and all the experimental speed responses are given on the same scale in Fig.3.26.

In Fig.3.25, due to the high stiction torque, the speed can not reach ω_{st} with this applied torque T_e. Thus, it does not get rid of the stiction effect. Although the speed is around the speed resolution (2.4 rpm), the experimental speed is still tracking the simulation result as seen in Fig.3.25.

In region "a" of Fig.3.26, the shaft initially speeds up slowly because T_e must exceed the stiction torque. This region of slow acceleration gets larger with increasing T_{st}. When the speed becomes higher than ω_{st}, the effect of the stiction disappears and the shaft speeds up more quickly. But this is not valid for the response of T_{st} = 0.95 because the speed never reaches ω_{st} due to the high stiction torque, so that the speed stays around zero for the all regions. The applied torque T_e reverses as shown in Fig.3.21. Therefore, for T_{st} = 0.7 and 0.9375, the speeds reach a maximum value and then start to decrease.

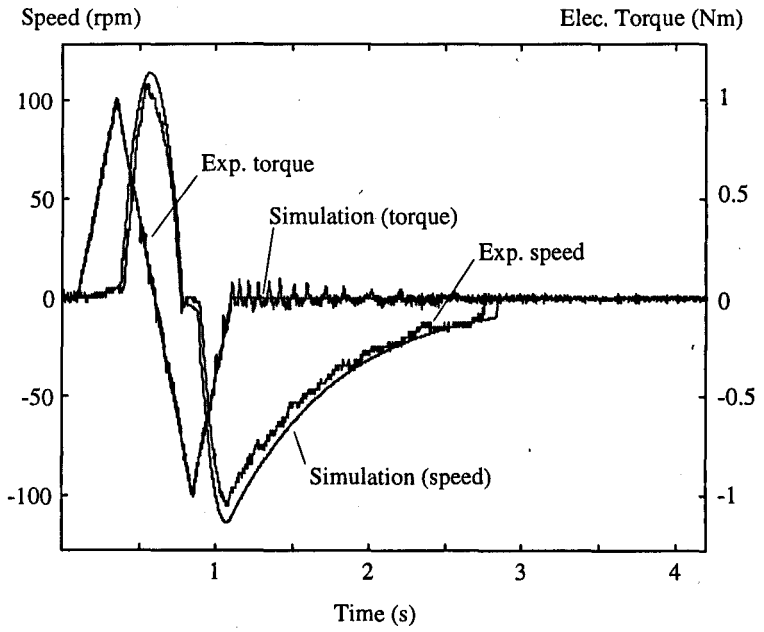


Figure 3.24 Experimental and simulated speed and electrical torque responses for T_{st} = 0.9375 Nm

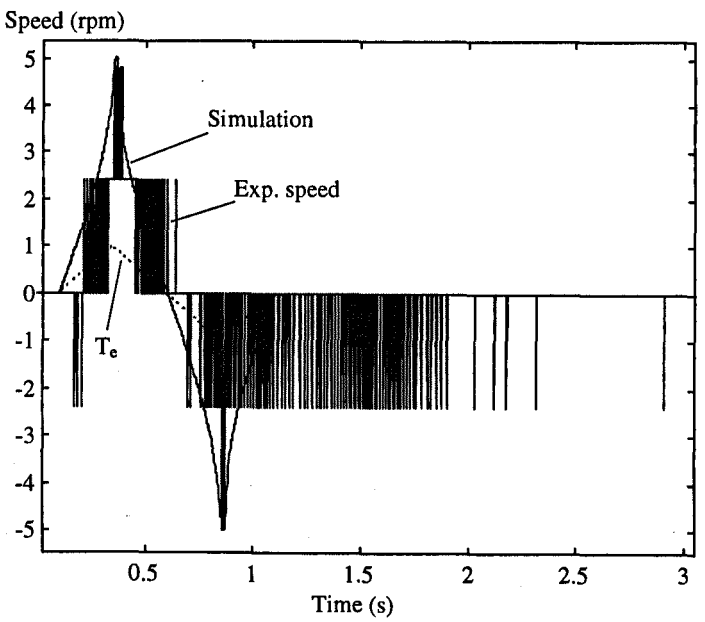


Figure 3.25 Experimental and simulated speed and electrical torque responses for T_{st} = 0.95 Nm

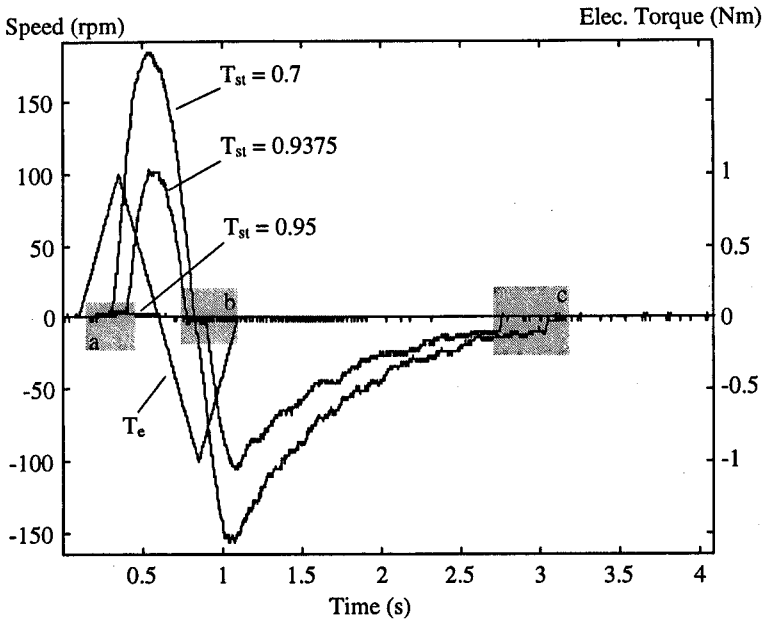


Figure 3.26 Experimental speed responses for T_{st} = 0.7, 0.9375 and 0.95 Nm

In region "b" of Fig.3.26, when the speed falls below ω_{st}, stiction becomes effective again. For T_{st} = 0.7 and 0.9375, the system enters this zone $(|\omega| \le \omega_{st})$ with a negative acceleration. The stiction seems not to affect the speed response for T_{st} = 0.7 due to the high acceleration. The acceleration for T_{st} = 0.9375 is not high enough to prevent stiction effect becoming apparent; the speed deceleration is very slow because of the high stiction torque. T_e becomes zero for t > 1s. After the speeds reach their negative maximum value, they start to fall freely towards zero.

In region "c" of Fig.3.26, the speeds enter the stiction zone $(|\omega| \le \omega_{st})$ again. Since this time T_e = 0, the stiction torque quickly stops the shaft's movement.

Finally, Fig.3.27 shows the effect of local instability when the stiction parameters are set to ω_{st} = 0.5rad/s, T_{st} = 2Nm, T_{emax} = 1.3Nm so violating the stability condition of (3.37) since A = 4.4915. The local instability is clearly seen as the speed attempts to settle about zero. In practice, the local instability appears as shaft vibrations.

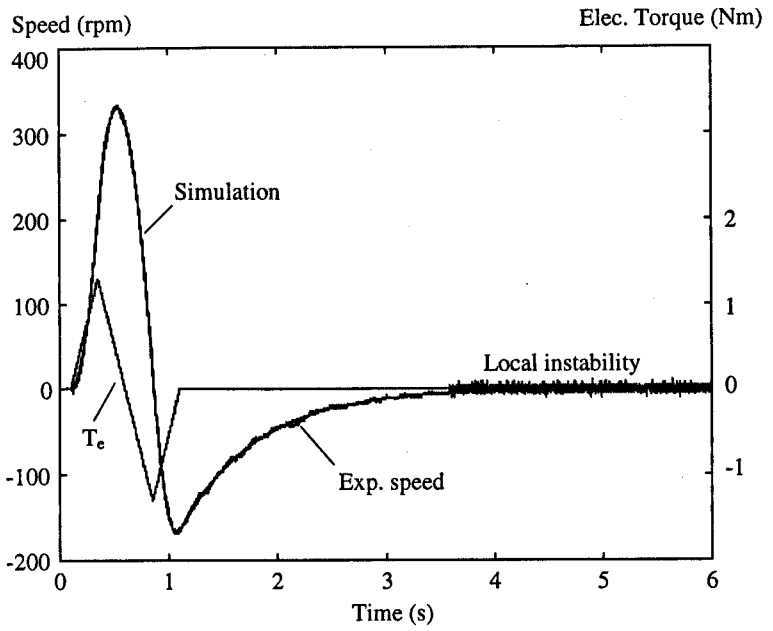


Figure 3.27 Experimental and simulated speed responses for local instability case

The stiction emulation in a closed loop system

As discussed in Section 3.1, it is important that the emulation should preserve the model mechanical dynamics when the emulated load is a part of a closed loop control system. Thus, the load model (3.32) is implemented in the G_{em} block of the experimental closed loop speed control system shown in Fig.3.12. Similarly, for the comparison system, (3.30) and (3.31) are implemented in the load block of the simulated system shown in Fig.3.11. In order to see the effect of the stiction clearly, a triangular reference speed (peak-to-peak ±50rpm) is applied. Fig.3.28a and 3.28c show the speed and the electrical torque responses of the experimental and simulated systems for the parameters J_{em} = 2J, B_{em} = B, ω_{st} = 0.7rad/s and T_{st} = 0.7Nm. In the stiction region ($|\omega| \le \omega_{st}$), the stiction torque prevents the output speed to follow the reference speed and thus results in a speed error. The error is integrated by the PI speed controller. When the speed reaches the boundary of the stiction region (note that the stiction region is very narrow, just 2.5 times of the encoder resolution), the stiction torque suddenly disappears and T_e becomes much higher than a required value due to the integrated error during the time in the stiction region. This results in a sudden change on the output speed as seen in Fig.3.28. Again, very good agreement between simulated and experimental responses is achieved.

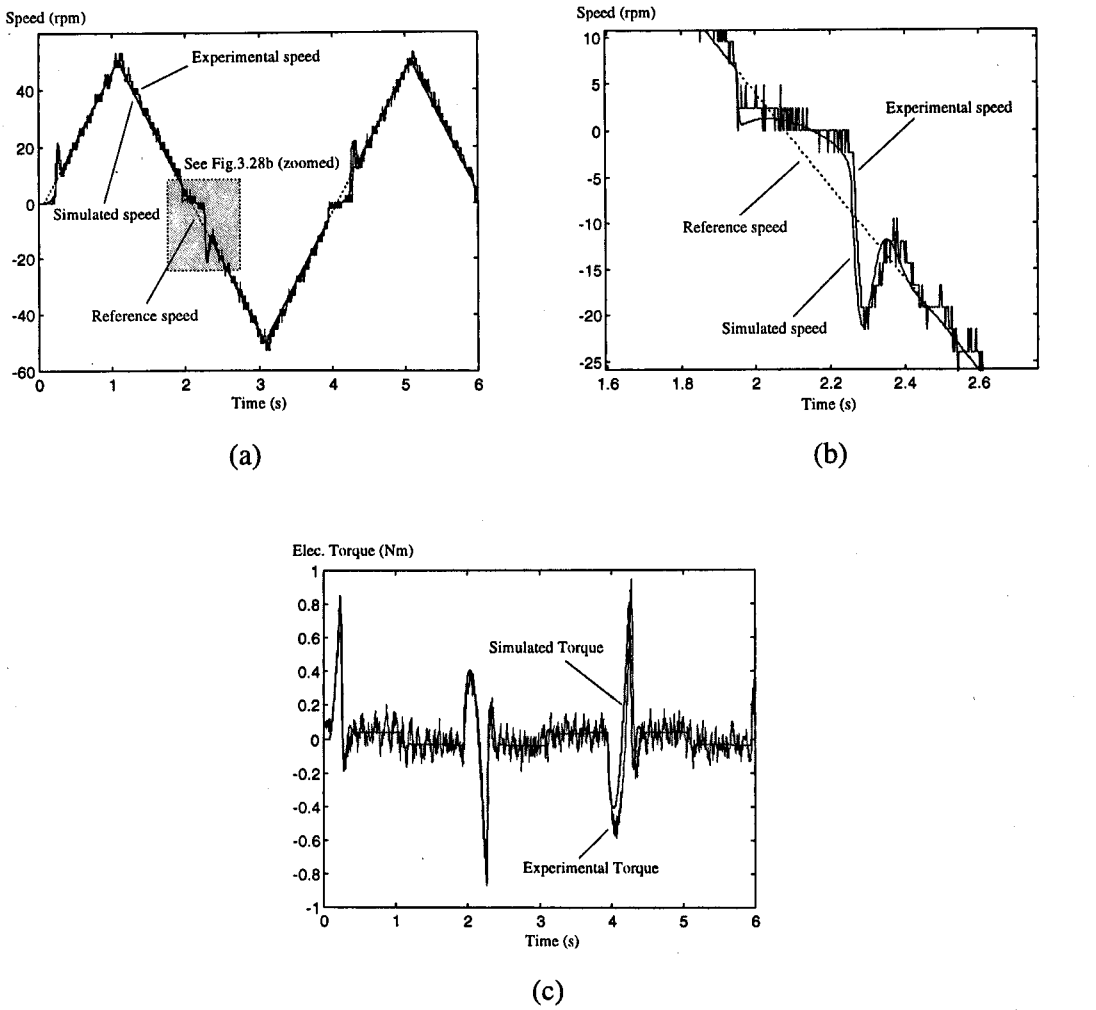


Figure 3.28 The closed loop experimental and simulated responses of the load with stiction for an triangular speed reference (peak-to-peak ±50rpm)

(a) The speed responses (b) Shaded area of Fig.3.28a

(c) The electrical torque responses

3.6.5 Watt Governor

The Watt governor is a good example of a non-linear device having an effective inertia (and friction) variation during motion. The physical structure of the governor is shown in Fig.3.29. It consists of two pendulums fixed at O; when the shaft rotates, the balls fly outwards due to the centrifugal force. The original governor contains further mechanics for the regulation of the shaft speed (e.g. the balls are connected by a link to a sleeve sliding parallel on the shaft). The extra linkages are of little or no interest if the aim is only to emulate a realistic variable inertial load.

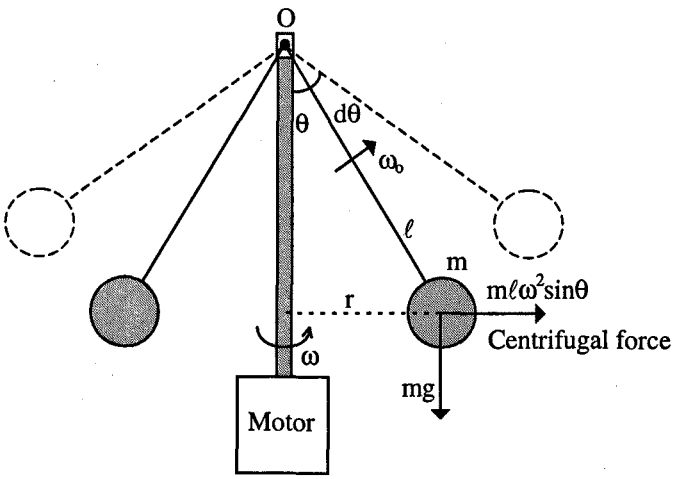


Figure 3.29 The physical structure of a Watt governor

The moment about the fixed point O of all the forces acting on the mass m is equal to the time rate of change of angular momentum of the mass about O [53] :

$$\sum \overline{M}_o = \dot{\overline{H}}_o = \overline{\ell} \times m\ell \dot{\overline{\omega}}_o \tag{3.38}$$

where $\sum \overline{M}_o$ is the total moment about O, $\dot{\overline{H}}_o$ is the time rate of change of angular momentum of the mass m about O, $\overline{\ell}$ is the position vector and $\dot{\overline{\omega}}_o$ is the time rate of change of the angular velocity of m about O. Equation (3.38) can be written for the system shown in Fig.3.29 as :

$$m\ell^2\omega^2 \sin\theta\cos\theta - mg\ell\sin\theta = m\ell^2\ddot{\theta} \tag{3.39}$$

The angular momentum about the motor shaft axis can be written as [53] :

$$\left|\overline{H}\right| = \left|\overline{r} \times mr\overline{\omega}\right| = 2m\ell^2 \sin^2\theta\omega \tag{3.40}$$

and the total moment or torque becomes

$$\sum Moment = T_e - B_{em}\omega = \left|\dot{H}\right| + J_{em}\dot{\omega} \tag{3.41}$$

where J_{em} and B_{em} are shaft inertia and friction as before. Thus we have

$$T_e = J_{ef}\dot{\omega} + B_{ef}\omega \tag{3.42}$$

where $J_{ef} = J_{em} + 2m\ell^2 \sin^2\theta$ and $B_{ef} = B_{em} + 2m\ell^2\dot{\theta}\sin(2\theta)$

In order to include some damping, friction against motion in θ can be introduced at the junction O. Equation (3.39) becomes

$$\frac{1}{2}m\ell^2\omega^2 \sin 2\theta - mg\ell \sin\theta - B_o\dot{\theta} = m\ell^2\ddot{\theta} \tag{3.43}$$

where B_o is the viscous friction constant at the junction O.

Defining the states as $\underline{x} = [\omega, \dot{\theta}, \theta]^T$, the state equations can be derived from (3.42) and (3.43) as

$$\begin{aligned}
\dot{x}_1 &= -\frac{B_{em} + 2m\ell^2 x_2 \sin(2x_3)}{J_{em} + 2m\ell^2 \sin^2(x_3)} x_1 + \frac{1}{J_{em} + 2m\ell^2 \sin^2(x_3)} T_e \\
\dot{x}_2 &= -\frac{B_o}{m\ell^2} x_2 + \frac{1}{2} x_1^2 \sin(2x_3) - \frac{g}{\ell}\sin(x_3) \\
\dot{x}_3 &= x_2
\end{aligned} \tag{3.44}$$

Using the backward discretization, the difference equations become

$$\begin{aligned}
x_1(k) &= (1 - \frac{B_{em}T_s + 2m\ell^2 x_2(k-1)\sin(2x_3(k-1))T_s}{J_{em} + 2m\ell^2 \sin^2(x_3(k-1))}) x_1(k-1) + \frac{T_s}{J_{em} + 2m\ell^2 \sin^2(x_3(k-1))} T_e(k) \\
x_2(k) &= (1 - \frac{B_o T_s}{m\ell^2}) x_2(k-1) + \frac{T_s}{2} x_1^2(k-1)\sin(2x_3(k-1)) - \frac{gT_s}{\ell}\sin(x_3(k-1)) \\
x_3(k) &= x_3(k-1) + T_s x_2(k-1)
\end{aligned} \tag{3.45}$$

Note that due to non-zero radius of the balls, an initial and minimum θ value (θ_{init}) should be introduced to obtain a more realistic model. The equations (3.44) and (3.45) are thus supplemented with the condition

if $x_3 < \theta_{init}$ then $x_3 = \theta_{init}$.

The open loop emulation of the Watt governor

No outer speed controller (PI speed controller-1) is placed around the load for the initial investigation. The closed loop speed control of this load model will be considered later.

The block diagram of the experimental implementation is shown in Fig.3.30. Note that the simulated system is directly implemented using the state equations (3.44) in SIMULINK.

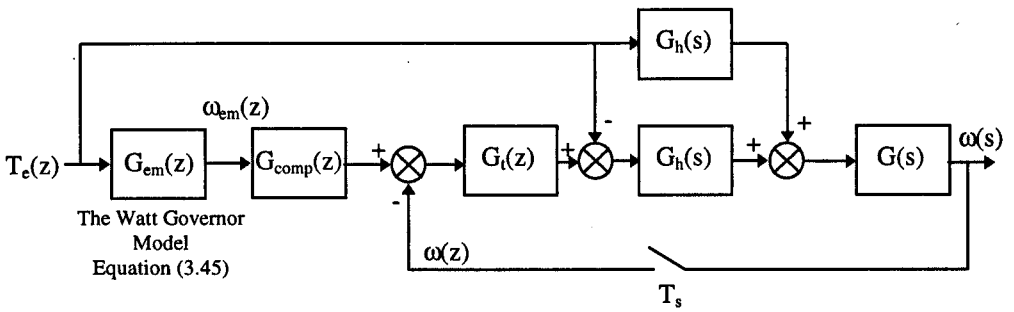
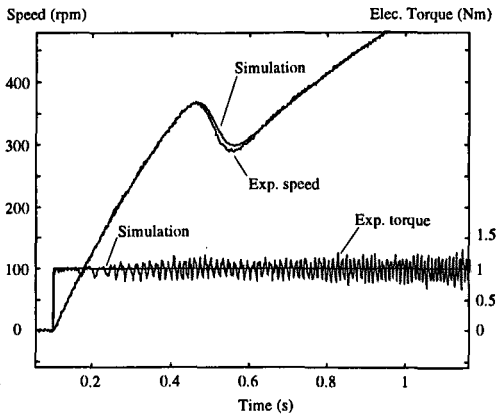
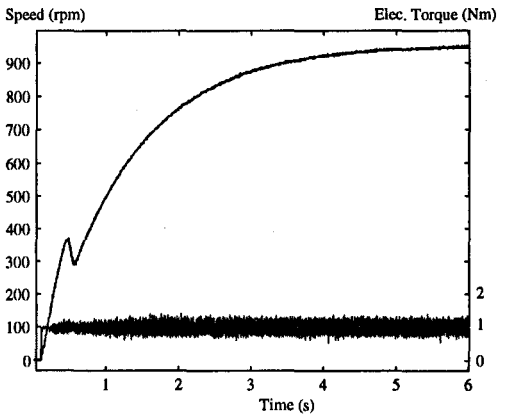


Figure 3.30 The block diagram of the experimental implementation of the Watt governor model (the open loop emulation)

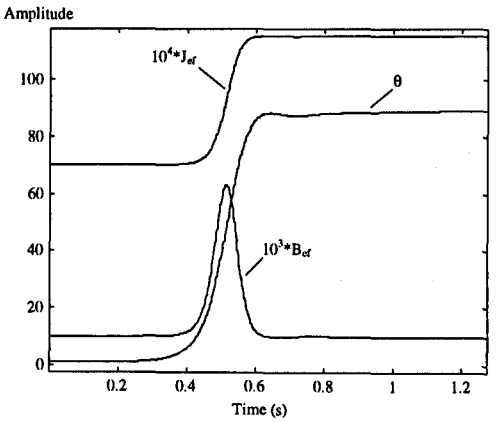
Fig.3.31a and b show the experimental speed and electrical torque responses in comparison with SIMULINK simulation when the system is driven by a 1Nm step torque input (the parameters are m = 0.1kg, ℓ = 0.15m, B_o = 0.1Nms, J_{em} = 0.007kgm^2 (2J), B_{em} = 0.01Nms (14.3B)). A glitch occurs at about 0.5s; this is due to the sudden increase in θ caused by the centrifugal force lifting the balls. As the balls fly outwards, the effective friction, B_{ef}, rapidly increases and slows down the increase in shaft speed. Again, excellent agreement between the simulated and the experimental responses is achieved. Fig.3.31c shows the variations of the angle θ, J_{ef} and B_{ef} during the shaft acceleration. These values are of course taken from the governor model implementation of the emulated load block, $G_{em}(z)$, of Fig.3.30 since they do not exist as real mechanical parameters/variables in the experimental rig.



(a)



(b)



(c)

Figure 3.31

(a) Experimental and simulated speed and electrical torque responses to a step torque demand

(b) Expanded time of Fig.3.31a

(c) The variations of θ, J_{ef} and B_{ef}

The steady state value of θ can be calculated from (3.43) as

$$\theta_{ss} = \cos^{-1}\left(\frac{g}{\ell \omega_{ss}^2}\right) \tag{3.46}$$

where ω_{ss} is the steady state value ω̇. Similarly, ω_{ss} can be derived from (3.42) as

$$\omega_{ss} = \frac{T_e}{B_{em}} \tag{3.47}$$

If the parameters used in the emulation are substituted in (3.46) and (3.47) then ω_{ss} and θ_{ss} become 100rad/s (955rpm) and 89.63° respectively. These calculated values validate the results shown in Fig.3.31.

The emulation of the Watt governor in a closed loop speed control system

The emulation of the Watt governor is placed in the experimental closed loop speed control system. Fig.3.32 and 3.33 show the experimental and simulated speed and electrical torque responses to a step reference input (300rpm) for m = 0.1kg and m = 0.3kg respectively. The other parameters are kept constant for both cases (ℓ = 0.15m, B_o = 0.07Nms, J_{em} = 2J, B_{em} = B). The PI speed controller-1 designed in Section 3.5.1 is also kept constant for both values of m. Fig.3.34 illustrates the variation of the angle θ (see Fig.3.29), the effective inertia J_{ef} and the effective viscous friction B_{ef} during the motion for both values of m. These values are taken from the governor model implementation of the emulated load block of Fig.3.12 (i.e., they are the calculated variables in the experimental software). Note that these variables do not exist as real mechanical variables in the experimental rig.

Fig.3.32 and 3.33 shows that increasing the mass m extensively affect the speed and electrical torque responses. This is because, when m = 0.3kg, the effective inertia and friction increases more comparing to the case of m = 0.1kg as clearly seen in Fig.3.34.

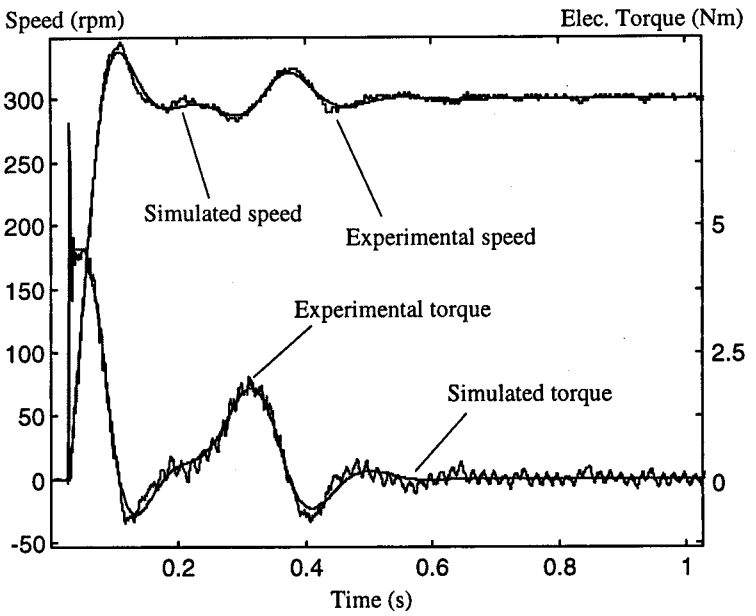


Figure 3.32 Experimental and simulated speed and electrical torque responses for m = 0.1kg

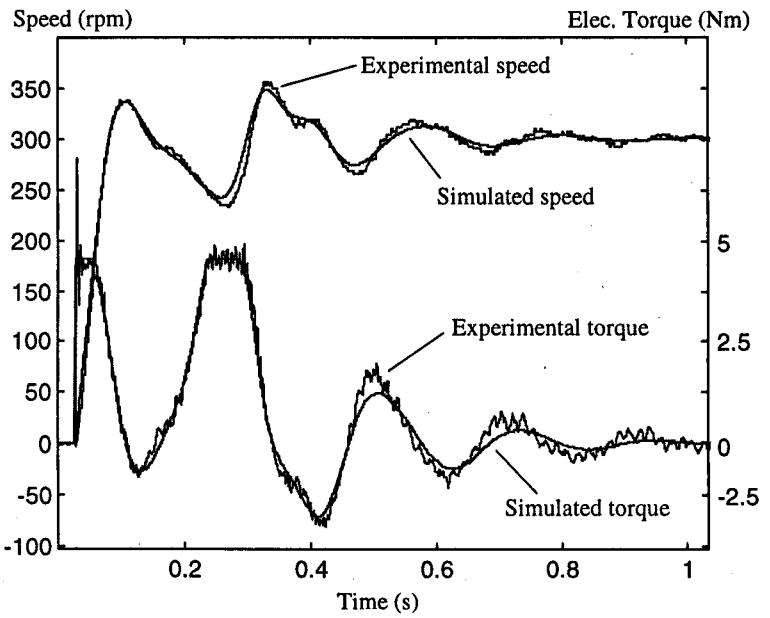


Figure 3.33 Experimental and simulated speed and electrical torque responses for m = 0.3kg

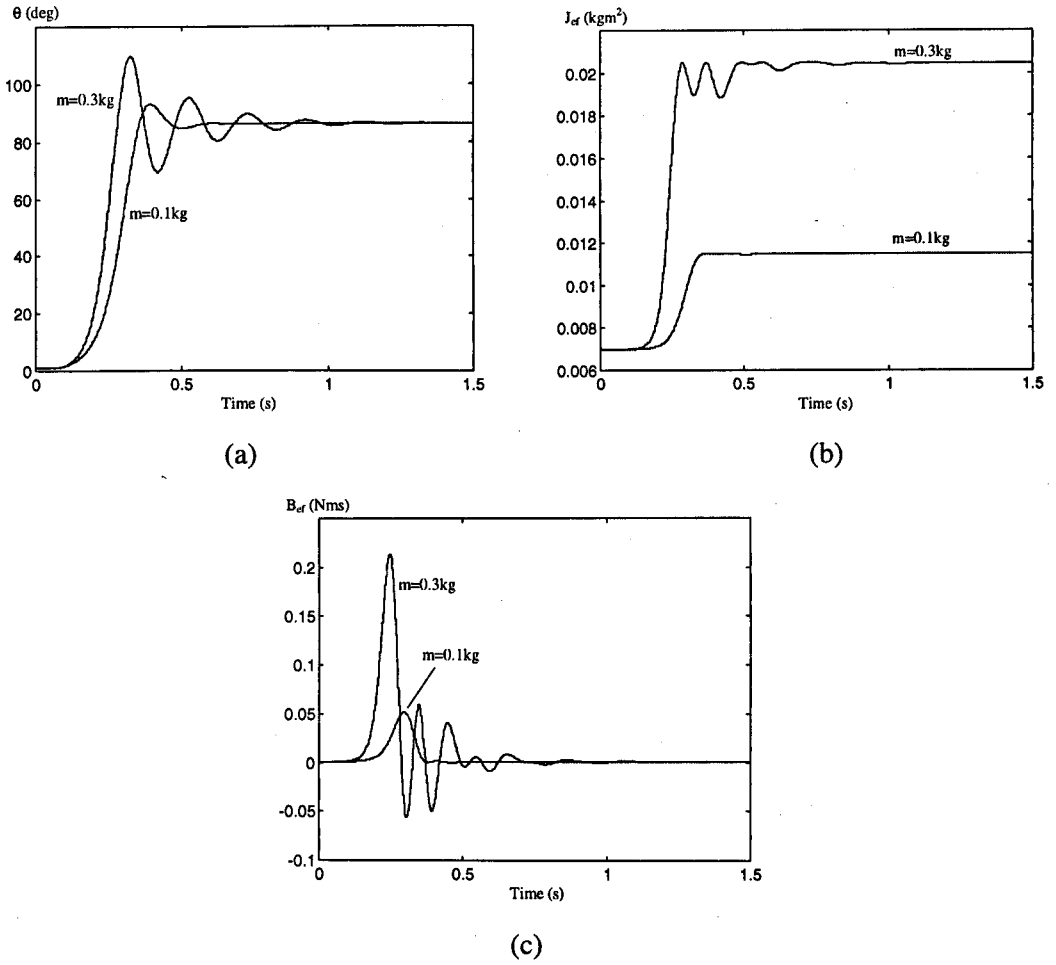


Figure 3.34 Variation of
(a) θ (b) J_{ef} (c) B_{ef}

3.7 Conclusions

In this chapter, a new dynamometer control strategy has been developed for the emulation of both linear and non-linear mechanical load dynamics such that these dynamics (or pole-zero structure for a linear load) are preserved during the emulation. The emulation can thus be used for the testing of motor drive control strategies. The emulation strategy is based on a speed tracking control with *implicit* feed-forward of the inverse dynamics and compensation for closed loop tracking control dynamics. The inverse dynamics do not need to be implemented in practice; no derivative computations are necessary and the scheme does not suffer from noise.

Experimental validation is based on the principle of output speed equivalence to a given input drive torque. Experimental results on a vector-controlled induction motor-dynamometer rig have shown excellent equivalence with simulation. The equivalence achieved when the emulated dynamics are placed in a closed speed control loop are impressive and indicate that the mechanical dynamics are preserved for frequencies within the control loop bandwidth.

The validation has not been done on a real mechanical rig. However, it is noted that any electronic emulation can only be an emulation of a load model. If the emulation is validated against the model (as this chapter has shown), then it follows that using a real load would serve only to validate the model and not the emulation.

The emulation requires the drive motor torque reference signal. This is not a restriction given the aim of this work to provide a test-bed for motor drive control strategies. If a torque reference signal is not available, it is recommended, where possible, that the motor drive voltages and currents be measured and fed to an electrical torque observer based on the model equations of the drive machine. It is felt that errors in the estimated torque would be less problematic than the discretization and noise effects arising from the inverse dynamics.

In Chapter 5, the emulations will be used to provide linear, non-linear and time varying loads for the experimental validation of a new robust control method developed using sliding mode and fuzzy control methods.

Chapter 4

Equivalence of Fuzzy and Classical Controllers : An Approach to Fuzzy Control Design

4.1 Introduction

Fuzzy theory was first introduced by Zadeh in 1965 [29]. During the last two decades, Fuzzy Logic Control (FLC) has emerged as one of the most attractive and fruitful areas for research in the application of the fuzzy theory to the real engineering problems. FLC is actually a practical alternative to the conventional control methods for a variety of control applications since it provides a convenient method for implementing linear and non-linear controllers via the use of both heuristic and mathematical information.

Fuzzy logic has found wide applications in the control of electrical drive systems. However, the classical linear controllers (e.g., PI, PD, PI+lead, etc.,) are still the most widely used controllers in the practical applications due to their simplicity of design and microprocessor implementations. This chapter addresses the fact that any linear controller can be exactly represented by a Fuzzy Controller (FC) for a given input universe of discourse. It may be thought that there is no point in implementing a linear control law by a FC; however, it should not be viewed as a goal itself but as a preliminary step in designing FCs for systems with known non-linearity (deterministic non-linear systems). It seems more reasonable to implement the desired non-linear global behaviour by piecewise linear approximation. In addition, although there are many successful fuzzy speed and position control applications, usually these controllers are designed by trial and error methods [17,30]. In most cases, no formal approach is used to chose the number and shape of the membership functions. The derivation of the fuzzy equivalence of a linear controller actually generates an automatic design procedure for the FCs. Finally, from another aspect, the equivalence principle may also help to obtain a fair

comparison between fuzzy and linear controllers: there are many research papers presenting such performance comparisons between fuzzy and linear controllers [17,31-34]. Some of these papers result in a doubt about the fairness of the comparisons. It is reasonable to assume that in order to have a fair comparison, the controllers under evaluation should give exactly or very similar closed loop output responses to the same input references for the nominal conditions [31]. Hence, when the parameters of the plant are changed or an external disturbance is applied, one can easily see which method gives the more robust control performance. Using the equivalence principle, the FC under evaluation may be designed to satisfy this comparison criteria.

Section 4.2 gives a brief overview of the FLC theory. The fuzzy equivalence of a second order linear controller is discussed in Section 4.3. The fuzzy equivalence of a general linear controller is derived in Section 4.4. For a class of non-linear deterministic systems, a FC design is explained in Section 4.5 and finally the control of the non-deterministic systems is discussed in Section 4.6.

4.2 Fuzzy Logic Control : A Brief Overview

In the design of control systems, as a general principle, all the available information should be able to be used efficiently. In most practical systems, there are mainly three types of information: an approximate mathematical model of the system, sensory measurements and the experience of the human experts. Fuzzy logic basically provides a convenient transform technique to combine all this information in a common framework for implementing linear and non-linear controllers.

In this section, the fuzzy logic and fuzzy set operations are briefly summarised and the basic structure of FC is given with details of its main components.

4.2.1 Fuzzy Logic and Fuzzy Set Operations

In Boolean logic based set theory, a particular object is either a member of a given set (logic '1') or not a member of the given set (logic '0'). These kind of sets are called crisp sets. However, in fuzzy set theory based on fuzzy logic, a particular object has a degree of

membership in a given set. This membership degree may take a value in the interval [0,1]. In other words, a particular object may *partially* be a member of a set. This property allows fuzzy logic to transform the human knowledge base into a mathematical expression more efficiently since it is more close to the human thinking compared to the TRUE-FALSE logic. For example, Fig.4.1 shows the crisp and fuzzy sets for the speed of an electrical motor. The speed is defined by the crisp and fuzzy sets characterised by the membership functions named LOW, MEDIUM and HIGH.

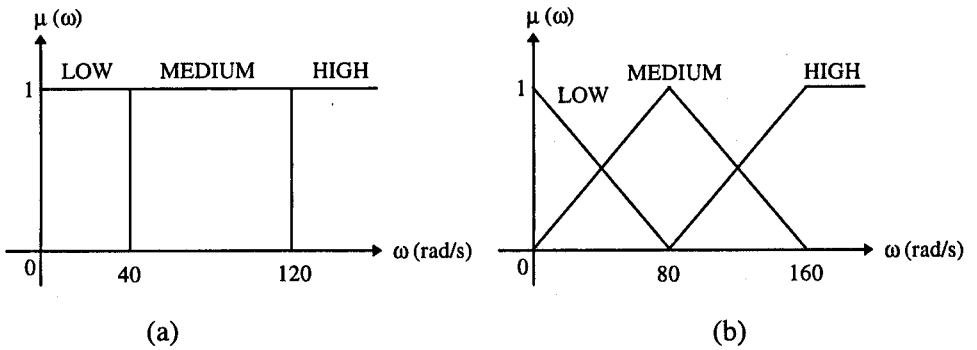


Figure 4.1 Membership functions of
(a) Crisp sets
(b) Fuzzy sets

As seen in Fig.4.1, for example, a speed value 35 rad/s *completely* belongs to the set LOW in crisp sets. However, in fuzzy sets, it belongs to the sets LOW and MEDIUM by 56.25% (degree of membership = 0.5625) and 43.75% (degree of membership = 0.4375) respectively. The membership functions shown in Fig.4.1b are called triangular shaped membership functions. Trapezoidal shaped and bell-shaped (Gaussian) membership functions are also widely used in literature [37,40,54,55].

In fuzzy set terminology, all the possible values of a variable are called the *universe of discourse*, and the fuzzy sets characterised by membership functions cover the whole universe of discourse. In other words, a universe of discourse for an input or output of a fuzzy system is simply the range of values the input and output can take on. For example, in Fig.4.1b, while the vertical axis represents certainty, the horizontal axis is the universe of discourse for the speed $\omega(t)$ since it provides the range of values of the speed.

Fuzzy Set Operations :

Assume that A and B are two fuzzy sets with the universe of discourse X and they are characterised by the membership functions $\mu_A(x)$ and $\mu_B(x)$ respectively, where $x \in X$. The basic set operations for fuzzy sets are defined via the membership functions as follows :

Union : The membership function $\mu_{A\cup B}(x)$ of the union $A\cup B$ is defined for all $x \in X$ by

$$\mu_{A\cup B}(x) = \max\{\mu_A(x), \mu_B(x)\} \tag{4.1}$$

Intersection : The membership function $\mu_{A\cap B}(x)$ of the intersection $A\cap B$ is defined for all $x \in X$ by

$$\mu_{A\cap B}(x) = \min\{\mu_A(x), \mu_B(x)\} \tag{4.2}$$

Complement : The membership function $\mu_{A'}(x)$ of the complement of a fuzzy set A is defined for all $x \in X$ by

$$\mu_{A'}(x) = 1 - \mu_A(x) \tag{4.3}$$

The fuzzy set operations union, intersection and complement correspond to Boolean OR, AND and NOT operators respectively. The other commonly used union and intersection operations are defined as follows :

$$\mu_{A\cup B}(x) = \min\{1, \mu_A(x) + \mu_B(x)\} \tag{4.4}$$

$$\mu_{A\cap B}(x) = \mu_A(x) \cdot \mu_B(x) \tag{4.5}$$

The details of fuzzy set theory is beyond the scope of this study and can be found in [37,40].

4.2.2 Basic Structure of a Fuzzy Controller

A Fuzzy Controller (FC) basically consists of four main components as shown in Fig.4.2. These are the fuzzifier, the rule-base, the inference mechanism (also called the inference engine) and the defuzzifier.

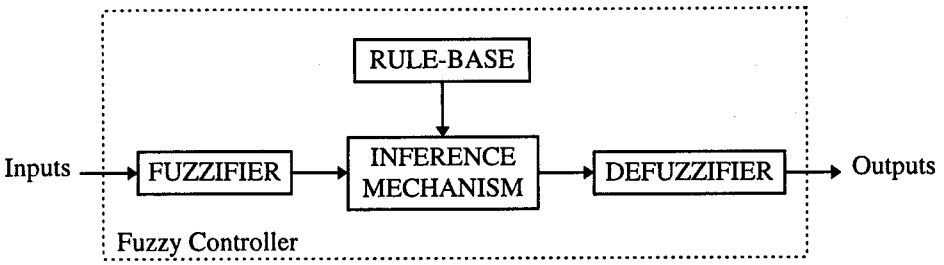


Figure 4.2 Basic structure of a fuzzy controller

In this section, these components are explained in details. The design of the FCs will be considered in Sections 4.3 - 4.5.

FUZZIFIER :

The fuzzifier converts the crisp values of input variables into information which can be easily used in the inference mechanism. After the fuzzification process, each input value is represented by a membership degree for each fuzzy set defined for the corresponding input variable. For example, assume that the controller inputs are error e(k) and change-in-error $\delta e(k)$, where $\delta e(k) = e(k) - e(k-1)$. Fig.4.3 shows the fuzzy sets characterised by the membership functions for these inputs. It also shows the fuzzification of the input values e(k) = 30 rad/s and $\delta e(k)$ = -15 rad/s. In Table 4.1, the fuzzification results (i.e. the membership degrees of the input values e(k) = 30 and $\delta e(k)$ = -15) are presented for the fuzzy sets which are named NL, NS, ZE, PS and PL where the letters P, N, L, S and ZE refer to Positive, Negative, Large, Small and Zero respectively. The membership degrees shown in Table 4.1 are used in the inference mechanism together with the rule-base to obtain a resultant output fuzzy set.

The fuzzification method explained above is known as *singleton fuzzification* which is the most widely used fuzzification method in the control applications [40,57]. There are also other fuzzification methods (e.g., Gaussian and triangular fuzzification) which produce a fuzzy set

instead of a membership degree, but they are not preferred in control applications since they introduce an extra computational complexity in the fuzzification and inference processes. Additionally, a very good functional capability can be achieved with the fuzzy systems when only singleton fuzzification is used [40,57].

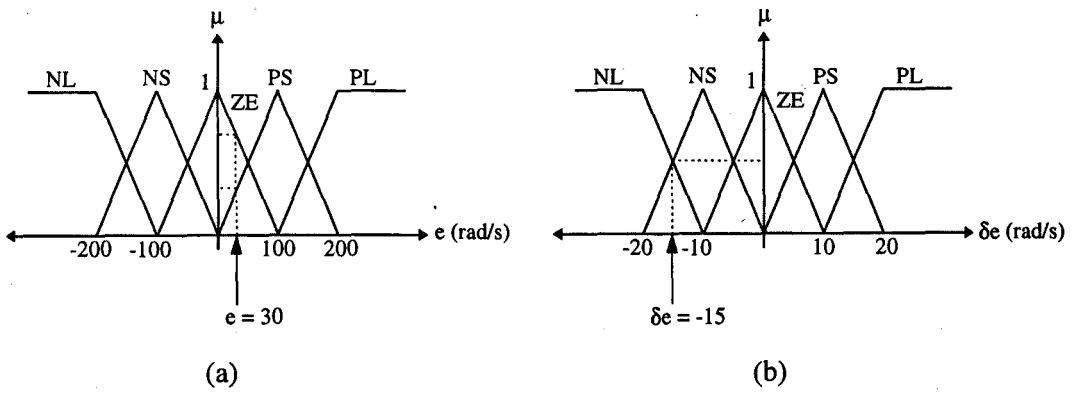


Figure 4.3 Membership functions for the input variables

(a) error

(b) change-in-error

Table 4.1 Fuzzification results for the input values e(k) = 30 and δe(k) = -15

Error (e(k) = 30)		Change-in-Error (δe(k) = -15)	
Membership Degree	Fuzzy Sets	Membership Degree	Fuzzy Sets
0.7	ZE	0.5	NS
0.3	PS	0.5	NL
0	NL, NS, PL	0	ZE, PS, PL

RULE-BASE :

The rule base of a FC consists of a set of fuzzy IF-THEN rules which are usually obtained from the expert's linguistic descriptions. However, the rules can also be derived from a mathematical relation. This will be discussed in Section 4.3 and 4.4. The rule base is the heart of the FC since all the other components are used to implement these rules in a reasonable and efficient manner. Table 4.2 shows a rule-base consisting of 25 rules. The input variables are the error and the change-in-error whose membership functions are given in Fig.4.3. The output of the FC

is u and Fig.4.4 shows its membership functions. A rule from Table 4.2, for example, can be written as

IF (e is ZE **AND** δe is NS) **THEN** (u is NS)

The general form of the linguistic rules shown in Table 4.2 is

IF premise **THEN** consequent

The premises are associated with the controller inputs and the consequents are associated with the controller outputs. Each premise consists of two terms connected with the AND operator for this example. Note that the premise may contain more than two terms and they may be connected with the other fuzzy set operators such as OR and NOT. The consequent may also be composed of the conjunction of several terms if the number of output is more than 1.

Table 4.2 Rule-base

u / e \ δe	NL	NS	ZE	PS	PL
NL	NL	NL	NM	NS	ZE
NS	NL	NM	NS	ZE	PS
ZE	NM	NS	ZE	PS	PM
PS	NS	ZE	PS	PM	PL
PL	ZE	PS	PM	PL	PL

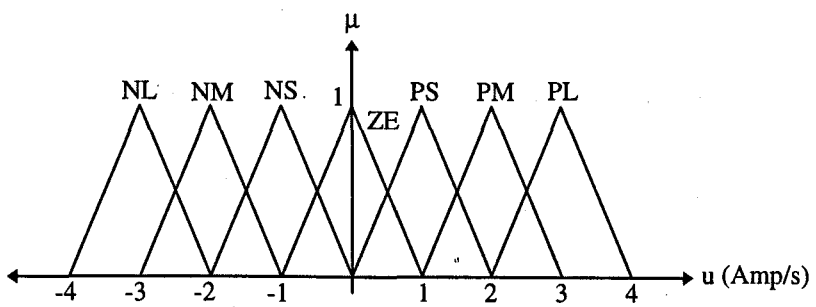


Figure 4.4 The membership functions for the output variable u

INFERENCE MECHANISM :

In the inference process, the outputs of the fuzzifier (i.e. the membership degrees of the input values for each fuzzy set of the corresponding input variable) and the rule-base are used to produce an output fuzzy set (or sets) which will be interpreted by the defuzzifier to calculate the output of the controller. There are two main approaches to the inference mechanism [37,40]. The first is *composition based inference* and the second is *individual-rule based inference*. In the composition based inference method, all rules in the rule-base are first combined into a single fuzzy relation. Then the inference is performed between the fuzzified inputs and the fuzzy relation representing the meaning of the overall set of rules. Finally, a fuzzy set describing the fuzzy value of the overall control output is obtained. However, in the individual-rule based inference method, each rule in the rule-base determines an individual output fuzzy set and the overall output of the fuzzy inference mechanism is obtained by aggregating the individual output fuzzy sets. Usually, the individual-rule based inference method is preferred because it is computationally more efficient and saves a lot of memory comparing to the composition based inference method. The individual-rule based inference mechanism will be considered in this section.

In the inference mechanism, there are three main operation types which should be considered : the first type is the operations between the terms of the premises of the rules in the rule-base. The terms are usually connected by the operators such as AND, OR and NOT. However, in most of the control applications, only the AND operator is used to connect the terms in a premise [37,40]. Basically, these operators perform the operations between the membership degrees of the input values and thus a single result called *certainty of the rule* is obtained for each rule in the rule base. The second type operation is the *implication* which is the operation between the certainty of the rule and the output fuzzy set of the corresponding rule. After the implication process, each rule in the rule base results in an *implied output fuzzy set*. The third type operation is the *aggregation* which combines all the implied output fuzzy sets to obtain a *resultant output fuzzy set*. Some of the defuzzification methods directly use the implied output fuzzy sets produced after the implication process. Thus, in this case, the aggregation process may-not be required. This will be discussed later in this section under the DEFUZZIFIER title.

The most widely used operators for the AND, OR, IMPLICATION and AGGREGATION operations are as follows [37,40,54,55] :

AND : min, prod
OR : max, probor
IMPLICATION : min, prod
AGGREGATION : max, sum

where prod, probor and sum represent the *algebraic product, probability or* and *algebraic summation* respectively.

Let us consider the rule-base given by Table 4.2 and the membership functions shown in Fig.4.3 and 4.4. Assume that *prod, min* and *max* are used for the AND, IMPLICATION and AGGREGATION operations respectively. In order to have a convenient notation for the formulations, the rules in the rule-base shown in Table 4.2 can be represented as

IF e is E_i **AND** δe is δE_j **THEN** u is U_{ij}

where i and j are the indexes referring to the rows and the columns of the rule-base respectively. The fuzzy sets E_i, δE_j and U_{ij} correspond to the fuzzy sets determined by the indexes i and j (i = 1 to 5 and j = 1 to 5). For example, if i = 2 and j = 3 then $E_i \equiv$ NS, $\delta E_j \equiv$ ZE and $U_{ij} \equiv$ NS as seen in Table 4.2.

The implied output fuzzy set for each rule can be expressed as

$$\mu_{ij}(u) = \min\left\{\mu_{ij}^{CR}, \mu_{ij}^{U}(u)\right\} \tag{4.6}$$

where

$$\mu_{ij}^{CR} = \mu_i^E(e^*) \cdot \mu_j^{\delta E}(\delta e^*) \tag{4.7}$$

which is the certainty of the rule specified by i and j. $\mu_i^E(\cdot)$, $\mu_j^{\delta E}(\cdot)$ and $\mu_{ij}^U(u)$ are the membership functions corresponding to the fuzzy sets E_i, δE_j and U_{ij} respectively. Note that e^* and δe^* are the numerical input values and thus $\mu_i^E(e^*)$ and $\mu_j^{\delta E}(\delta e^*)$ are the *membership degrees* of e^* and δe^* in the fuzzy sets E_i and δE_j respectively.

Using the aggregation method max , the resultant output fuzzy set can be expressed as

$$\mu_{out}(u) = \max\{\mu_{11}(u), \mu_{12}(u), \ldots\ldots, \mu_{54}(u), \mu_{55}(u)\} \tag{4.8}$$

Let us again consider the numerical input values e = 30 and δe = -15 (e^* = 30 and δe^* = -15). The fuzzification results for these numerical input values are

$\mu_3^E(30) = 0.7$; $\mu_4^E(30) = 0.3$ and $\mu_i^E(30) = 0$ for i = 1,2,5.

$\mu_1^{\delta E}(-15) = 0.5$; $\mu_2^{\delta E}(-15) = 0.5$ and $\mu_j^{\delta E}(-15) = 0$ for j = 3,4,5.

If (4.7) is used then the certainty of the rules becomes

$\mu_{31}^{CR} = 0.35$; $\mu_{32}^{CR} = 0.35$; $\mu_{41}^{CR} = 0.15$; $\mu_{42}^{CR} = 0.15$

$\mu_{ij}^{CR} = 0$ if $(i,j) \neq \{(3,1) ; (3,2) ; (4,1) ; (4,2)\}$.

Thus, the implied output fuzzy sets can be expressed as

$\mu_{31}(u) = \min\{0.35, \mu_{31}^U(u)\}$; $\mu_{32}(u) = \min\{0.35, \mu_{32}^U(u)\}$

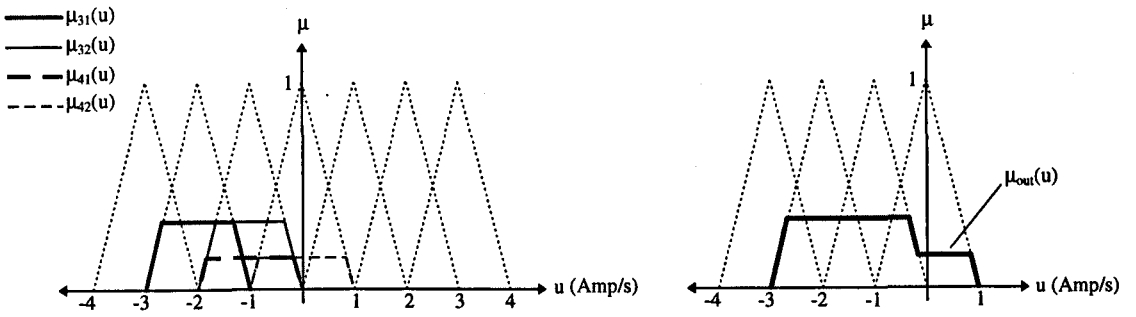
$\mu_{41}(u) = \min\{0.15, \mu_{41}^U(u)\}$; $\mu_{42}(u) = \min\{0.15, \mu_{42}^U(u)\}$

$\mu_{ij}(u) = 0$ if $(i,j) \neq \{(3,1) ; (3,2) ; (4,1) ; (4,2)\}$.

The implied output fuzzy sets and the aggregation of these sets (i.e. the resultant output fuzzy set) are shown in Fig.4.5. Note that the resultant output fuzzy set shown in Fig.4.5b is obtained using (4.8) as

$$\mu_{out}(u) = \max\{\mu_{31}(u), \mu_{32}(u), \mu_{41}(u), \mu_{42}(u)\}$$

since $\mu_{ij}(u) = 0$ for $(i,j) \neq \{(3,1) ; (3,2) ; (4,1) ; (4,2)\}$. This implies that only the rules for {i = 3, 4 and j = 1, 2} are used to obtain the resultant output fuzzy set. The other 21 rules have no effect on the controller output for this numerical input values. The controller output is calculated after the defuzzification of the resultant output fuzzy set.



(a) (b)

Figure 4.5

(a) The implied output fuzzy sets

(b) The resultant output fuzzy set $\mu_{out}(u)$

DEFUZZIFIER :

In the defuzzification process, the resultant output fuzzy set is defuzzified to obtain a numerical (crisp) controller output. The most popular defuzzification method is the Center of Gravity (COG) method [37,40] which specifies the crisp controller output (u^*) as the center of the area covered by the membership function $\mu_{out}(u)$, that is

$$u^* = \frac{\int u\mu_{out}(u)du}{\int \mu_{out}(u)du} \tag{4.9}$$

However, the integral calculations in the COG method are not computationally easy because $\mu_{out}(u)$ is usually irregular. Since the resultant output fuzzy set is the union of the implied output fuzzy sets, a good approximation of (4.9) may be the weighted average of the centers of the implied output fuzzy sets where the weights equal the heights of the corresponding implied output fuzzy sets [40]. Let u_n be the center of the n^{th} implied output fuzzy set and w_n be its height; the center average defuzzifier determines u^* as

$$u^* = \frac{\sum_{n=1}^{M} w_n u_n}{\sum_{n=1}^{M} w_n} \tag{4.10}$$

where M is the total number of the rules in the rule-base. The center average defuzzification method is one of the most widely used defuzzification method in practical fuzzy control applications because it is computationally very simple. In this method, the aggregation process is not required since the crisp output u^* is calculated directly by using the implied output fuzzy sets, and the weight w_n in (4.10) is actually the certainty of the corresponding rule. It should be noted that if the center average defuzzifier is used, then the shape of the output fuzzy sets becomes unimportant because only the centers of the output fuzzy sets are used in the calculations.

Let us again consider the above example (e = 30, δe = -15). If the center average defuzzifier is used then the output of the controller is calculated as (see Fig.4.5a)

$$u^* = \frac{0.35*(-2)+0.35*(-1)+0.15*(-1)+0.15*0}{0.35+0.35+0.15+0.15} = -1.2$$

The FC introduced in this section is called *Mamdani* type controller. Another commonly used FC type is the *Sugeno* type controller which differs from the Mamdani type controller in the format of the IF-THEN rules. For example, a Sugeno type controller has the rules in the form of

IF x_1 is A **AND** x_2 is B **THEN** $y = g(x_1, x_2)$

where g(.) may be a linear or a non-linear function of the input variables. Defuzzifier does not exist in the Sugeno type controller because the THEN part is an algebraic equation. The overall output is usually calculated using the weighted average of the output of each rule. The Sugeno type controller is especially appropriate for interpolating between different control laws.

In the following section, the equivalence between fuzzy and classical controllers will be considered for both Mamdani and Sugeno type fuzzy controllers.

4.3 Fuzzy Equivalence of a Second Order Linear Discrete Controller

The fuzzy equivalence of a linear discrete PI controller has been considered by Galichet and Foulloy in [56]. In this section, the fuzzy equivalence of a second order linear discrete controller which has a transfer function

$$G_c(z) = \frac{u(z)}{e(z)} = \frac{K_c(z-a)(z-b)}{(z-1)(z-c)} \tag{4.11}$$

will be considered since (4.11) can be easily converted to a PI, PD or PID controller if the parameters b and c are chosen properly. For example, if c is set to zero then (4.11) becomes a representation of a PID controller, if b and c are set to zero then it becomes a PI controller. If b is set to 1 and c is set to zero then (4.11) becomes a representation of a PD controller. Note that (4.11) is originally a PI+lead controller (if c < b) and it can also be converted to a lead or lag controller if the parameter a is set to 1, and b and c are chosen according to the desired lead or lag compensation. The fuzzy equivalence of a general (n-poles and m-zeros) linear discrete controller will be derived in Section 4.5.

In the discrete time domain, the output of the controller (4.11) can be written as

$$u(k) = u(k-1) + \delta u(k) \tag{4.12}$$

where

$$\delta u(k) = c\delta u(k-1) + \alpha_1 e(k) + \alpha_2 \delta e(k) + \alpha_3 \delta e(k-1) \tag{4.13}$$

The constants α_1, α_2 and α_3 are given as

$$\begin{aligned} \alpha_1 &= K_c\left(1-(a+b)+ab\right) \\ \alpha_2 &= K_c\left(a+b-ab\right) \\ \alpha_3 &= -abK_c \end{aligned} \tag{4.14}$$

and the δ operator is defined as

$$\delta x(k) = x(k) - x(k-1) \tag{4.15}$$

In the following subsections, the fuzzy equivalence of the control law u(k) given by (4.12) will be considered; however, the output of the FC will be δu(k) rather than u(k) because, in many practical applications, the actuating signal u(k) should be limited (to protect the electronic circuits) with an anti-windup mechanism which stops the integration in the controller. The anti-

windup mechanism can be easily implemented in (4.12) if $\delta u(k)$ is chosen as the output of the FC (addition ,or integration, is stopped when u(k) reaches the saturation limit, i.e., $\delta u(k)$ is not added to the previous value u(k-1) during the saturation). Note that u(k) is the numerical integration of $\delta u(k)$ as seen in (4.12) which can be easily implemented outside the controller to obtain the actuating signal u(k).

4.3.1 Sugeno Type Fuzzy Equivalence

In this section, the main purpose is to design a Sugeno type FC which is precisely equivalent to the controller given by (4.11). As mentioned in Section 4.2, a Sugeno type FC has a rule-base consisting of the rules in the form of

IF x_1 is A_1 **AND** x_2 is A_2 **AND**.....**AND** x_n is A_n **THEN** $y = g(x_1,x_2,....,x_n)$.

Since the output y is a function of the input variables, any control law can be directly implemented by choosing the output y as the desired control law if the membership functions of the input variables are chosen so that they provide a linear mapping between the inputs and the output of the controller.

As mentioned in Section 4.3, the output of the FC will be $\delta u(k)$ and u(k) will be obtained by using (4.12). Thus from (4.13), the inputs to the FC become $\delta u(k-1)$, e(k), $\delta e(k)$ and $\delta e(k-1)$. In order to keep the FC as simple as possible and to have a linear mapping, the membership functions for the input variables are chosen as shown in Fig 4.6, where v_i ,for i = 1 to 4, represents the input variables $\delta u(k-1)$, e(k), $\delta e(k)$ and $\delta e(k-1)$ respectively. It should be noted that the input variables are assumed to be bounded and M_i is the maximum value that the magnitude of the corresponding input variable can take on. In other words, M_i determines the limits of the universe of discourse for the corresponding input variable.

The Sugeno type FC will have $2^4 = 16$ rules since there are 4 input variables and 2 membership functions for each input variable. The rules can be represented in a general form as

IF $\delta u(k-1)$ is A_{1m} **AND** e(k) is A_{2n} **AND** $\delta e(k)$ is A_{3p} **AND** $\delta e(k-1)$ is A_{4q}

THEN $\delta u_{SU}(k) = c\delta u(k-1) + \alpha_1 e(k) + \alpha_2 \delta e(k) + \alpha_3 \delta e(k-1)$

where the indices {m,n,p,q} = {1,2} and α_1, α_2 and α_3 are given by (4.14).

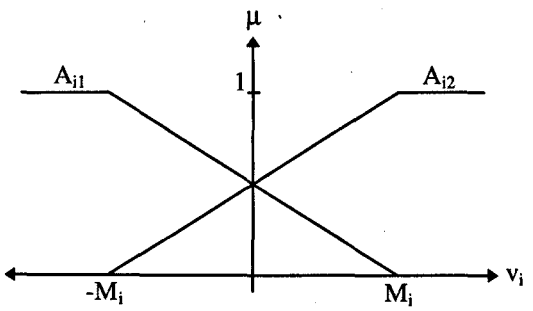


Figure 4.6 Membership functions for the input variables (i = 1,..,4)

Example 4.1 : Consider the system shown in Fig.4.7 :

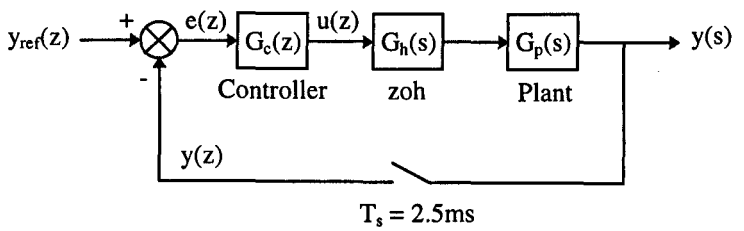


Figure 4.7 The control system block diagram

The transfer functions of the plant, zero-order-hold (zoh) and the controller are given as

$$G_p(s) = \frac{100}{(s+5)(s+10)} \tag{4.16}$$

$$G_h(s) = \frac{1-e^{-T_s s}}{s} \tag{4.17}$$

$$G_c(z) = \frac{2(z-0.9876)(z-0.95)}{(z-1)(z-0.88)} \tag{4.18}$$

Now the aim is to design a Sugeno type FC which will be precisely equivalent to $G_c(z)$ and thus give exactly the same closed loop responses as the system shown in Fig.4.7.

The input variables are $\delta u(k-1)$, $e(k)$, $\delta e(k)$ and $\delta e(k-1)$. Their membership functions are chosen as shown in Fig.4.6, where M_i is selected as 250 for i = 1 to 4 (i.e. for all the input variables). Note that the value of M_i can be chosen arbitrarily high because, as long as the magnitude of the input values do not exceed the corresponding M_i, the equivalence between $G_c(z)$ and the FC will be valid. However, in most of the practical applications, the reference input and the control signal u(k) are usually limited. Hence, all the input variables of the controller are bounded due to these limitations. For example, assume that y_{ref} is a step demand and limited as $|y_{ref}| \le Y_{max}$. If the worst case is considered and $|y(0)| \le Y_{max}$, then for a non-minimum phase stable system the limit for e(k) becomes (note that $e(k) = y_{ref} - y(k)$)

$$|e(k)| \le 2Y_{max}$$

and thus $\delta e(k)$ and $\delta e(k-1)$ are limited as

$$|\delta e| \le 4Y_{max}.$$

If the controller output u(k) is limited by

$$|u(k)| \le U_{max}$$

then $\delta u(k-1)$ is bounded as

$$|\delta u(k-1)| \le 2U_{max}$$

thus, the bounds of the input variables are $M_1 = 2U_{max}$, $M_2 = 2Y_{max}$, $M_3 = M_4 = 4Y_{max}$.

The number of rules are $2^4 = 16$ (there are 4 input variables and 2 membership functions for each input variable) and the rules can be represented in a general form as

IF $\delta u(k-1)$ is A_{1m} **AND** $e(k)$ is A_{2n} **AND** $\delta e(k)$ is A_{3p} **AND** $\delta e(k-1)$ is A_{4q}

THEN $\delta u_{SU}(k) = 0.88\delta u(k-1) + 0.00124e(k) + 1.99876\delta e(k) + 1.87644\delta e(k-1)$

where the indices $\{m,n,p,q\} = \{1,2\}$.

Fig.4.8 shows the simulation results for both $G_c(z)$ and the Sugeno type FC. The output response (y) and the control signal (u) are exactly same for both controllers as expected. The reference input (y_{ref}) is a unit step function applied at t = 0.1 s.

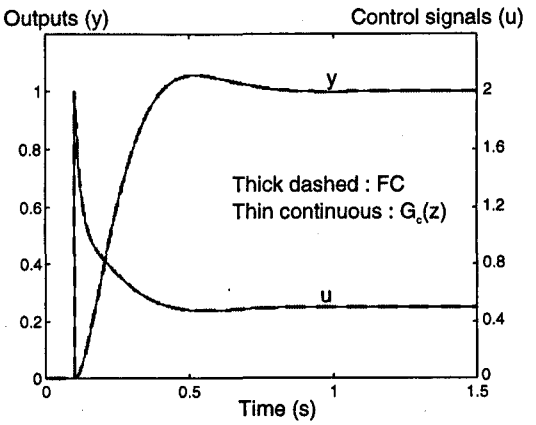


Figure 4.8 Simulation results showing the equivalence between $G_c(z)$ and the Sugeno type FC

It should be noted that the use of Sugeno type FC becomes more reasonable and meaningful when several control laws are to be implemented in a single controller rather than implementing only one control law. However, in this section, the main purpose was to illustrate the equivalence between a linear controller and a Sugeno type FC as a preliminary step for the implementation of several control laws in a FC. The implementation of two linear control laws in a single FC will be considered in Section 4.5 to show how to use the FLC for the control of non-linear deterministic systems.

4.3.2 Mamdani Type Fuzzy Equivalence

Mamdani type fuzzy equivalence of the linear controller (4.11) will be considered in this section. As discussed in Section 4.2, Mamdani type FCs do not have algebraic equations in the THEN part of the rules; rather they have output membership functions and there is a defuzzification process to produce a control output value. Therefore, the control law of the linear controller can not be directly used in the Mamdani type FCs. Since a linear control law is to be implemented, the inference operators (AND, implication and aggregation) and the defuzzification method should be chosen properly in order to not to lead to a non-linearity in the FC. It is possible to find different ways for implementing a linear control law in a FC, but

one of the simplest way is to chose the algebraic product for the AND and Implication operations and to use the center-average-defuzzification method. As discussed in Section 4.2, in the center-average-defuzzification method, the aggregation method is not required and the centers of the output membership functions are the quantity of interest, not the shapes of the membership functions. Therefore, the output membership functions can be simply chosen as singletons centred at the appropriate points as shown in Fig.4.9. It should be noted that the output membership functions do not have to be regularly distributed.

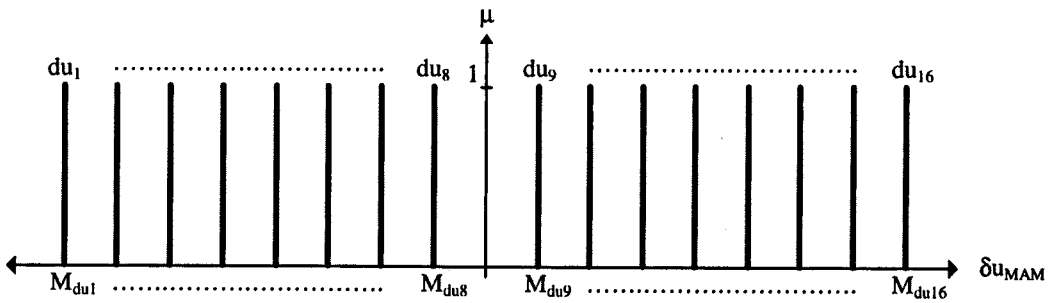


Figure 4.9 Output membership functions

The input membership functions should also not introduce any non-linearity. For example, if the input membership functions are chosen as shown in Fig.4.6, they not only provide a linear mapping but also result in the smallest possible rule-base (i.e. the number of the rules becomes minimum since there are only two membership functions for each input variable).

Using the input and output fuzzy sets shown in Fig.4.6 and Fig.4.9, the rule-base of the Mamdani type FC can be given in a tabular form as shown in Table 4.3. The rule-base consists of 16 rules since there are 4 inputs and 2 membership functions for each input variable.

In Table 4.3, the symbols e, δe, δu1 and δe1 represent the input variables e(k), δe(k), δu(k-1) and δe(k-1) respectively. The output is represented by δu_{MAM} and the symbols A_{i1}, A_{i2} (i = 1,..,4) and du_1, du_2,.....,du_{16} refer to the input and output membership functions shown in Fig.4.6 and Fig.4.9 respectively.

Table 4.3 The rule-base of the Mamdani type FC

δu_{MAM}	$\delta u1 = A_{11}$, $e = A_{21}$	
$\delta e1 \setminus \delta e$	A_{31}	A_{32}
A_{41}	du_1	du_2
A_{42}	du_3	du_4

δu_{MAM}	$\delta u1 = A_{11}$, $e = A_{22}$	
$\delta e1 \setminus \delta e$	A_{31}	A_{32}
A_{41}	du_5	du_6
A_{42}	du_7	du_8

δu_{MAM}	$\delta u1 = A_{12}$, $e = A_{21}$	
$\delta e1 \setminus \delta e$	A_{31}	A_{32}
A_{41}	du_9	du_{10}
A_{42}	du_{11}	du_{12}

δu_{MAM}	$\delta u1 = A_{12}$, $e = A_{22}$	
$\delta e1 \setminus \delta e$	A_{31}	A_{32}
A_{41}	du_{13}	du_{14}
A_{42}	du_{15}	du_{16}

Thus, the equivalence problem has been reduced to the determination of the values of M_1,..,M_4 and M_{du1}, M_{du2},.....M_{du16} for the membership functions of the input and output variables respectively. The selection of M_1,..,M_4 has been discussed for the Sugeno type FC in Section 4.3.1 and this discussion is also valid for the Mamdani type FC since there is no difference between Mamdani and Sugeno type FCs in terms of the input fuzzification process. On the other hand, the output membership function parameters M_{du1}, M_{du2},.....M_{du16} can be determined by using the desired linear control law, the rule-base and the extreme values of the input variables since the FC is expected to implement the desired linear control law between the extremes of the input variables using the rules in the rule-base. For example, from Table 4.3, consider the rule

IF $\delta u1$ is A_{11} **AND** e is A_{22} **AND** δe is A_{31} **AND** $\delta e1$ is A_{42} **THEN** δu_{MAM} is du_7

which implies that if the certainty of the rule is 1 (that means all the input values are full members of the corresponding fuzzy set, i.e. membership degree = 1, and thus the input values are the extremes), then the output fuzzy set du_7 should have a center at

$$M_{du7} = -cM_1 + \alpha_1 M_2 - \alpha_2 M_3 + \alpha_3 M_4 \tag{4.19}$$

to satisfy the equivalence between the controllers at these extreme values of the input variables. In this manner, the parameters $M_{du1},....,M_{du16}$ can be calculated as shown in Table 4.4.

Table 4.4 The centers of the output membership functions

$M_{du1} = -cM_1 - \alpha_1M_2 - \alpha_2M_3 - \alpha_3M_4$	$M_{du9} = -M_{du8}$
$M_{du2} = -cM_1 - \alpha_1M_2 + \alpha_2M_3 - \alpha_3M_4$	$M_{du10} = -M_{du7}$
$M_{du3} = -cM_1 - \alpha_1M_2 - \alpha_2M_3 + \alpha_3M_4$	$M_{du11} = -M_{du6}$
$M_{du4} = -cM_1 - \alpha_1M_2 + \alpha_2M_3 + \alpha_3M_4$	$M_{du12} = -M_{du5}$
$M_{du5} = -cM_1 + \alpha_1M_2 - \alpha_2M_3 - \alpha_3M_4$	$M_{du13} = -M_{du4}$
$M_{du6} = -cM_1 + \alpha_1M_2 + \alpha_2M_3 - \alpha_3M_4$	$M_{du14} = -M_{du3}$
$M_{du7} = -cM_1 + \alpha_1M_2 - \alpha_2M_3 + \alpha_3M_4$	$M_{du15} = -M_{du2}$
$M_{du8} = -cM_1 + \alpha_1M_2 + \alpha_2M_3 + \alpha_3M_4$	$M_{du16} = -M_{du1}$

Thus, if the centers of the output membership functions are chosen as shown in Table 4.4, the FC will provide the desired linear control behaviour by implementing a linear interpolation between the output values corresponding to the extremes of the input variables.

Example 4.2 : Let us again consider the system shown in Fig.4.7 with the plant and the linear controller given by (4.16) and (4.18) respectively. The aim is to design a Mamdani type controller which is precisely equivalent to the linear controller given by (4.18).

The input membership functions are chosen as shown in Example 4.1 since there is no difference between Mamdani and Sugeno type controllers in terms of the fuzzification process. Therefore, the membership functions are as shown in Fig.4.6, where $M_i = 250$ (for i = 1 to 4) for all the input variables (the selection of M_i has been already discussed in Example 4.1). The output membership functions are chosen as shown in Fig.4.9 and thus the rule base is as shown in Table 4.3.

By comparing (4.11) and (4.18), the linear controller parameters become

$K_c = 2$, $a = 0.9876$, $b = 0.95$ and $c = 0.88$.

Using (4.14), the constants α_1, α_2 and α_3 are calculated as

$\alpha_1 = 0.00124$, $\alpha_2 = 1.99876$ and $\alpha_3 = -1.87644$.

The centers of the output membership functions are obtained by using Table 4.4 as shown in Table 4.5.

Table 4.5 The centers of the output membership functions

$M_{du1} = -250.89$	$M_{du9} = 189.11$
$M_{du2} = 748.49$	$M_{du10} = 1188.49$
$M_{du3} = -1189.11$	$M_{du11} = -749.11$
$M_{du4} = -189.73$	$M_{du12} = 250.27$
$M_{du5} = -250.27$	$M_{du13} = 189.73$
$M_{du6} = 749.11$	$M_{du14} = 1189.11$
$M_{du7} = -1188.49$	$M_{du15} = -748.49$
$M_{du8} = -189.11$	$M_{du16} = 250.89$

Fig.4.10 shows the simulation results for the designed Mamdani type FC in comparison with the linear controller (4.18). The output response (y) and the control signal (u) are exactly same for both controllers as expected. The reference input (y_{ref}) is a unit step function applied at t = 0.1s.

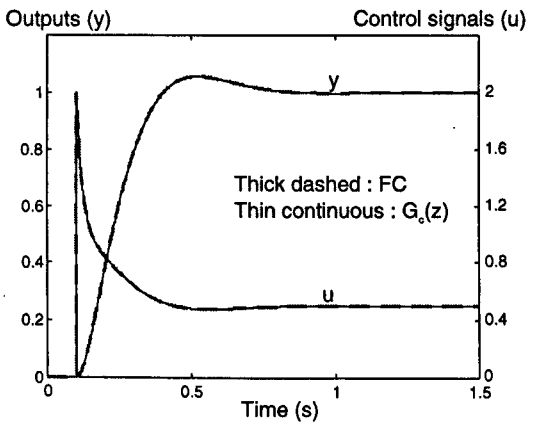


Figure 4.10 Simulation results showing the equivalence between $G_c(z)$ and the Mamdani type FC

4.4 Fuzzy Equivalence of a General Linear Discrete Controller and Its Implementation using Hierarchical Fuzzy Control

In the previous section, the fuzzy equivalence of a second order linear controller has been considered. There were 4 inputs to the FC and 2 membership functions were defined for each input variable. Thus, the number of rules was $2^4 = 16$ which implies that the number of rules increases exponentially with the number of inputs.

Consider a general linear discrete controller with m zeros and n poles ($m \leq n$) :

$$G_c(z) = \frac{u(z)}{e(z)} = \frac{K_c(z-z_1)(z-z_2).....(z-z_m)}{(z-p_1)(z-p_2).....(z-p_n)} \tag{4.20}$$

which can be written as

$$G_c(z) = \frac{u(z)}{e(z)} = \frac{a_0 + a_1 z^{-1} + ... + a_n z^{-n}}{1 + b_1 z^{-1} + ... + b_n z^{-n}} \tag{4.21}$$

where the coefficients a_i and b_i may be zero so that (4.21) can represent the general controller transfer function given by (4.20). From (4.21), the control law can be derived as

$$u(k) = \sum_{i=0}^{n} a_i e(k-i) - \sum_{j=1}^{n} b_j u(k-j) \tag{4.22}$$

which can be rewritten in a more compact form as

$$u(k) = \sum_{i=0}^{2n} c_i x_i \tag{4.23}$$

where

$$c_i = \begin{cases} a_i & for \;\; i = 0 \;\; to \;\; n \\ -b_{i-n} & for \;\; i = n+1 \;\; to \;\; 2n \end{cases} \tag{4.24}$$

and

$$x_i = \begin{cases} e(k-i) & for \;\; i=0 \;\; to \;\; n \\ u(k-(i-n)) & for \;\; i=n+1 \;\; to \;\; 2n \end{cases} \tag{4.25}$$

Note that in the control law (4.23), c_i may be zero depending on the zeros and the poles of the controller given by (4.20). Suppose the control law (4.23) has N non-zero terms (i.e. $c_i \neq 0$), then (4.23) can be rewritten as

$$u(k) = \sum_{j=1}^{N} \alpha_j v_j \tag{4.26}$$

where $\alpha_j = c_i$ and $v_j = x_i$ if $c_i \neq 0$ for $i = 0$ to $2n$ and $j = 1$ to N. In other words, (4.26) is precisely equivalent to (4.23) but it contains only non-zero terms of (4.23). Hence, (4.26) is the linear control law which will be implemented in the FC.

It is seen from (4.26) that there are N inputs to the FC. Suppose M membership functions are defined for each input variable, then the number of rules becomes M^N. For a large N, M^N is a huge number. For example, assume that a linear controller has 5 zeros and 5 poles, and $c_i \neq 0$ for i = 0 to 2n in the control law (4.23). Thus, the number of inputs to the FC becomes 11. This means, even if 2 (minimum) membership functions are defined for the input variables, the number of rules becomes $2^{11} = 2048$. It is impractical to implement a FC with thousands of rules. Therefore, a mechanism is required to reduce the number of the rules in the FC. The *Hierarchical Fuzzy Control* aims to reduce the number of rules in a FC by constructing the FC in a cascade form [40]. Fig.4.11 shows the implementation of the control law (4.26) using a Hierarchical Fuzzy Controller (HFC) which contains N-1 sub-FCs having only 2 inputs and 1 output each.

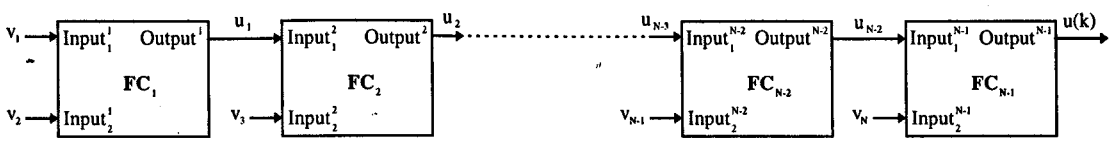


Figure 4.11 Hierarchical Fuzzy Controller (HFC)

Fig.4.11 implies that if M membership functions are defined for each input variable then the total number of rules in the HFC becomes

$$R_{HFC} = M^2(N-1) \tag{4.27}$$

For example, let us consider again 11 inputs (N = 11) with 2 membership functions (M = 2) for each input variable. Thus, R_{HFC} becomes 40 which is considerably less than 2048 that is the number of rules required for the normal FC implementation.

In Fig.4.11, the sub-controllers denoted as FC_1, FC_2,...FC_{N-1} can be implemented as either in Sugeno or Mamdani type. Let us first consider the Sugeno type implementation :

The input membership functions defined for the sub-FCs should provide a linear mapping between their inputs and the outputs since they will implement the linear control law (4.26) by piecewise linear construction. Hence, the input membership functions are chosen as shown in Fig.4.12, where q refers to a sub-FC shown in Fig.4.11, p refers to one of the inputs of this FC and M_p^q is the maximum value that the magnitude of the corresponding input variable can take on.

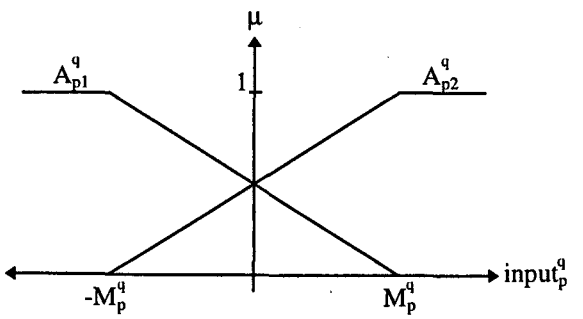


Figure 4.12 Membership functions of the input variables for the HFC
(p = 1,2 and q = 1,..., N-1)

Thus, the rules for the HFC can be written in a general form as

IF $input_1^q$ is A_{1i}^q **AND** $input_2^q$ is A_{2j}^q **THEN** $output^q = u_q$

where i =1,2 ; j = 1,2 ; q = 1,....,N-1 and the output functions are given as

$$u_q = \begin{cases} \alpha_q v_q + \alpha_{q+1} v_{q+1} & for \quad q = 1 \\ u_{q-1} + \alpha_{q+1} v_{q+1} & for \quad 2 \le q \le N-1 \end{cases} \tag{4.28}$$

Note that u_{N-1} is the output of the HFC and corresponds to u(k).

For the Mamdani type implementation, as discussed in Section 4.3, the desired linear control law can not be used in the rules of the FCs since the THEN part of the controller is not an algebraic equation. The rules can be represented in a general form as

IF $input_1^q$ is A_{1i}^q **AND** $input_2^q$ is A_{2j}^q **THEN** $output^q$ is U_{ij}^q

where i =1,2 ; j = 1,2 ; q = 1,....,N-1 and U_{ij}^q represents the corresponding output membership function. As discussed in Section 4.3, if the center -average defuzzifier is used and the algebraic product is chosen for the AND and Implication operations, then the FC will result in a linear interpolation between the output values determined using the extremes of the input variables. Fig.4.13 shows the membership functions for the outputs of the sub-FCs in the Mamdani type implementation of the HFC, where H_{ij}^q is the center of the corresponding output membership function.

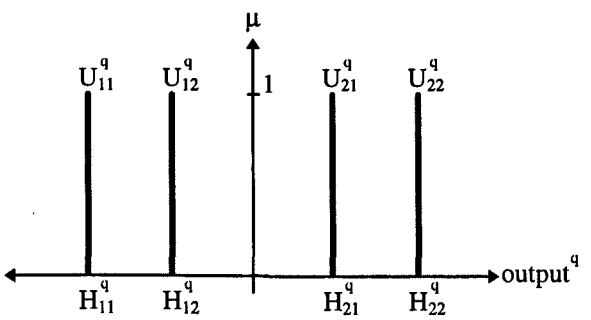


Figure 4.13 Membership functions for the outputs of the sub-FCs in the Mamdani type implementation of the HFC (q = 1,....,N-1)

The centers of the output membership functions are the crucial part of the Mamdani type design. In order to obtain the equivalence between the linear control law (4.26) and the HFC, all the FCs in the HFC should implement their own linear control law specified by (4.28). Hence, the centers H_{ij}^{q} should be calculated by considering the rule-base of each FC, extremes of the corresponding input variables and the individual linear control laws given by (4.28). Table 4.6 shows the rule-base of the FCs in the HFC, where q refers to the related FC in the HFC.

Table 4.6 Rule-base for the FCs in the HFC (q = 1,....,N-1)

$output^{q}$ \ $input_2^{q}$	A_{21}^{q}	A_{21}^{q}
A_{11}^{q}	U_{11}^{q}	U_{12}^{q}
A_{12}^{q}	U_{21}^{q}	U_{22}^{q}

($input_1^{q}$ labels the rows)

Thus, using (4.28), Table 4.6 and the extreme values of the input variables, the centers are calculated as

$$H_{ij}^{q} = \begin{cases} (-1)^{i}\alpha_{q}M_{1}^{q} + (-1)^{j}\alpha_{q+1}M_{2}^{q} & for \;\; q = 1 \\ (-1)^{i}M_{1}^{q} + (-1)^{j}\alpha_{q+1}M_{2}^{q} & for \;\; 2 \le q \le N-1 \end{cases} \tag{4.29}$$

where i = 1,2 and j = 1,2.

It should be noted that if the number of inputs is greater than 4 then it is reasonable to use the HFC instead of the normal FC. This is simply because 4 inputs lead 12 rules in HFC whilst the normal FC requires 16 rules (assuming 2 membership functions defined for each input variable). However, if there are 5 inputs, the HFC has 16 rules and the normal FC has 32 rules. The difference obviously increases exponentially with the number of inputs.

Example 4.3 : Consider the linear controller given by

$$G_c(z) = \frac{0.2(z-0.1)(z+0.3)(z-0.5)(z-0.4+0.6j)(z-0.4-0.6j)}{(z-0.2)(z+0.6)(z-0.9)(z-0.6+0.7j)(z-0.6-0.7j)} \tag{4.30}$$

which can be written as

$$G_c(z) = \frac{u(z)}{e(z)} = \frac{0.2-0.22z^{-1}+0.126z^{-2}-0.0074z^{-3}-0.0159z^{-4}+0.0016z^{-5}}{1-1.7z^{-1}+0.97z^{-2}+0.259z^{-3}-0.5376z^{-4}+0.0918z^{-5}} \tag{4.31}$$

Thus, the control law becomes

$$u(k) = \sum_{j=1}^{11} \alpha_j v_j \tag{4.32}$$

where

$$v_j = \begin{cases} e(k-(j-1)) & for \quad j=1 \ \ to \ \ 6 \\ u(k-(j-6)) & for \quad j=7 \ \ to \ \ 11 \end{cases} \tag{4.33}$$

and the coefficients of the control law (α_j) are given in Table 4.7.

Table 4.7 The coefficients of the control law (4.32)

$\alpha_1 = 0.2$	$\alpha_7 = 1.7$
$\alpha_2 = -0.22$	$\alpha_8 = -0.97$
$\alpha_3 = 0.126$	$\alpha_9 = -0.259$
$\alpha_4 = -0.0074$	$\alpha_{10} = 0.5376$
$\alpha_5 = -0.0159$	$\alpha_{11} = -0.0918$
$\alpha_6 = 0.0016$	

The HFC equivalence of the controller (4.30) is implemented as shown in Fig.4.11, where N = 11. As stated before, the sub-FCs in the HFC can be designed as Sugeno or Mamdani type. Let us first consider the Sugeno type implementation.

a) *Sugeno type implementation* : The membership functions for the input variables are chosen as shown in Fig.4.12, where, for simplicity, M_p^q is chosen as 10 for all the input variables of the HFC (p = 1,2 and q = 1,...,10). As discussed in Section 4.3.1, the M_p^q values can be chosen arbitrarily large. However, if the error e(k) and the controller output u(k) are bounded (usually this is the case for most of the practical applications), then the bounds of all the input variables can be estimated using (4.33) and (4.34) in a similar way illustrated in Section 4.3.1.

The rules of the HFC are

IF $input_1^q$ is A_{1i}^q **AND** $input_2^q$ is A_{2j}^q **THEN** $output^q = u_q$

where i =1,2 ; j = 1,2 ; q = 1,....,10 and the output functions are given as

$$u_q = \begin{cases} \alpha_q v_q + \alpha_{q+1} v_{q+1} & for \quad q = 1 \\ u_{q-1} + \alpha_{q+1} v_{q+1} & for \quad 2 \le q \le 10 \end{cases} \tag{4.34}$$

Note that the α coefficients are given in Table 4.7 and input variables (v_q) are specified by (4.33).

Fig.4.14a shows the simulated unit step responses of the designed HFC and the linear controller (4.30). The output responses are also shown for an sinusoidal input in Fig.4.14b. As seen in the figures, the outputs of the controllers are identical as expected.

b) *Mamdani type implementation* : The input membership functions are chosen exactly same with the ones used for the Sugeno type implementation above. The rules of the HFC based on the Mamdani type implementation are

IF $input_1^q$ is A_{1i}^q **AND** $input_2^q$ is A_{2j}^q **THEN** $output^q$ is U_{ij}^q

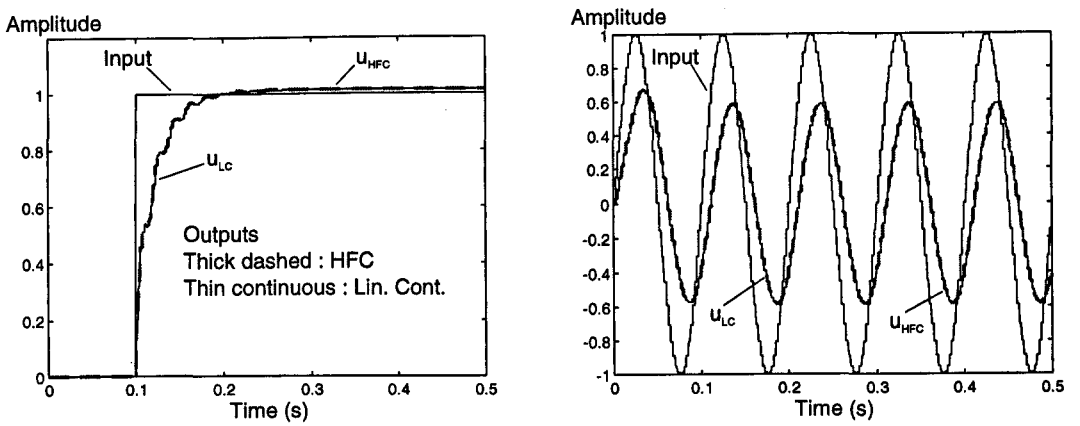


Figure 4.14 Controller output responses for

(a) unit step input

(b) a sinusoidal input (10Hz)

(u_{HFC} : output of the Sugeno type HFC, u_{LC} : output of the linear controller (4.30))

where i =1,2 ; j = 1,2 ; q = 1,....,N-1 and U_{ij}^{q} are the output membership functions shown in Fig.4.13. The centers of the output membership functions (H_{ij}^{q}) are calculated by using (4.29) and given in Table 4.8.

Table 4.8 The numerical values of H_{ij}^{q}

ij \ q	1	2	3	4	5	6	7	8	9	10
11	0.2	-11.26	-9.926	-9.841	-10.016	-27	-0.3	-7.41	-15.376	-9.082
12	-4.2	-8.74	-10.074	-10.159	-9.984	7	-19.7	-12.59	-4.624	-10.918
21	4.2	8.74	10.074	10.159	9.984	-7	19.7	12.59	4.624	10.918
22	-0.2	11.26	9.926	9.841	10.016	27	0.3	7.41	15.376	9.082

Fig.4.15a and b show the simulated unit step and sinusoidal input responses respectively for the controller given by (4.30) and the HFC based on the Mamdani type implementation. As expected, the output responses are identical for both input functions as shown in Fig.4.15.

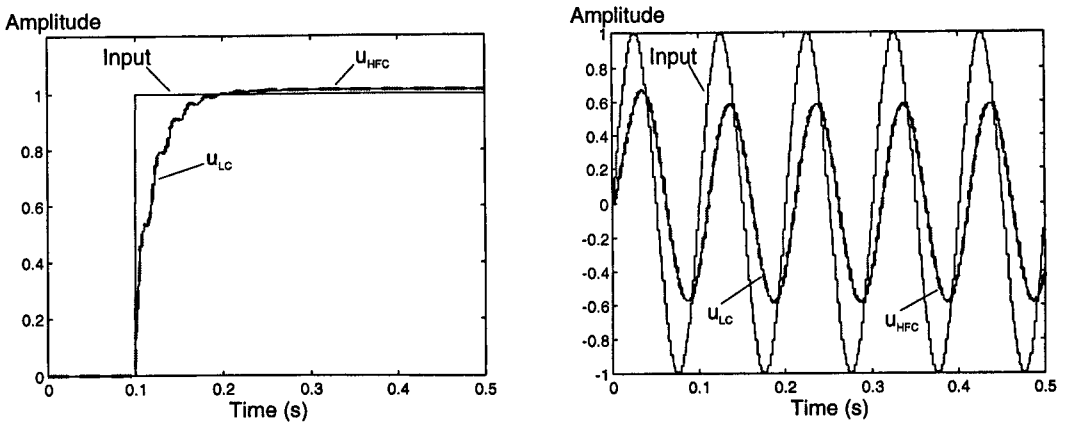


Figure 4.15 Controller output responses for

(a) unit step input

(b) a sinusoidal input (10Hz)

(u_{HFC} : output of the Mamdani type HFC, u_{LC} : output of the linear controller (4.30))

4.5 Fuzzy Controller Design for a Class of Non-linear Deterministic Systems

The FCs are generally preferred when the linear controllers are not able to control the system with a satisfactory performance. If the system is non-linear, the linear controllers usually do not show a good control performance. However, let us consider a non-linear system which can be satisfactorily controlled by different linear controllers at different operating points (i.e., one controller is good for one operating point and another controller is good for another operating point). In this section, an example will be considered to show how to design a FC for this kind of non-linear systems.

Example 4.4 : Consider the non-linear system represented by

$$\begin{aligned} \dot{x}_1 &= -(2+x_2)x_1 - x_2^2 + 100u \\ \dot{x}_2 &= x_1 \\ y &= x_2 \end{aligned} \tag{4.35}$$

where x_1, x_2 are the state variables, u is the control input and y is the output of the system. If the small signal linearization [58] is used, about the operating point $\underline{x}^o$, the system can be represented as

$$\frac{\Delta y}{\Delta u} = \frac{100}{(s+2)(s+y^o)} \tag{4.36}$$

where y^o is the output at the operating point. Obviously, (4.36) implies that the system has a pole changing with the operating point. Fig.4.16 shows the closed loop control of the non-linear system represented by (4.35). Note that y_{ref} actually corresponds to y^o since y_{ref} is the desired output operating point.

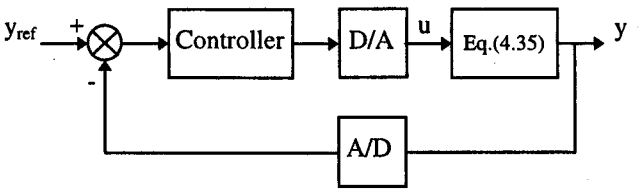


Figure 4.16 Closed loop control of the non-linear system represented by (4.35)

Assume that the operating region is defined as $2 \leq y_{ref} \leq 20$. If a linear controller is designed for a certain operating point, it may not give an satisfactory output response for other operating points in the operating region because the system has a pole changing with the operating point. Therefore, a controller is required which will choose the right control action according to the operating point.

The linearized transfer functions of the non-linear system (about the operating points) at the extremes of the operating region are

$$\begin{aligned} G_{p1}(s) &= \frac{100}{(s+2)(s+2)} \quad (y_{ref} = 2) \\ G_{p2}(s) &= \frac{100}{(s+2)(s+20)} \quad (y_{ref} = 20) \end{aligned} \tag{4.37}$$

Suppose we have two linear controllers given by

$$G_{c1}(z) = \frac{30(z-0.9948)(z-0.99745)}{(z-1)(z-0.25)}$$

$$G_{c2}(z) = \frac{4(z-0.857)(z-0.995)}{(z-1)(z-0.25)} \tag{4.38}$$

which are designed for the extremes of the operating region to have an output response without overshoot and a settling time (1%) less than 0.5s. However, $G_{c1}(z)$ can control the system satisfactorily only if y_{ref} is close to 2 and $G_{c2}(z)$ can control the system properly only if y_{ref} is close to the other extreme 20.

Now, the purpose is to design a FC (or HFC) which can control the system with a satisfactory performance over the entire operating region. This can be managed by implementing the control laws of (4.38) in a FC which will calculate the control action by interpolating between these control laws according to the operating point. The control laws of the controllers given by (4.38) can be written as

$$u_{cr}(k) = \sum_{j=1}^{5} \alpha_j^r v_j \tag{4.39}$$

where the index r = 1,2 and it refers to $G_{c1}(z)$ and $G_{c2}(z)$ respectively, and

$$v_j = \begin{cases} e(k-(j-1)) & for \quad j=1 \ \ to \ \ 3 \\ u(k-(j-3)) & for \quad j=4 \ \ to \ \ 5 \end{cases} \tag{4.40}$$

The coefficients α_j^r are given in Table 4.9.

Table 4.9 The numerical values of α_j^r

r \ j	1	2	3	4	5
1	30	-59.7675	29.7679	1.25	-0.25
2	4	-7.408	3.41086	1.25	-0.25

The FC will calculate the controller output according to the output operating point which is actually the value of y_{ref}. Therefore, y_{ref} should also be one of the inputs to the FC. Thus, there are totally 6 inputs to the FC and it is reasonable to use a HFC in order to reduce the total number of the rules in the controller. Fig.4.17 show the structure of the HFC which has 4 sub-FCs and each FC has 3 inputs. Note that the third input of each FC is y_{ref} since it is required to decide on the control action of each FC.

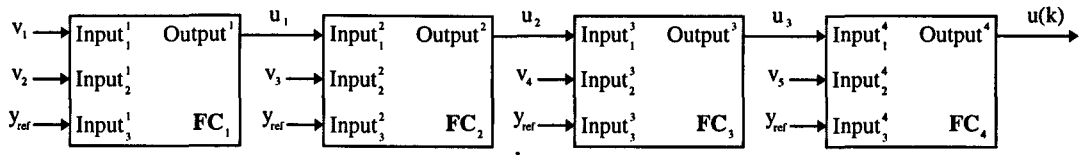


Figure 4.17 The structure of the HFC

As discussed in Section 4.4, the sub-FCs may be implemented either in Sugeno or Mamdani type. Let us first consider the Sugeno type implementation.

a) *Sugeno type implementation of the HFC* : The membership functions of the inputs $input_1^q$ and $input_2^q$ (q = 1,..,4) are chosen as shown in Fig.4.12, where, for simplicity, M_p^q is chosen as 1000 for all the input variables (The selection of M_p^q is discussed in the previous sections). Since the operating region is defined as $2 \leq y_{ref} \leq 20$, the membership functions of the input $input_3^q$ are chosen as shown in Fig.4.18.

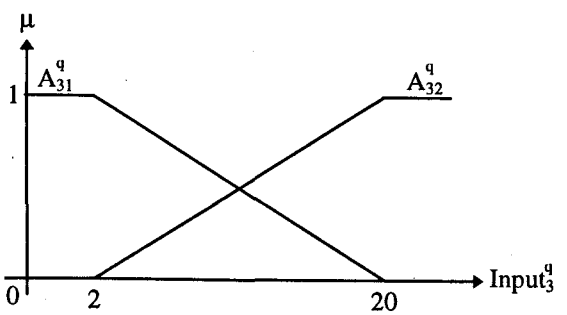


Figure 4.18 Membership functions of the input variable $input_3^q$ (q = 1,..,4)

The rules of the Sugeno type implementation can be given in a general form as

IF $input_1^q$ is A_{1i}^q **AND** $input_2^q$ is A_{2j}^q **AND** $input_3^q$ is A_{3r}^q **THEN** $output^q = f_q^r$

where {i, j, r} = {1,2} ; q = 1,..,4 and the output functions are defined as

$$f_q^r = \begin{cases} \alpha_q^r v_q + \alpha_{q+1}^r v_{q+1} & for \quad q = 1 \\ u_{q-1} + \alpha_{q+1}^r v_{q+1} & for \quad 2 \le q \le 4 \end{cases} \tag{4.41}$$

The closed loop system shown in Fig.4.16 is simulated and the results are shown in Fig.4.19 and 4.20 which illustrate the variation of the output (y) for different operating points (i.e., for different y_{ref} values) by using the designed HFC, $G_{c1}(z)$ and $G_{c2}(z)$.

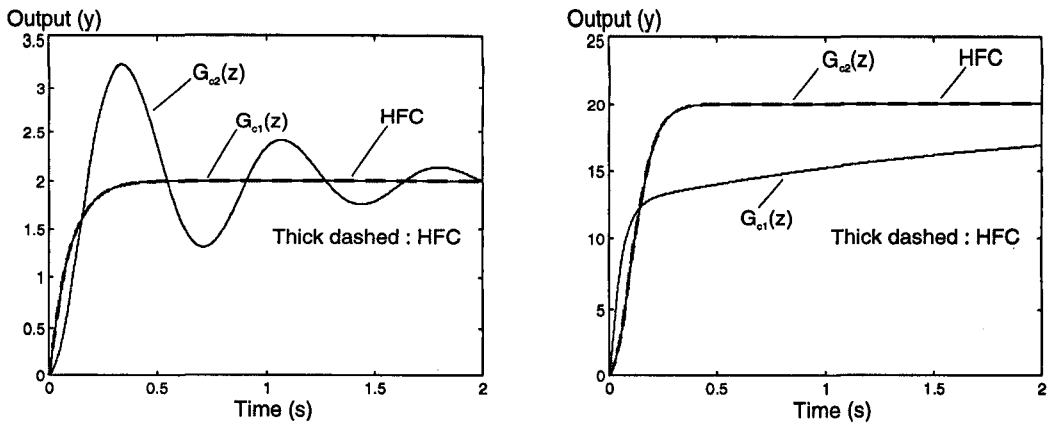


Figure 4.19 The output responses to a step demand for
(a) $y_{ref} = 2$ (b) $y_{ref} = 20$

As seen in Fig.4.19a, the HFC gives identical response with $G_{c1}(z)$ since, for $y_{ref} = 2$, the HFC is identical with $G_{c1}(z)$. However, $G_{c2}(z)$ results in a oscillatory output response for $y_{ref} = 2$. Similarly, the HFC gives identical response with $G_{c2}(z)$ for $y_{ref} = 20$ as seen in Fig.4.19b and $G_{c1}(z)$ gives an unsatisfactory response for $y_{ref} = 20$ as expected. On the other hand, Fig.4.20 shows variations of the output y for some operating points between the extremes of the operating region. As seen in Fig.4.20, the HFC gives reasonable responses for all the operating points whilst $G_{c1}(z)$ and $G_{c2}(z)$ do not give satisfactory results.

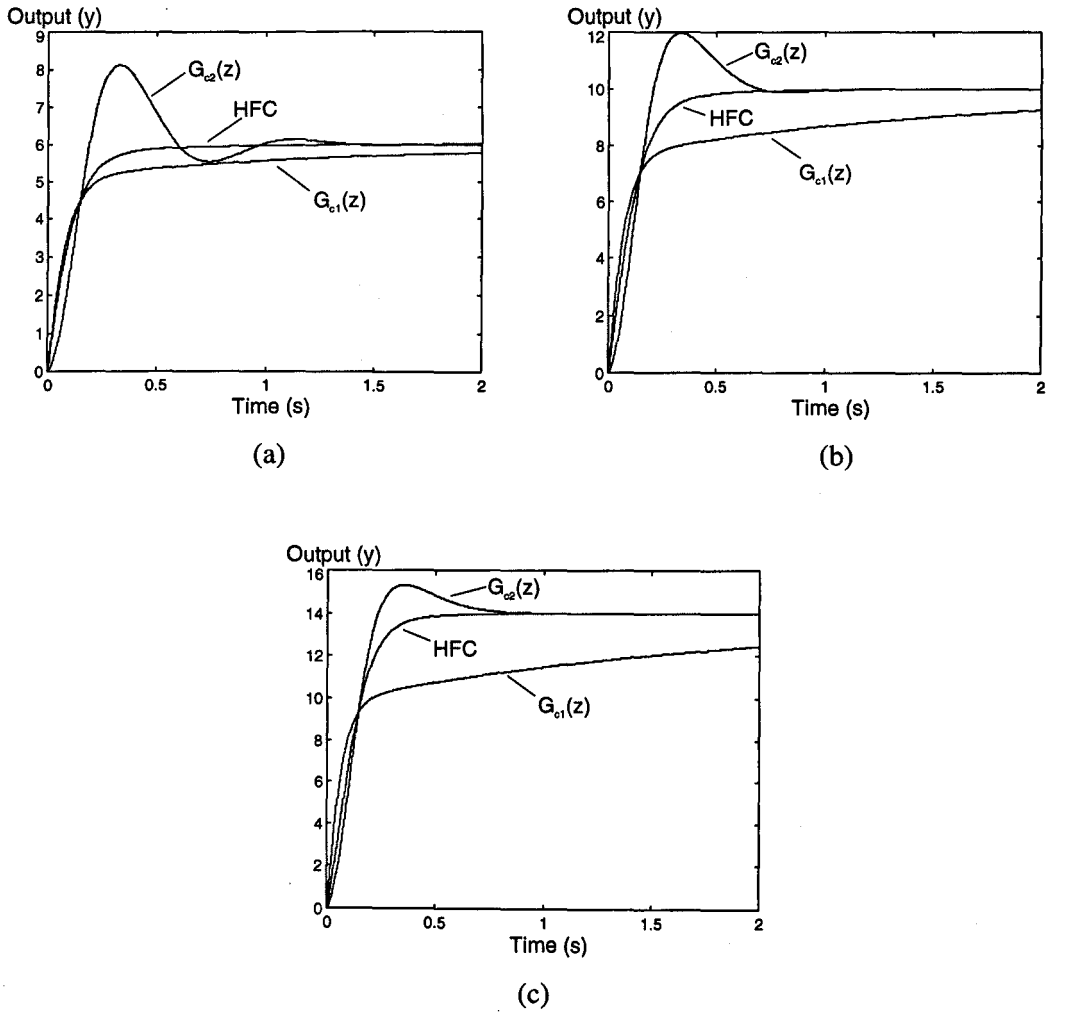


Figure 4.20 The output responses to a step demand for
(a) $y_{ref} = 6$ (b) $y_{ref} = 10$ (c) $y_{ref} = 14$

Thus, as seen in Fig.4.19 and 4.20, the designed HFC controls the system with a reasonable performance over the operating region.

b) *Mamdani type implementation of the HFC* : The input membership functions are chosen as exactly same with the ones used in the Sugeno type implementation. The rules of the Mamdani type implementation can be expressed in a general form as

IF $input_1^q$ is A_{1i}^q **AND** $input_2^q$ is A_{2j}^q **AND** $input_3^q$ is A_{3r}^q **THEN** $output^q$ is U_{ij}^{rq}

where {i, j, r} = {1,2} ; q = 1,..,4. The output membership functions U_{ij}^{rq} are shown in Fig.4.21.

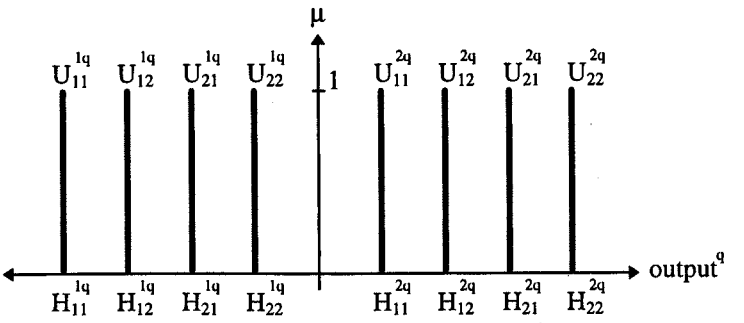


Figure 4.21 Membership functions of the output variables (q = 1,..,4)

The centers of the output membership functions are given as

$$H_{ij}^{rq} = \begin{cases} (-1)^i \alpha_q^r M_1^q + (-1)^j \alpha_{q+1}^r M_2^q & for \quad q = 1 \\ (-1)^i M_1^q + (-1)^j \alpha_{q+1}^r M_2^q & for \quad 2 \le q \le 4 \end{cases} \tag{4.42}$$

where {i, j, r} = {1,2}. The numerical values of H_{ij}^{rq} are calculated using (4.42) and given in Table 4.10.

Table 4.10 The numerical values of H_{ij}^{rq}

ijr \ q	1	2	3	4
111	29767.5	-30767.9	-2250	-750
121	-89767.5	28767.9	250	-1250
211	89767.5	-28767.9	-250	1250
221	-29767.5	30767.9	2250	750
112	3408	-4410.86	-2250	-750
122	-11408	2410.86	250	-1250
212	11408	-2410.86	-250	1250
222	-3408	4410.86	2250	750

The simulation results of the Mamdani type implementation are not presented since they are exactly same with the ones shown in Fig.4.19 and 4.20. This is simply because the Mamdani type implementation is precisely equivalent to the Sugeno type implementation.

4.6 Conclusions

In this chapter, the fuzzy equivalence of the linear controllers has been investigated. It has been shown that any linear control law can be precisely implemented in a FC for a given input universe of discourse. The equivalence may be interpreted as a sort of bridge between the classical and fuzzy control approaches. This is an important point because the equivalence may be used to combine the classical and the fuzzy control approaches in a same framework and thus a controller using the advantages of both control methods may be designed. For example, in Section 4.5, a robust FC is designed for a deterministic non-linear system by using two linear control laws previously designed for the extremes of the operating region.

In most practical applications however, the system (plant) is usually non-deterministic. In other words, the plant model or the parameters of the system are not known exactly and thus the direct information about the system's non-linearity and the variation of the parameters is not available. Hence, the method explained in Section 4.5 is not directly applicable for these sort of control problems. For example, consider the speed control of a motor drive system. Assume that the mechanical load is a simple inertia and friction and they may change within a range as $J_{min} \leq J \leq J_{max}$ and $B_{min} \leq B \leq B_{max}$. A satisfactory PI controller can be designed for any values of J and B if they are known. Suppose that four PI controllers are designed for the extreme values of J and B (i.e., for the cases $\{J_{min}, B_{min}\}$, $\{J_{min}, B_{max}\}$, $\{J_{max}, B_{min}\}$ and $\{J_{max}, B_{max}\}$). Now, we would like to design a FC which will interpolate between these PI control laws in order to give a satisfactory response for any value of J and B within their limits. Note that the FC needs information about the values of the inertia and friction in order to calculate an appropriate control action by interpolating between the PI control laws. The nominal inputs to the FC are the speed error and the change of error. Unfortunately, these input variables do not supply the necessary information about the values of the inertia and friction to use in the decision making mechanism of the FC. Thus, the FC can not choose the correct control action using these input variables. Therefore, a method is required to supply useful information about the parameter variations for the FC to calculate an appropriate control output.

In Chapter 5, a robust speed controller design based on the sliding mode and fuzzy control approaches will be considered. In this design approach, a reference model will be used to get information about the values of the inertia and friction and thus this information will be used as an input to the FC to choose a proper control action.

Chapter 5

Robust Speed Controller Design using Sliding Mode and Fuzzy Logic Control

5.1 Introduction

Sliding Mode Control (SMC), often called Variable Structure Control (VSC), is a powerful technique to control the non-linear and uncertain (non-deterministic) systems [35,59,60]. It is a robust control method and can be applied in the presence of model uncertainties, parameter fluctuations and external disturbances provided that the bounds of these uncertainties and disturbances are known. The SMC approach is probably the most popular method for the robust control of electrical drives whose mechanical loads are non-linear or change over a wide range [9-16]. The main disadvantage of the method is the assumption that the control signal can be switched from one value to another at infinite rate. In practical systems, however, it is impossible to manage this since the microprocessor implementation of the control strategy requires a finite sampling time (an analogue implementation of an SMC is conceivable, but even here delay or hysteresis effects would result in finite switching times). Direct microprocessor application of the SMC method results in a high frequency oscillation (chattering) about the desired equilibrium point. Although there may exist some applications in which this chattering may be utilised (e.g. direct production of PWM signals), it is generally undesirable since chattering excites the unmodeled high frequency dynamics of the systems. A significant research effort has been directed at eliminating or reducing the chattering [35].

The SMC method has also attracted attention in order to design a robust Fuzzy Controller (FC) for the non-linear and uncertain systems [37-41]. For second order systems, this approach to FC design divides the phase plane of error and change of error by a *switching line* and determines the magnitude of the control effort depending on the distance of the state vector from the

switching line [37]. The principle is similar to the SMC technique in which the system trajectory is forced to stay on a predetermined switching line. The Sliding Mode Fuzzy Controllers (SMFCs) will be briefly discussed in Section 5.3.1.

A new SMC technique called Reaching Law Control (RLC) has been introduced by Gao and Hung in [36]. This approach not only establishes a reaching condition to the sliding line (or surface) directly but also specifies the dynamic characteristics of the system during the reaching phase. Additional merits of the RLC approach include simplification of the solution for SMC and providing a measure for the reduction of chattering. Since the RLC approach is quite new and the classical SMC is a well known technique, there are only a few practical applications of the RLC approach to motor drive control systems [69,70].

The main objective of this chapter is to investigate the SMC approach for speed control systems and to develop a practical robust control design procedure using the SMC and the Fuzzy Logic Control (FLC) methods. Section 5.2 describes the basic idea of the SMC and discusses the chattering problem due the discrete time implementation of the SMC strategy. The SMC with Boundary Layer (BL), which is the most widely used chattering elimination method, is explained in Section 5.3. Section 5.4 summarises the SMC approaches to the speed control systems. The RLC technique is explained in Section 5.5 and used to develop a method to get information about the parameter variations and external disturbances. This information is then used as an input to the FC to take an appropriate control action in the case of parameter variations and external disturbances. The new robust control method based on the RLC and FLC approaches is described in Section 5.6.

5.2 Sliding Mode Control (SMC)

The basic idea of SMC was originally illustrated by a second order system as reported in [35]. Let us consider a single input second order linear uncertain system:

$$\begin{aligned}\dot{\underline{x}}(t) &= A\underline{x}(t) + \underline{b}u(t) + \underline{d}f(t) \\ &= (A_n + \Delta A)\underline{x}(t) + (\underline{b}_n + \Delta\underline{b})u(t) + (\underline{d}_n + \Delta\underline{d})f(t)\end{aligned} \tag{5.1}$$

where $\underline{x}(t)$ is the state vector, u(t) is the control input, A_n, $\underline{b}_n$ and $\underline{d}_n$ are composed of nominal system parameters, ΔA, $\Delta \underline{b}$ and $\Delta \underline{d}$ are the uncertainties introduced by unknown system parameters and f(t) is the external disturbance. ΔA, $\Delta \underline{b}$, $\Delta \underline{d}$ and f(t) are not known exactly but they are bounded. Equation (5.1) can also be written as

$$\dot{\underline{x}}(t) = A_n \underline{x}(t) + \underline{b}_n\left(u(t) + L(\underline{x},t)\right) \tag{5.2}$$

$L(\underline{x},t)$ is called *lumped uncertainty* given by $L(\underline{x},t) = B_p\left(\Delta A \underline{x}(t) + \Delta \underline{b} u(t) + \underline{d} f(t)\right)$ and bounded as $\left|L(\underline{x},t)\right| \le L_{\max}$. Note that B_p is the pseudo inverse of $\underline{b}_n$ and given as $B_p = \left(\underline{b}_n^T \underline{b}_n\right)^{-1} \underline{b}_n^T$.

The control problem is to find a control input u such that the state vector $\underline{x}$ tracks a desired trajectory $\underline{x}^d$ in the presence of model uncertainties and external disturbance. The tracking error is defined as (the argument *t* is omitted in the following for simplicity of notation)

$$\underline{e} = \underline{x} - \underline{x}^d = [e, \dot{e}]^T \tag{5.3}$$

Note that (5.3) implies that the states are chosen as $[x, \dot{x}]^T$ which is called the control canonical form [58]. All controllable systems can be converted to this form and there is no loss of generality in assuming the form (5.3).

The switching function is

$$S = \lambda e + \dot{e} = C\underline{e} \tag{5.4}$$

If the initial condition

$$S(0) = 0 \tag{5.5}$$

is not satisfied then the tracking can only be achieved after a transient (reaching mode or phase) [60]. The tracking problem requires the design of a control law such that the trajectory $\underline{e}$, starting from any initial condition, reaches the switching line S = 0 in a finite time and then slides along it towards the origin $\underline{e}(0)$ exponentially with a time constant $1/\lambda$. In order to derive such a control law, a Lyapunov function is defined as

$$V = \frac{1}{2}S^2 \tag{5.6}$$

where V(0) = 0. A sufficient condition for this control requirement is [60]

$$V = \frac{1}{2}\frac{d}{dt}\left(S^2\right) \le -\eta|S| \tag{5.7}$$

where $\eta>0$. From (5.7), we obtain

$$\dot{S}\,\mathrm{sgn}(S) \le -\eta \tag{5.8}$$

$$\text{where } \mathrm{sgn}(S) = \begin{cases} +1 & S>0 \\ -1 & S<0 \end{cases}$$

Equation (5.8) is called *reaching condition* (or the existence condition of the sliding mode) and if satisfied, it drives the system into the *sliding mode*. Once the trajectory of $\underline{e}$ has reached the sliding line S = 0, the system trajectory remains on it while sliding into the origin $\underline{e} = \underline{0}$, independently of parameter uncertainties and external disturbances. This phenomena is called the *sliding mode*. Reaching mode defines the trajectory of $\underline{e}$ prior to reaching the sliding line. Fig.5.1 shows the reaching and sliding modes for a second order system with $\underline{e}(0) = [e_0, 0]^T$.

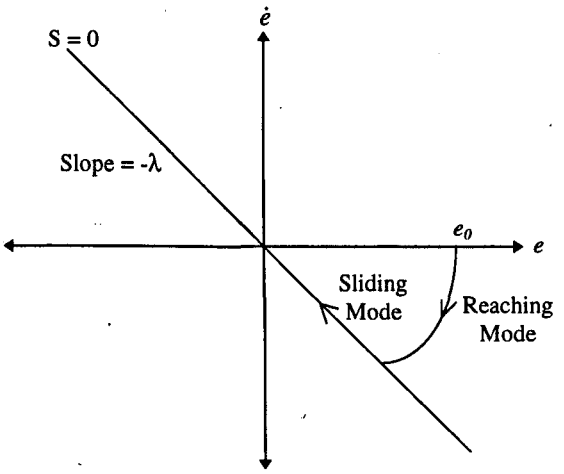


Figure 5.1 Reaching and sliding modes for a second order system

Note that reaching mode occurs if the initial condition (5.5) does not hold. However, satisfying (5.8) guarantees that the trajectory $\underline{e}$ will reach S = 0 line in a finite time less than $|S(0)|/\eta$ [60]. The dynamics in sliding mode can be expressed as $\dot{S}=0$ and by solving this equation for the control input, an expression is obtained for u called the *equivalent control*, u_{eq}, which can be interpreted as the continuous control law that would maintain $\dot{S}=0$ if the dynamics were exactly known. Let us consider the system (5.2) and assume that there is no uncertainty and disturbance (i.e. the system is with nominal parameters). Equation (5.2) becomes

$$\dot{\underline{x}} = A_n \underline{x} + \underline{b}_n u \tag{5.9}$$

Setting $\dot{S}=0$ in (5.4) and substituting (5.3) and (5.9), u_{eq} can be derived as

$$u_{eq} = \left(C\underline{b}_n\right)^{-1} C\left(\dot{\underline{x}}^d - A_n \underline{x}\right) \tag{5.10}$$

The typical structure of a robust controller is composed of a nominal part (u_{eq}) and additional terms aimed at dealing with the uncertainties and disturbances. In order to satisfy the reaching condition (5.8) under such uncertainties and disturbances, a term which is discontinuous across the line S = 0 is added to u_{eq}. The control input becomes

$$u = u_{eq} - U_{\max}\,\mathrm{sgn}(S) \tag{5.11}$$

Now the problem is to find U_{max} which should satisfy the reaching condition (5.8) in the presence of model uncertainties and external disturbance. By substituting (5.11) in (5.2) and using (5.3) and (5.4), we obtain

$$\dot{S} = C\underline{b}_n\left(-U_{\max}\,\mathrm{sgn}(S) + L\right) \tag{5.12}$$

and if (5.12) is substituted in (5.8), then we have

$$C\underline{b}_n U_{\max} \geq C\underline{b}_n L\,\mathrm{sgn}(S) + \eta \tag{5.13}$$

which requires

$$U_{\max} \geq L_{\max} + (C\underline{b}_n)^{-1}\eta \quad if \quad C\underline{b}_n > 0$$
$$U_{\max} \leq -L_{\max} + (C\underline{b}_n)^{-1}\eta \quad if \quad C\underline{b}_n < 0 \tag{5.14}$$

since $|L| \leq L_{\max}$ implies $-L_{\max} \leq L \leq L_{\max}$. If U_{max} is chosen according to (5.14) and the control law (5.11) is used, then the reaching condition is satisfied for the uncertain system given by (5.1).

5.2.1 Discrete Time Implementation of the SMC Strategy and the Chattering Problem

As shown in Section 5.2, the control input u contains a sgn(.) function (the ideal relay characteristic) to deal with the uncertainties and disturbances. In continuous time SMC systems, it is assumed that this function switches between +1 and -1 at infinite rate about the S = 0 line. Because of this infinitely fast switching of the control input, an ideal sliding mode exists on the line S = 0, meaning there is no chattering [35]. However, in practical systems, it is impossible to achieve the ideal infinite switching of the control input due to the microprocessor implementation of the control law which requires a finite computation time. Since it is impossible to switch the control input at infinite rate, chattering always occurs in the sliding and steady state modes of a practical SMC system. A typical phase plane of a second order system which exhibits chattering problem as a result of imperfect control switching is shown in Fig.5.2.

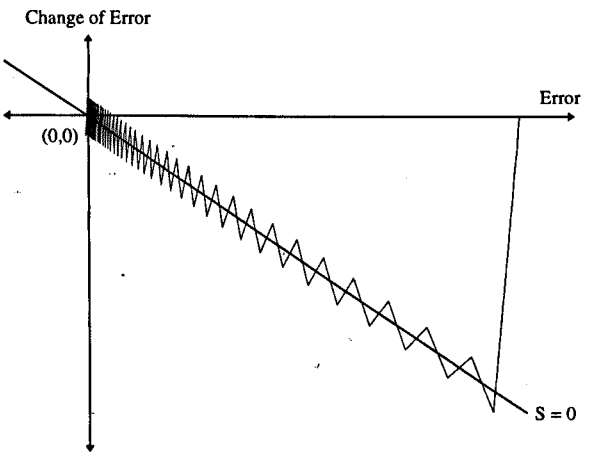


Figure 5.2 Phase plane of a second order system as a result of imperfect control switching

Chattering appears as a high frequency oscillation about the desired equilibrium point in the steady state and can excite the unmodelled high frequency dynamics of the system. Since chattering is almost always undesirable for most practical applications, many researchers have directed their work to this problem as reported in [35].

There are several common methods which are used to eliminate or reduce the chattering : The most popular is to replace the discontinuous term sgn(S) by

$$sat(S) = \begin{cases} S/\phi & |S| \le \phi \\ \text{sgn}(S) & |S| > \phi \end{cases} \tag{5.15}$$

where ϕ is an positive constant and usually called *boundary layer thickness* since using (5.15) means that a boundary layer around the switching line (or surface) is introduced to eliminate the chattering. Another common continuation method is to replace sgn(S) by

$$\frac{S}{|S| + \delta} \tag{5.16}$$

where δ is a positive small constant.

A different approach for designing the sliding mode controllers is introduced by Gao and Hung in [36]. This approach is called the *reaching law approach* and will be considered in details in Section 5.5. The reaching law is defined by

$$\dot{S} = -q\,\text{sgn}(S) - \alpha S \tag{5.17}$$

The extra term α allows the dynamics of S to be fast for a small q which is desirable for reducing chattering. The reaching law approach has other advantages and these will be discussed in Section 5.5.

In the following section, the most popular chattering elimination method represented by (5.15) is discussed in more details. Sliding Mode Fuzzy Controllers (SMFCs) based on the idea of SMC with Boundary Layer (BL) are also briefly discussed in the following section.

5.3 Sliding Mode Control with Boundary Layer

In the previous section, it has been shown that the switching of the control input at a finite rate (imperfect switching) results in a chattering problem. Introducing a Boundary Layer (BL) around the switching line is one of the most common method to eliminate the chattering. Fig5.3a shows a typical phase plane of a second order SMC system with a BL under no parameter variations and disturbances.

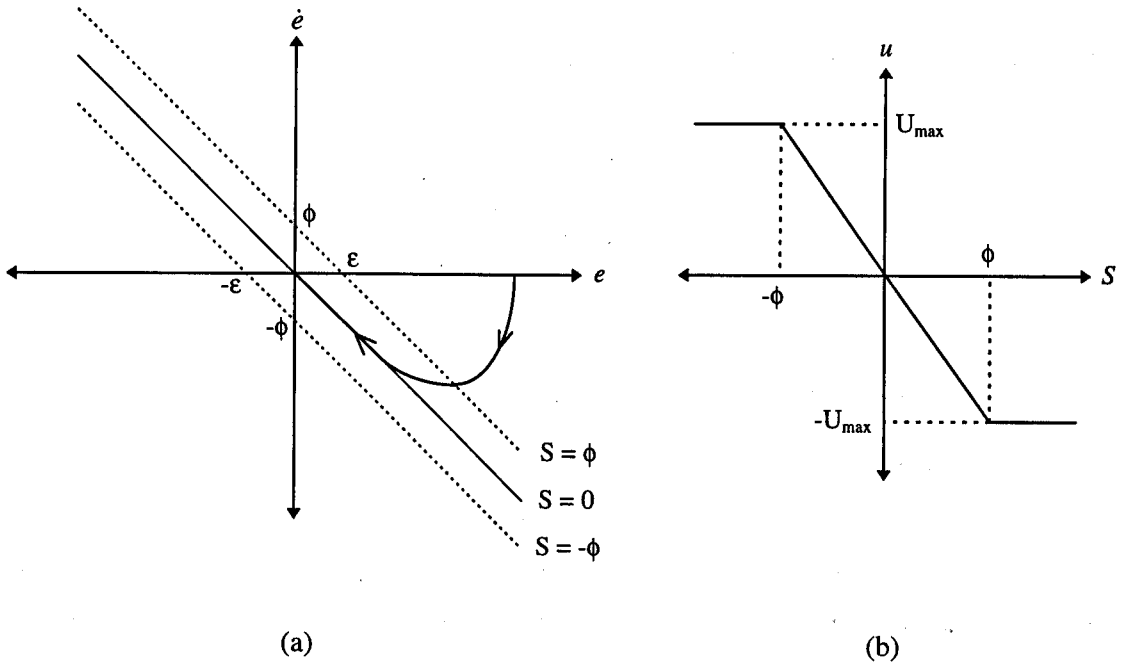


Figure 5.3 :

(a) A typical phase plane of a second order SMC system with BL

(b) Transfer characteristic of the control input with BL (excluding u_{eq})

The width of the BL is 2ϕ and ε is the *guaranteed tracking precision* [60] and given by

$$\varepsilon = \phi / \lambda \tag{5.18}$$

The BL smoothes out the dynamics of the control input u and ensures that the system trajectory remains within the layer. Fig.5.3b shows the transfer characteristic of the control input (excluding u_{eq}) with the BL. The control input becomes

$$u = u_{eq} - U_{\max} sat(S) \tag{5.19}$$

where sat(S) is given by (5.15).

The S = 0 line is the domain of attraction in the *ideal* SMC. However, in the SMC with BL, the boundary layer becomes the domain of attraction. This implies that the chattering is eliminated with the cost of a decrease in the tracking performance [59,60]: unless integral action is introduced elsewhere in the control, the effect of the BL is to increase the tracking error.

Let us consider the system (5.2). In the BL, the behaviour of the system can be monitored by the dynamics of S since the variation of S with time is a compact descriptor of the closed loop behaviour [60]. From (5.12) and using (5.19) instead of (5.11) gives the dynamics of S in the BL as

$$\dot{S} + \frac{C\underline{b}_n U_{\max}}{\phi} S = C\underline{b}_n L \tag{5.20}$$

It can be seen from (5.20) that the switching function S can be viewed as the output of a first order filter whose input is the lumped uncertainty L($\underline{x}$,t) (perturbations). The dynamics of the equation (5.20) describes the trajectory $\underline{e}$ approaching the sliding line in the BL [38,60]. The structure of the closed loop error dynamics can be summarised by Fig.5.4: lumped uncertainty is filtered according to (5.20) to give S, and then e is obtained after another low pass filter which is actually the definition of S (p is the differential operator). If L = 0 then the system slides smoothly down S = 0 under the action of u_{eq}; in this case the trajectory $\underline{e}$ arises from the initial condition of the $1/(p+\lambda)$ filter. Also as $\phi \rightarrow 0$, the gain of the first filter becomes zero to all finite frequencies of L; this corresponds to the perfect switching case which is completely robust.

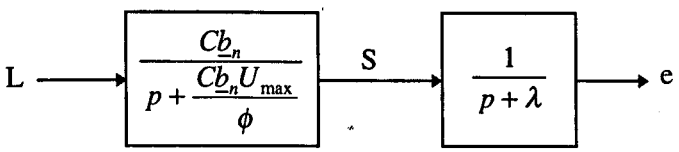


Figure 5.4 The structure of the closed loop error dynamics

5.3.1 Sliding Mode Fuzzy Controllers

For a large class of second order systems, Fuzzy Controllers (FCs) are designed by using the phase plane composed of error and change of error. The rule base of such a FC is shown in Fig.5.5. As seen in Fig.5.5, there is a virtual switching line on the rule base of the FC. The magnitude of the FC output depends on the distance of the error vector $\underline{e} = [e, \dot{e}]^T$ from the switching line S = 0. The rules are conditioned in such a way that above the switching line a negative control output is generated and below it, a positive one is generated similar to the SMC with BL. The output of the FC can be expressed in the form of [37-39]

$$u_{FZ} = -U_F(e, \dot{e}, \lambda)\,\mathrm{sgn}(S) \tag{5.21}$$

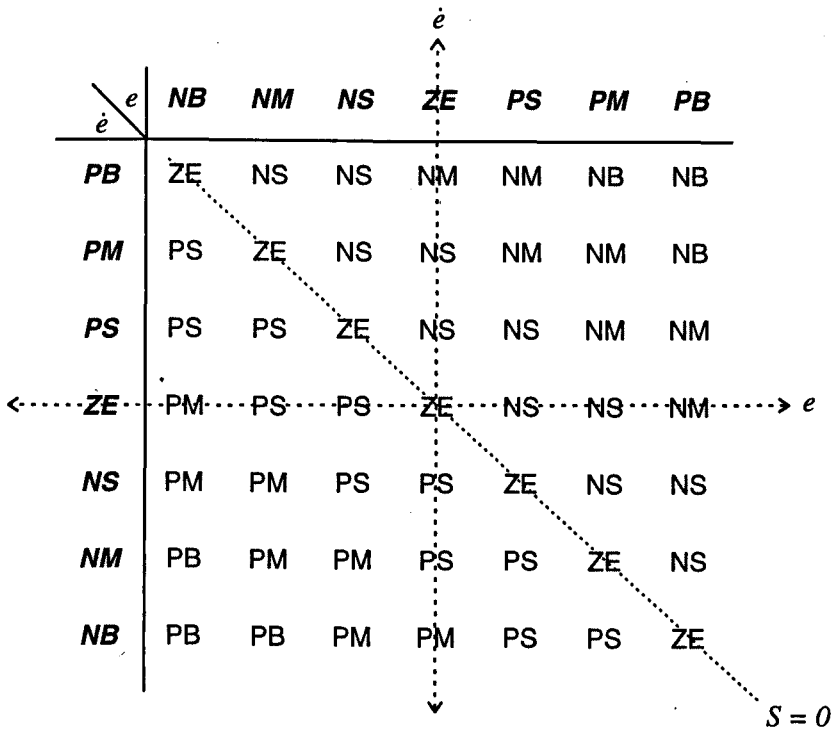


Figure 5.5 Rule base of the FC

Since the FC changes the magnitude of u_{FZ} depending on the distance $|S|$ between the error vector $\underline{e}$ and diagonal S = 0, (5.21) can also be written as [38]

$$u_{FZ} = -U_F(|S|)\,\mathrm{sgn}(S) \tag{5.22}$$

On the other hand, the output of the sliding mode controller with BL expressed by (5.19) can be rewritten as (note that $S = |S|\,\mathrm{sgn}(S)$)

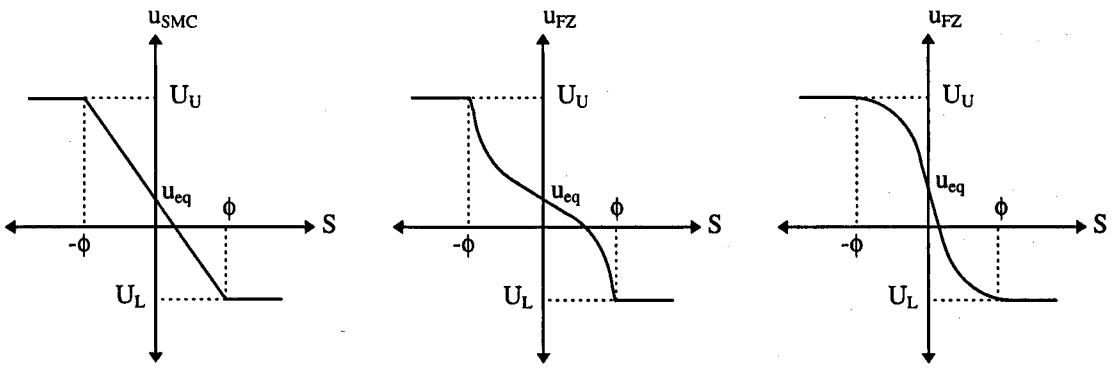
$$u_{SMC} = u_{eq} - \frac{U_{\max}}{\phi}|S|\,\mathrm{sgn}(S) \tag{5.23}$$

It is easy to see the similarity between (5.22) and the second term of (5.23). Because of this similarity, the fuzzy controllers designed in the above manner are called *Sliding Mode Fuzzy Controllers* (SMFCs) [37-39].

The term u_{eq} can easily be included in the SMFC and (5.22) can be rewritten as

$$u_{FZ} = u_{eq} - U_F(|S|)\,\mathrm{sgn}(S) \tag{5.24}$$

which implies that SMC with BL is a special case of SMFC. A SMC with BL provides a linear transfer characteristic with lower and upper bounds as seen in Fig.5.6a.



(a) (b) (c)

Figure 5.6 Controller transfer characteristics of

(a) the SMC with BL

(b) the SMFC : Low steady state gain for improving noise performance whilst maintaining high transient dynamics

(c) the SMFC : Large steady state gain for fast disturbance rejection

However, the transfer characteristic of an SMFC is not necessarily a straight line between these bounds, but a curve that can be adjusted by the tools of fuzzy logic (e.g. membership function) to reflect given performance requirements [38]. For example, the transfer characteristic of Fig.5.6b is preferred in order to have a little over-shoot and fast convergence whilst giving lower gain in steady state to be within system noise limits [38]. The controller characterised by Fig.5.6c can be chosen if the system is to be sensitive with respect to disturbances for small errors but fast transient responses are not so important. In fact, with S defined by (5.4), Fig.5.6b and c correspond to a low and high gain PD controller about e = 0.

5.4 Sliding Mode Speed Control Systems

In the control of electrical drive systems, SMC technique has been mostly applied in position control systems having second order dynamics [11,13,14,16,19,61,62]. This arises from a basic inertial and frictional load in which the states are defined as $\underline{x} = [\theta, \dot{\theta}]^T$ where θ is the shaft position. Speed control of the basic inertial and frictional load gives only a first order dynamics (electrical dynamics are neglected) and direct application of SMC is not applicable [9,10]. However, the system can be made second order by adding integral compensation which has the natural advantage of eliminating the speed error in dc steady state. There are two approaches to integral compensation: the first is called Sliding Mode Control with Integral Compensation (SMC-IC) [9] shown in Fig.5.7a and the second method, called Integral Sliding Mode Control (ISMC) or Integral Variable Structure Control (IVSC) [10,12], is shown in Fig5.7b (note that the control structures shown in Fig 5.7a and b are not exactly same with the ones used in [9] and [10,12] respectively, but the integral compensation techniques are same). ω and ω_{ref} are the actual and reference speeds respectively. The whole closed loop speed control systems are not shown for clarity.

The main difference between two structures is in the choice of switching function. In Fig.5.7a, classical switching function $S = \lambda e + \dot{e}$ is implemented and then the control input u is integrated to obtain the torque input to the first order plant. On the other hand, in Fig.5.7b, a switching function

$$S = e + \lambda \int_{-\infty}^{t} e d\tau \tag{5.25}$$

is employed [10,12]. In this approach, the acceleration (actually the derivative of the error), which has generally high bandwidth and is very noisy in practice, is not required. Although this seems to be an advantage over the previous structure shown in Fig.5.7a, both systems become equivalent in the BL. This is because, in the BL, the systems are linear and if the integral compensation term of Fig.5.7a is moved towards the left, it is easy to see that both systems become identical.

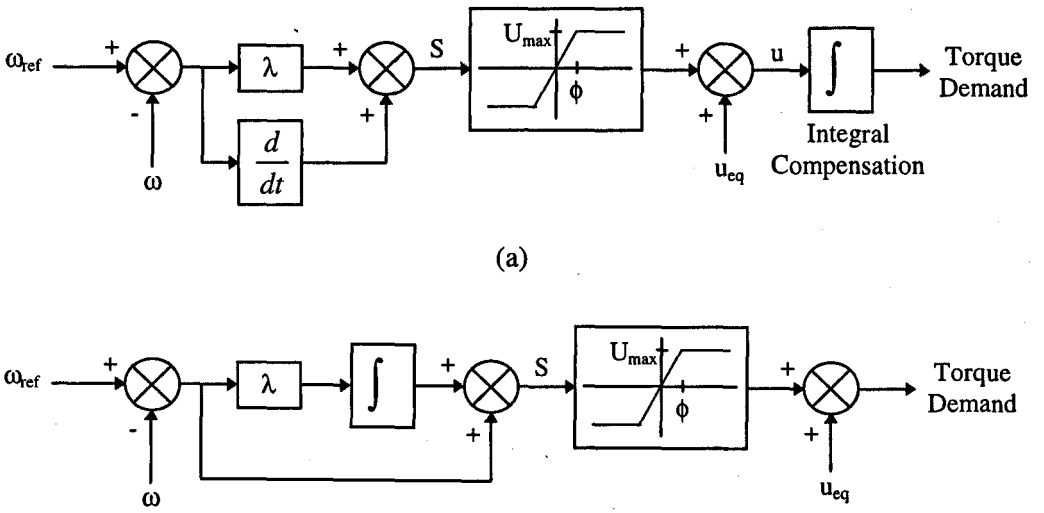


Figure 5.7 Implementation of SMC strategy for speed control systems using
(a) SMC with Integral Compensation (SMC-IC) technique
(b) Integral-SMC (ISMC) technique

There may be some advantages and disadvantages of both structures to each other when the practical implementation is considered, but this is not our main concern. In the Model Reference Reaching Law Control (MRRLC) approach introduced in Section 5.5.2, the structure shown in Fig.5.7a is used. This is simply because it is found more suitable for the MRRLC strategy and it is easier to implement the anti-windup mechanism which is required in practice to limit the torque demand and to stop the integration during the saturation. However, the structure in Fig.5.7b may also be used in the MRRLC strategy with some small modifications.

In Fig.5.7a, due to the limited output of the integrator with an anti-windup mechanism, there is no point to have the U_{max} limit in the control law. Thus, the gain U_{max}/ϕ becomes the quantity of interest. This will be discussed in more details in the following section.

5.5 Reaching Law Control (RLC)

Gao and Hung have introduced a new method called Reaching Law Control (RLC) for the design of SMC systems [36]. In their approach, a *reaching law* which is a differential equation specifying the dynamics of the switching function S is first chosen. The control input is then synthesised from the reaching law in conjunction with a known model of the plant and the known bounds of perturbations. It should be noted that the differential equation of an asymptotically stable S is actually a reaching condition. In addition, the dynamic quality of the SMC system in the reaching mode can be controlled by choosing the parameters in the differential equation.

Let us consider the system (5.2). If an asymptotically stable reaching law [36] is chosen as

$$\dot{S} = -q\,\mathrm{sgn}(S) - \alpha S \tag{5.26}$$

where q and α are positive constants, the control input u is derived by using (5.2)-(5.4) and (5.26) as

$$u = \left(C\underline{b}_n\right)^{-1}\left(-CA_n\underline{x} + C\dot{\underline{x}}^d - C\underline{b}_n L - q\,\mathrm{sgn}(S) - \alpha S\right) \tag{5.27}$$

All the quantities above the right hand side of (5.27) are known except the lumped uncertainty L. If L in (5.27) is replaced by a conservative known quantity L_c, then u becomes

$$u = \left(C\underline{b}_n\right)^{-1}\left(-CA_n\underline{x} + C\dot{\underline{x}}^d - C\underline{b}_n L_c - q\,\mathrm{sgn}(S) - \alpha S\right) \tag{5.28}$$

The dynamics of S is obtained by using (5.2)-(5.4) and (5.28) as

$$\dot{S} = -q\,\mathrm{sgn}(S) - \alpha S + C\underline{b}_n\left(L - L_c\right) \tag{5.29}$$

By comparing (5.29) with (5.26), it can be easily seen that an additional term $C\underline{b}_n(L-L_c)$ appears in the reaching dynamics of the perturbed system. L_c will be so chosen that it dominates the unknown lumped uncertainty L and thus ensures the reaching law (5.26). Since L is bounded as $|L(\underline{x},t)| \le L_{max}$ and assuming $C\underline{b}_n$ is a positive constant, a practical choice of L_c is

$$L_c = L_{max} \quad when \quad S > 0$$
$$L_c = -L_{max} \quad when \quad S < 0 \tag{5.30}$$

If $C\underline{b}_n$ is negative then the sign of L_{max} will be opposite in (5.30) which implies that

$$L_c = L_{max}\,\mathrm{sgn}(S) \tag{5.31}$$

and the dynamics of S becomes

$$\dot{S} = -Q\,\mathrm{sgn}(S) - \alpha S + C\underline{b}_n L \tag{5.32}$$

where $Q = q + C\underline{b}_n L_{max}$. It should be remembered that the term αS is added in the reaching law (5.26) to increase the reaching rate [36]. If α is set to zero, then (5.32) becomes equivalent to (5.12), which is the dynamics of S obtained by the *classical* SMC design approach. The control law (5.28) thus becomes equivalent to the control law (5.11). In (5.32), Q corresponds to $C\underline{b}_n U_{max}$ in (5.12). In this manner, the reaching law design approach is a more conservative form of the classical approach since an additional term L_c is used to compensate the perturbations to ensure the reaching law (5.26). In other words, even though L_c and α are set to zero in (5.28), the system can still be forced so that the sign of $\dot{S}$ is opposite to the sign of S (reaching condition) by choosing q sufficiently high. This is actually the classical SMC design approach.

The main difference between the classical and the reaching law SMC design approaches is the starting point of the design procedure: In the RLC approach, the design starts by defining the desired dynamics of S (the reaching law) and then the control law is easily derived using the reaching law and system equations. However, in the classical approach, a Lyapunov function is defined first and then the control law is *chosen* to satisfy the reaching condition derived from the Lyapunov function. Thus, the RLC method simplifies the derivation of the control law [36].

In the following sections, a speed control structure shown in Fig.5.7a will be considered. As mentioned in Section 5.4, in practice, the output of the integrator is limited by an anti-windup mechanism. Due to the limited integrator, there is no point in employing another limit (the U_{max} limit introduced by the sat(.) function) in the control law of SMC with BL. Thus, the gain U_{max}/ϕ, which is multiplied by S, becomes the quantity of interest. The RLC method implies that a control law without the sat(.) function can be directly obtained by simply setting the parameter q equal to zero (see (5.28)). Thus, the RLC method becomes more convenient for the practical application using the control structure shown in Fig.5.7a with a limited integrator.

5.5.1 Discrete Time Speed Control using the RLC Approach

In this section, the discrete time speed control system is considered since it is more convenient for experimental implementation. As seen in Section 5.2.1, direct implementation of sgn(.) function results in a chattering problem in the discrete time systems. In a speed control system, chattering of the torque demand is usually unacceptable since it may excite the unmodelled mechanical dynamics [9,35]. As discussed in Section 5.5, the reaching law approach is found more appropriate for the practical implementation. Hence, let us consider *a discrete time reaching law* without the sgn(.) function :

$$\Delta S(k+1) = -\alpha S(k) \tag{5.33}$$

where k is the sampling instant (i.e. k = 0,1,2,...), α is a positive constant and Δ operator is defined as

$$\Delta g(k+1) = \frac{g(k+1) - g(k)}{T_s} \tag{5.34}$$

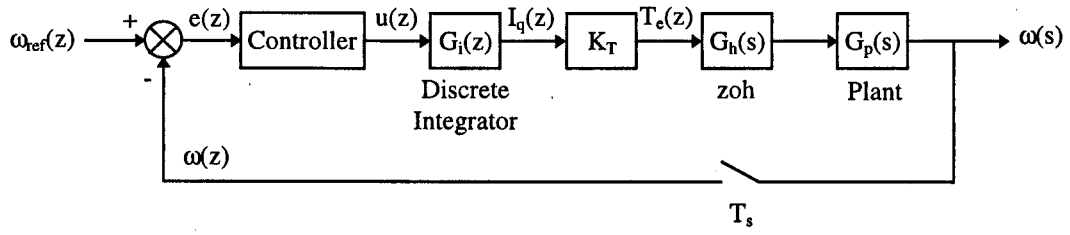
(T_s is the sampling time) which is supplemented with the condition $\{\Delta g(0) = 0\}$ and the switching function is given by

$$S(k) = \lambda e(k) + \Delta e(k) \tag{5.35}$$

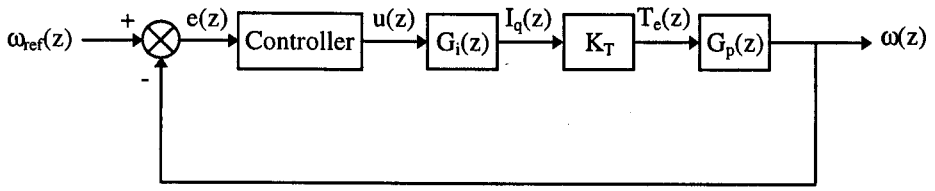
where

$$e(k) = \omega_{ref}(k) - \omega(k) \tag{5.36}$$

ω_{ref} and ω are the reference and actual speeds respectively. The reaching law (5.33) basically implies that the switching function S exponentially reduces to zero with a desired dynamics defined by α.



(a)



(b)

Figure 5.8

(a) Closed loop speed control system

(b) Pure discrete time representation of the speed control system

Now let us consider the closed loop speed control system shown in Fig.5.8a, where

$$G_i(z) = \frac{T_s z}{z-1} \tag{5.37}$$

$$G_p(s) = \frac{1}{Js + B} \tag{5.38}$$

and I_q is the q-axis current, K_T is the torque constant, T_e is the electrical torque demand (current control loop delay is ignored) and $G_h(s)$ represents the zero order hold (zoh). Fig.5.8b shows the pure discrete time representation of the speed control system, where

$$G_p(z) = Z\{G_h(s)G_p(s)\} = \frac{C_p}{z - P_p} \tag{5.39}$$

and

$$P_p = \exp(-BT_s / J) \quad \text{and} \quad C_p = (1 - P_p) / B.$$

From Fig.5.8b and using (5.34)-(5.36), we obtain

$$\Delta S(k+1) = \frac{(1+\lambda T_s)}{T_s}\left((P_p - \frac{1}{(1+\lambda T_s)})\Delta e(k) - K_T C_p u(k) + \Delta\omega_{ref}(k+1) - P_p\Delta\omega_{ref}(k)\right) \tag{5.40}$$

Setting (5.40) equal to -αS(k) from (5.33), and solving for u(k) gives the control law

$$u(k) = \left(\frac{T_s\alpha}{(1+\lambda T_s)K_T C_p}\right)S(k) + \left(\frac{(1+\lambda T_s)P_p - 1}{(1+\lambda T_s)K_T C_p}\right)\Delta e(k) + \frac{1}{K_T C_p}\left(\Delta\omega_{ref}(k+1) - P_p\Delta\omega_{ref}(k)\right) \tag{5.41}$$

An advance term $\Delta\omega_{ref}(k+1)$ is seen on the right hand side of (5.41), but this is not a problem since $\omega_{ref}(k)$ is a known reference input. For simplicity, let us assume that ω_{ref} is a step demand (i.e. $\Delta\omega_{ref}(k+1) = \Delta\omega_{ref}(k) = 0$). The control law then becomes

$$u(k) = KS(k) + K_{eq}\Delta e(k) \tag{5.42}$$

where

$$K = \frac{T_s\alpha}{(1+\lambda T_s)K_T C_p} \tag{5.43}$$

$$K_{eq} = \frac{(1+\lambda T_s)P_p - 1}{(1+\lambda T_s)K_T C_p} \tag{5.44}$$

The second term of (5.42), $K_{eq}\Delta e(k)$, actually corresponds to u_{eq} which is mentioned in Section 5.2. The speed control structure with the control law (5.42) is shown in Fig.5.9.

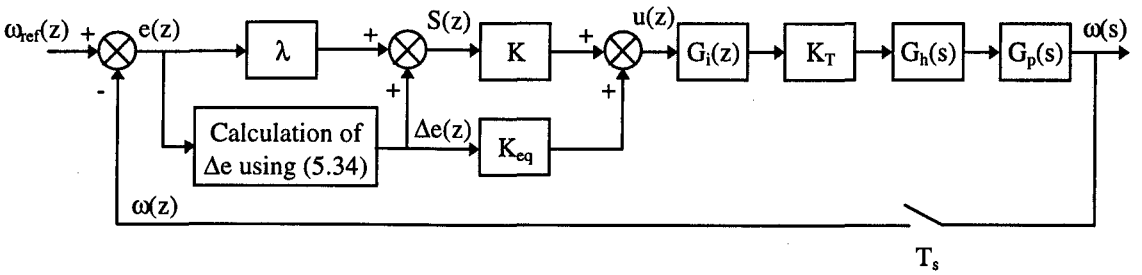


Figure 5.9 The control structure with the control law (5.42)

By using (5.33) and (5.34), an expression for S can be obtained as

$$S(k+1) = (1-\alpha T_s)S(k) \tag{5.45}$$

In order to have a stable system, α should satisfy

$$0 < \alpha < \frac{2}{T_s} \tag{5.46}$$

since a sufficient condition for the stability [63] is

$$|S(k+1)| < |S(k)| \tag{5.47}$$

which requires

$$|1-\alpha T_s| < 1 \tag{5.48}$$

It should be noted that if

$$\frac{-1}{T_s} < \alpha < \frac{2}{T_s} \tag{5.49}$$

then $1-\alpha T_s$ becomes a negative number and S will have damped chattering.

S will exponentially reduce to zero if

$$0 < \alpha \leq \frac{1}{T_s} \tag{5.50}$$

which requires

$$0 < K \leq K_m \tag{5.51}$$

where

$$K_m = \frac{1}{(1+\lambda T_s)K_T C_p} \tag{5.52}$$

The system shown in Fig.5.9 is simulated and Fig.5.10, 5.11 and 5.12 show the phase planes (e(k) & Δe(k)), variations of S, and the speed and torque responses to a step input demand (100 rad/s) for K = K_m, 0.25K_m and 1.75K_m respectively. The controller is designed for the nominal system parameters which are given in Table 5.1. λ is chosen as $25s^{-1}$ and the sampling time T_s is 2.5ms.

Table 5.1 Nominal system parameters

Inertia	Viscous Friction	Torque Constant
$J = 0.0035kgm^2$	B = 0.0007Nms	$K_T = 4.1788Nm/A$

In Fig.5.10, a perfect sliding occurs since K = K_m and α = 1/ T_s which means S becomes zero after one sampling period (i.e. S(k+1) = 0*S(k)). Fig.5.11 shows that S exponentially reduces to zero since K = 0.25K_m and α = 0.25/T_s (S(k+1) = 0.75*S(k)). On the other hand, a damped chattering is seen in Fig 5.12 because S(k+1) = -0.75*S(k) due to K = 1.75K_m and α = 1.75/T_s. Note that for K > 2K_m, the system becomes unstable.

Finally, Fig.5.13 shows the experimental and simulation results for K = 0.05 (≅ 0.16K_m) where the other parameters (e.g. λ, T_s, J, etc.) are exactly same with the ones used in the simulations above. The value of K = 0.16K_m arises from noise considerations since the encoder resolution noise restricts the controller gain. This practical limitation is discussed in Section 5.5.3. As

seen in Fig.5.13c, the torque demand is limited (with an anti-windup mechanism) in order to protect the inverter and other practical circuits in the experimental implementation (the torque demand limitation is also implemented in the simulation). In the experimental results shown in Fig.5.13, because of the encoder resolution, the signals and the phase plane are not smooth as expected.

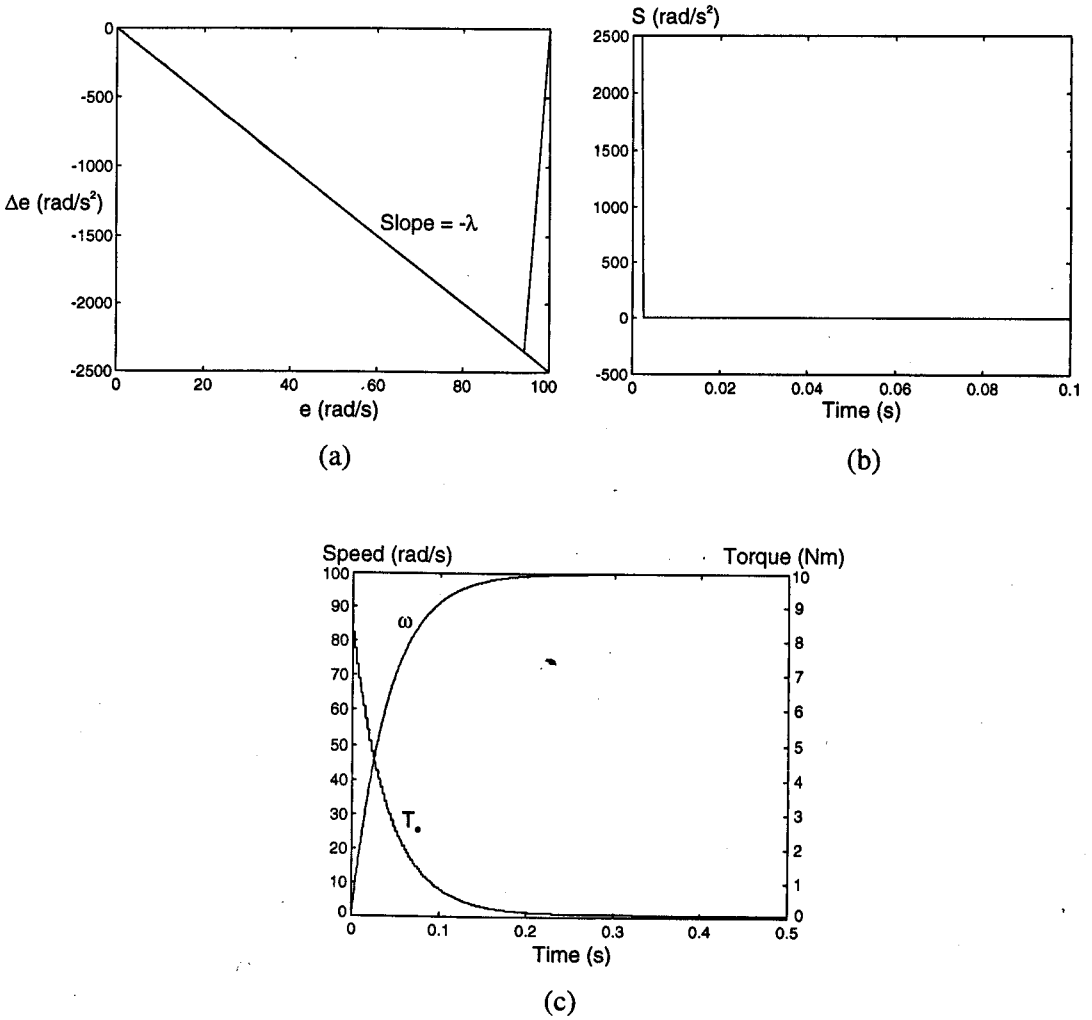
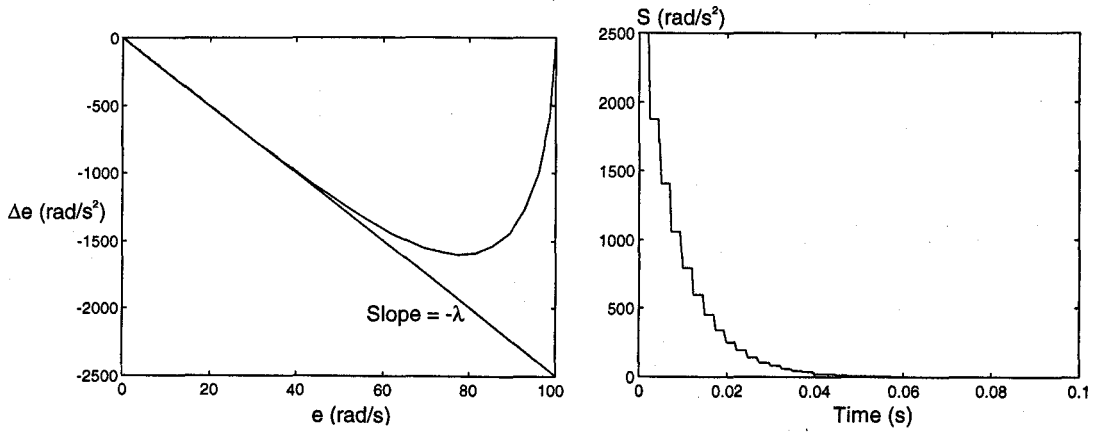


Figure 5.10 Simulation results for $K = K_m$

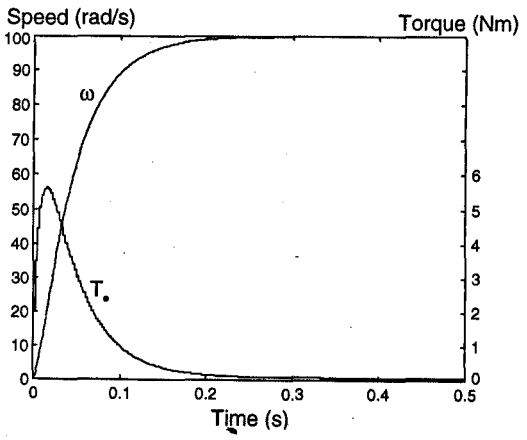
(a) Phase plane ($\Delta e(k)$ versus $e(k)$)

(b) Variation of S

(c) Speed and torque responses



(a) (b)



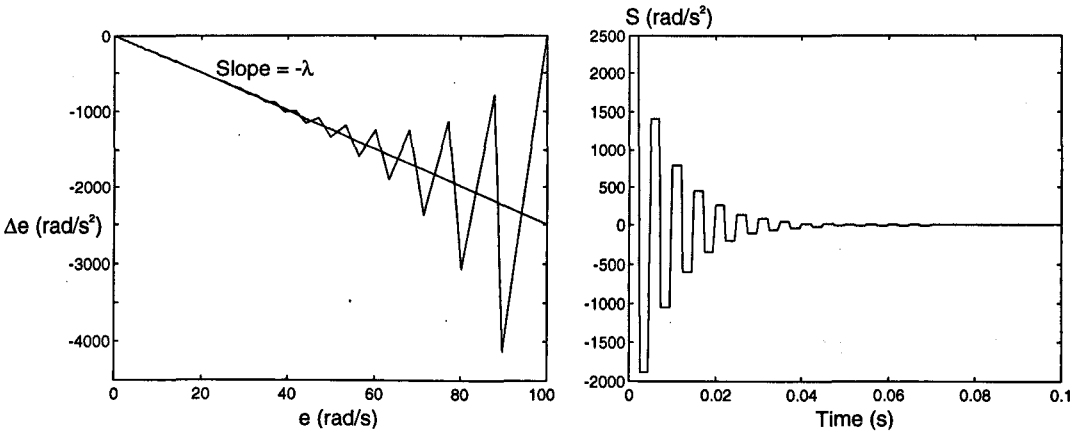
(c)

Figure 5.11 Simulation results for $K = 0.25K_m$

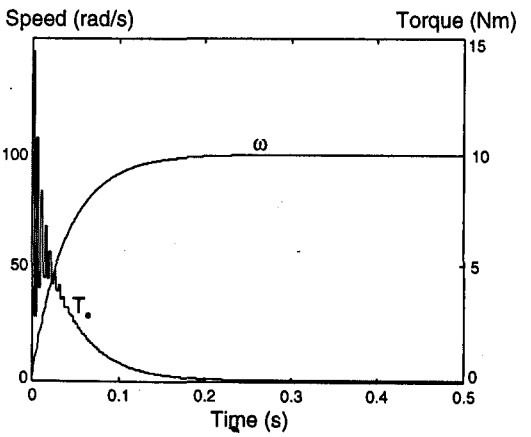
(a) Phase plane ($\Delta e(k)$ versus $e(k)$)

(b) Variation of S

(c) Speed and torque responses



(a) (b)



(c)

Figure 5.12 Simulation results for $K = 1.75K_m$

(a) Phase plane ($\Delta e(k)$ versus $e(k)$)

(b) Variation of S

(c) Speed and torque responses

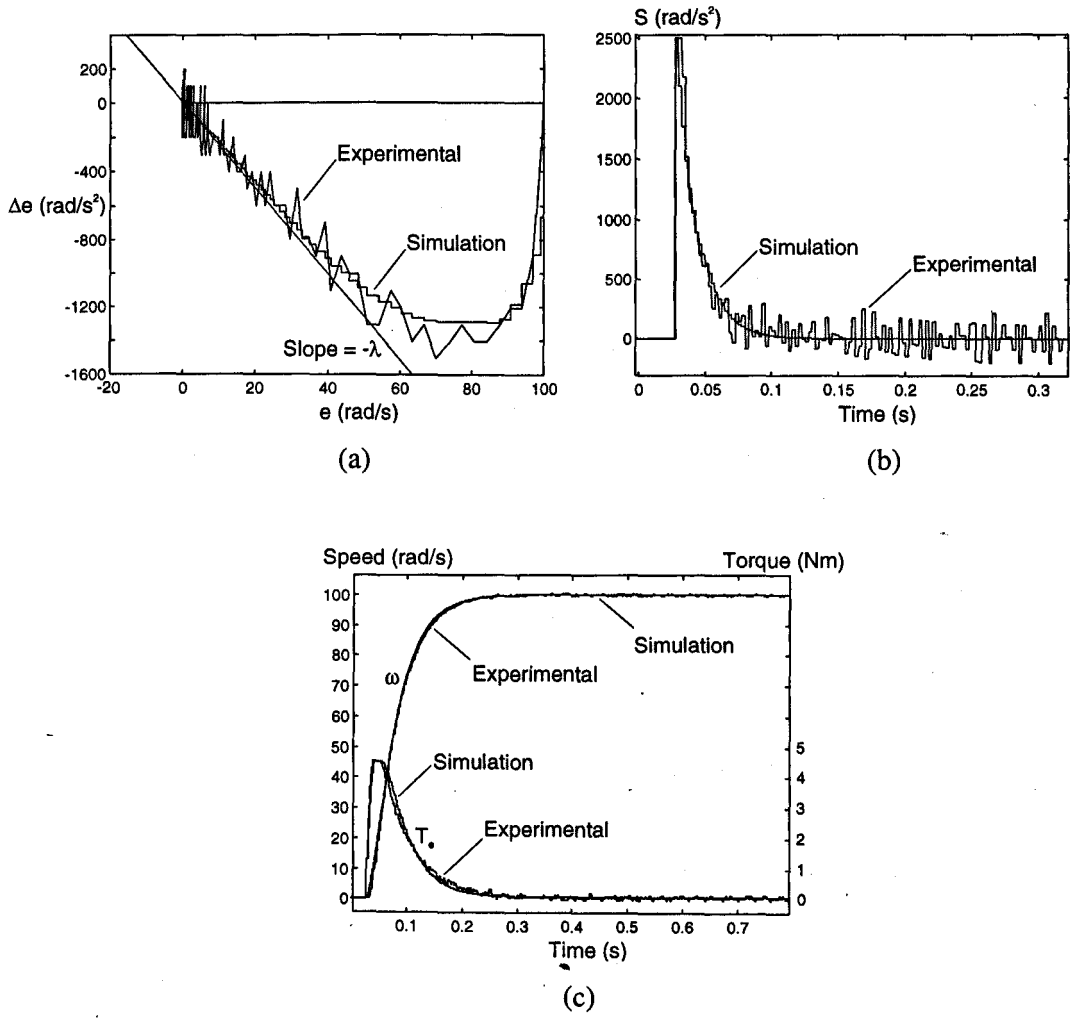


Figure 5.13 Experimental and simulation results for K = 0.05

(a) Phase planes (Δe(k) versus e(k))

(b) Variations of S

(c) Speed responses and electrical torque demands

In this section, the RLC approach has been applied to the discrete time speed control system. The plant was in the nominal conditions (i.e. no parameter variations and no external load torque). This design approach forms the basis of the MRRLC strategy which is explained in Section 5.5.2.

It should be noted that the control law similar to (5.41) can be obtained for the SMC with BL design approach as

$$u(k) = U_{max} sat(S(k)) + \left(\frac{(1+\lambda T_s)P_p - 1}{(1+\lambda T_s)K_T C_p} \right) \Delta e(k) + \frac{1}{K_T C_p} \left(\Delta \omega_{ref}(k+1) - P_p \Delta \omega_{ref}(k) \right) \quad (5.53)$$

In the BL, $sat(S(k)) = \frac{S(k)}{\phi}$, thus $\frac{U_{max}}{\phi}$ directly corresponds to the gain K of the RLC design approach if (5.41) and (5.53) are compared. The main difference between these two control law is the limitation due to the sat(.) function as seen in (5.53). Because of the limited integrator, as discussed in Section 5.5, the U_{max} limit becomes redundant and thus the gain U_{max}/ϕ becomes the quantity of interest. Therefore, the RLC approach has been found more appropriate for the practical implementation of the control structure shown in Fig.5.7a.

5.5.2 Model Reference Reaching Law Control (MRRLC)

In Section 5.5.1, the speed controller is designed for the nominal conditions. The system will not obviously respond as it does in the nominal conditions in the presence of parameter variations and external disturbances. In this section, a new method is developed to get information about the parameter variations and external disturbances. This information will then be used to take an appropriate control action in the case of inertial-frictional variations and external load torque disturbances.

Before starting to the new method, let us see how the perturbations affect the dynamics of the system. Consider the system shown in Fig.5.14, where $T_L(s)$ is an external load torque and the plant may have different inertia and viscous friction than the nominal ones.

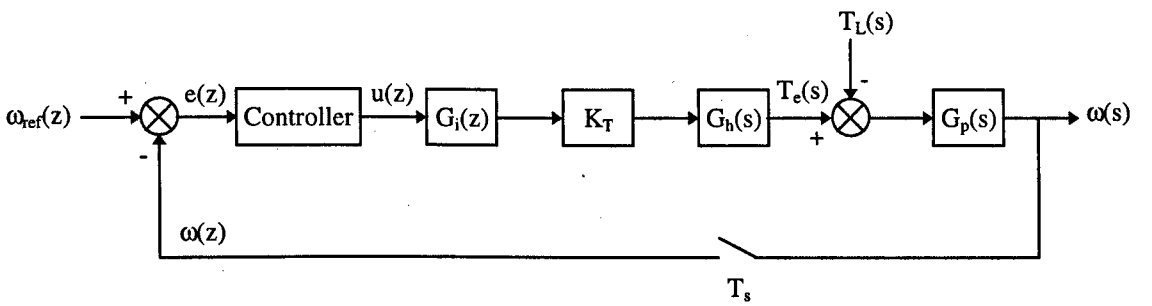


Figure 5.14 Control system in the presence of perturbations

Since the control law is designed for the nominal conditions, u can be written as (assuming that ω_{ref} is a step demand)

$$u(k) = \left(\frac{T_s\alpha}{(1+\lambda T_s)K_T C_n}\right)S(k) + \left(\frac{(1+\lambda T_s)P_n - 1}{(1+\lambda T_s)K_T C_n}\right)\Delta e(k) \tag{5.54}$$

where

$$P_n = \exp(-B_n T_s / J_n)$$
$$C_n = (1-P_n)/B_n \tag{5.55}$$

and J_n and B_n are the nominal inertia and viscous friction respectively. From Fig.5.14, the dynamic equation of S can be derived as

$$\Delta S(k+1) + \left(\frac{C_p}{C_n}\alpha\right)S(k) = \left(\frac{(1+\lambda T_s)(P_p C_n - P_n C_p) - (C_n - C_p)}{T_s C_n}\right)\Delta e(k) + \left(\frac{(1+\lambda T_s)C_p}{T_s}\right)\Delta T_L(k) \tag{5.56}$$

which obviously means that the S trajectory of the system will be different than the designed one. Hence, the speed response will not be as good as that under nominal conditions. However, the robustness of the control performance can be improved by forcing the system to follow *a reference S trajectory* which is actually the desired trajectory designed for the nominal condition. Thus the dynamic equation of the reference S trajectory is

$$\Delta S_{ref}(k+1) + \alpha S_{ref}(k) = 0 \tag{5.57}$$

which can be solved for S_{ref} as

$$S_{ref}(k+1) = (1-\alpha T_s)S_{ref}(k) \tag{5.58}$$

where $S_{ref}(0) = S(0)$ and $S(0) = \lambda e(0)$ since $\Delta e(0) = 0$.

Fig.5.15 shows the block diagram of the proposed control approach which is called Model Reference Reaching Law Control (MRRLC) because the control strategy is based on the Reaching Law Control (RLC) approach with an additional Reference Model. As seen in Fig.5.15, the error between S and S_{ref} (e_s) is multiplied with a gain K_e in order to produce an

additional control input which forces S to follow S_{ref}. For example, assume that the inertia of the plant is higher than the nominal one. In this case, S will reduce to zero slower in comparison with S_{ref} because increasing the inertia means reducing the plant's gain and therefore reducing the magnitude of the change of error. The error between S and S_{ref} is multiplied by K_e and added to the torque demand. This will increase the torque demand (See Fig.5.15) which is required since the system needs more torque to compensate the error between S and S_{ref}.

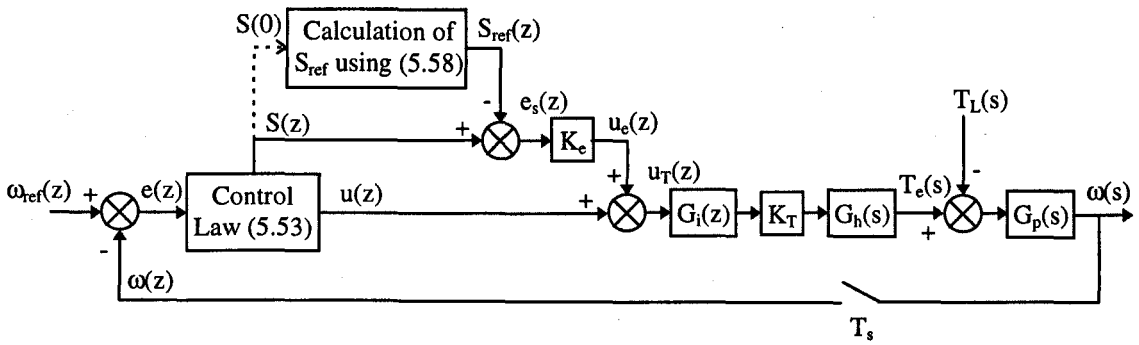


Figure 5.15 MRRLC approach

It is easy to see that when S_{ref} becomes zero, the gain K_e emerges as an additional gain to the gain K and thus $u_T(k)$ can be written as

$$u_T(k) = (K + K_e)S(k) + K_{eq}\Delta e(k) \tag{5.59}$$

As far as the stability is concerned, the limits of the gain K_e can be determined by considering the worst case which occurs when the plant has its maximum gain (Corresponding to the minimum inertia and friction which are J_n and B_n respectively). Thus, the gain K_e should be chosen as

$$0 \le K_e < \frac{2}{(1+\lambda T_s)K_T C_n} - K \tag{5.60}$$

which is obtained by substituting α in (5.46), where α is derived using (5.43) in which $K+K_e$ is substituted instead of K. Although (5.60) is true for the stability consideration, the condition $K+K_e \le K_m$ should be satisfied if a damped chattering is not acceptable (see Equations (5.48)-(5.52)). However, in practice, the selections of K and K_e are restricted by the encoder resolution

noise which will be discussed in Section 5.5.3. The practical selections of K and K_e will be considered in the final controller design procedure based on the FLC and MRRLC approaches in Section 5.6. In this section, the values of K and K_e are selected only to illustrate the principle of the MRRLC approach.

The MRRLC method shown in Fig.5.15 is simulated and simulation results are shown in Fig.5.16 in comparison with the RLC method for the nominal conditions ($J = J_n$, $B = B_n$ and no load torque). It should be remembered that when any two control methods are compared, then to have a fair comparison, both control methods should result in exactly same or very similar responses to the same inputs for the *nominal conditions*. When the plant parameters are changed or an external load torque is applied, then we can see which control method is more robust. Fig 5.16 shows that both control method give identical results since there is no difference between the S and S_{ref} trajectories. However, in Fig.5.17, the inertia and the viscous friction of the plant are increased to five times of the nominal values and it is clear that the MRRLC method shows more robust control performance than the RLC method. The parameters used in the simulations are given in Table 5.2, where $K_m = 0.3154$ determined by using (5.52) with the nominal parameters. The values of K and K_e in Table 5.2 are for the illustration of the MRRLC principle (their selection is considered in Section 5.6). The input is 100 rad/s step demand for both Figures.

Table 5.2 Numerical values used in the simulations

J_n	B_n	λ	α	K	K_e
$0.0035 kgm^2$	0.0007Nms	$25\ s^{-1}$	$80\ s^{-1}$	$0.2*K_m$	$0.8*K_m$

Fig.5.18a and b show the speed and torque responses to a step input demand (100 rad/s) for $J = J_n$, $B = B_n$ and $J = 5J_n$, $B = B_n$ respectively and a step external load torque (3Nm) is applied at t = 0.5s. for both systems. Once again, the MRRLC strategy shows better performance than the RLC method when an external disturbance is considered as seen in Fig.5.18.

Changing the viscous friction has a very little effect on the transient responses. Thus, the inertia is the dominant parameter of the plant because changing the inertia means changing directly the plant gain with the same rate.

Finally, Fig.5.19, 5.20 and 5.21 show the experimental results for the nominal parameters, five times of the nominal parameters ($J = 5J_n$ and $B = 5B_n$) and an external step load torque (3 Nm)

consideration respectively. As seen in Fig.5.19, for the nominal conditions, both control methods result in the same control performances as required in order to have a fair comparison. However, the ripples in the torque demand of the MRRLC strategy have been increased due to the additional gain K_e. This will be discussed in Section 5.5.3. The robust performance of the MRRLC method comparing to the RLC method is clearly seen in Fig.5.20 and 5.21. Since the torque demand is limited with an anti-windup mechanism, there is no control when the torque demand saturates. Therefore, in the practical system, the MRRLC method is supplemented with the condition :

if the torque demand is saturating or S is too large then $S_{ref}(k) = S(k)$ *until the torque demand becomes less than the torque limit.*

For Fig.5.19 - 5.21, the parameters used in the experimental implementations are shown in Table 5.3.

Table 5.3 Parameters used in the experimental implementation

J_n	B_n	λ	α	K	K_e
0.0035 Kgm^2	0.0007 Nms	25 s^{-1}	40 s^{-1}	$0.1*K_m$	$0.5*K_m$

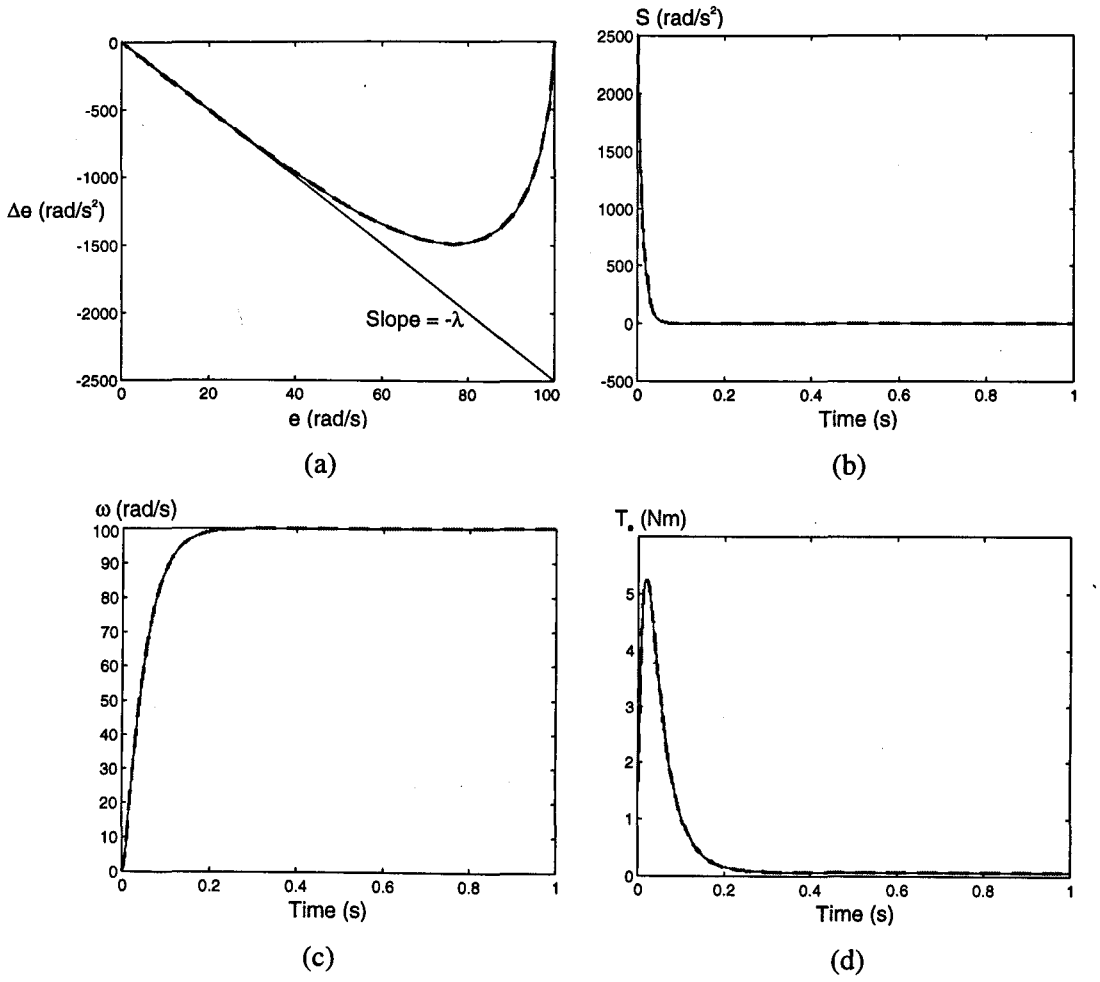


Figure 5.16 Simulation results for $J = J_n$ and $B = B_n$

(a) Phase planes (b) Variations of S

(c) Speed responses (d) Torque responses

(Tick dashed : RLC , Thin continuous : MRRLC)

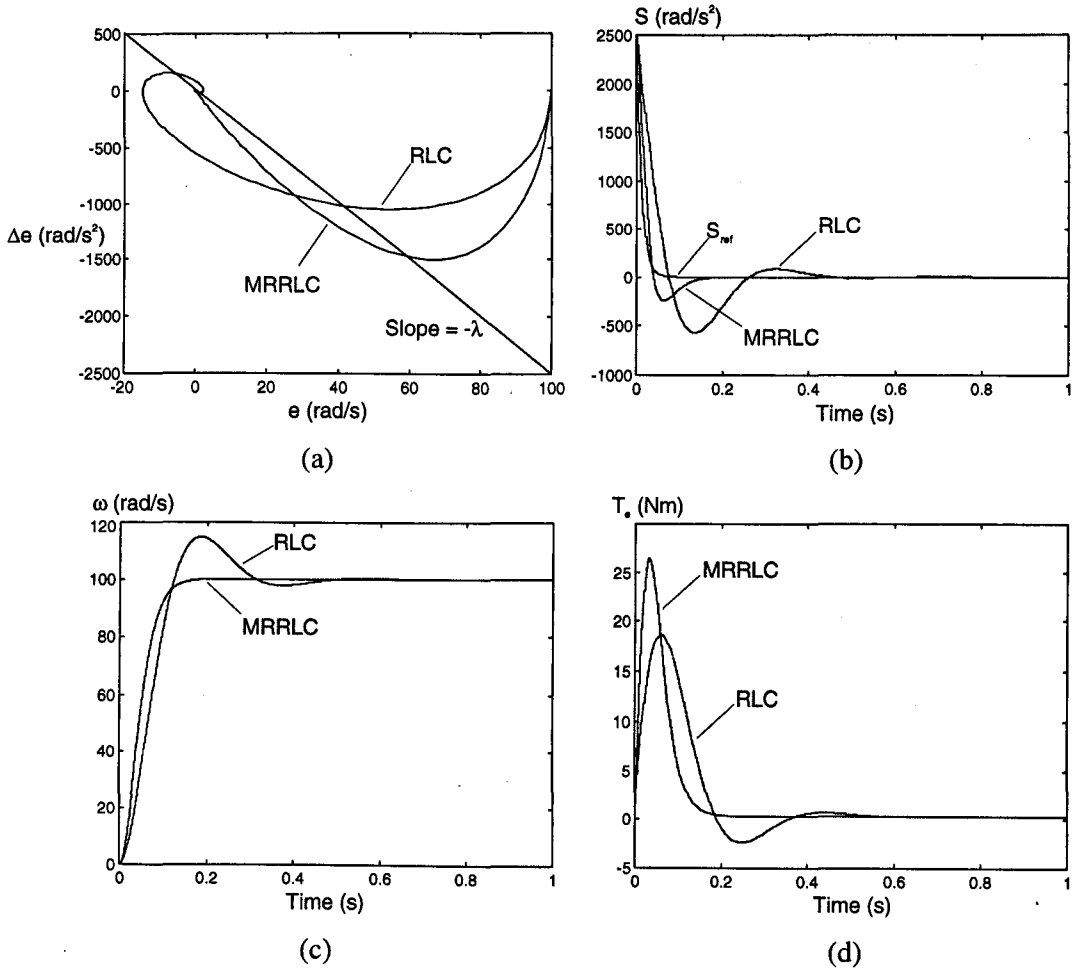


Figure 5.17 Simulation results for $J = 5J_n$ and $B = 5B_n$

(a) Phase planes (b) Variations of S

(c) Speed responses (d) Torque responses

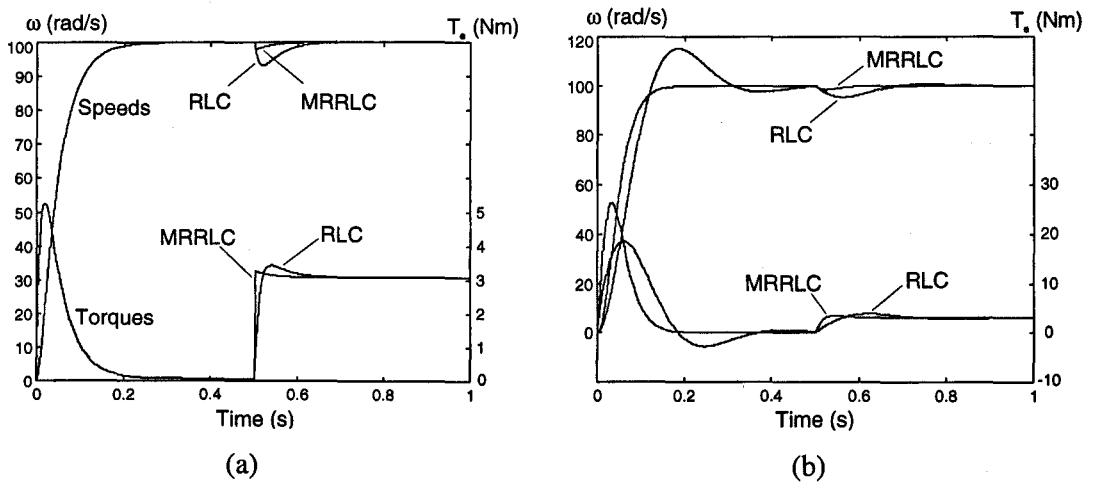


Figure 5.18 Speed and electrical torque responses to a step load torque (3 Nm) at t = 0.5s

(a) $J = J_n$ and $B = B_n$

(b) $J = 5J_n$ and $B = B_n$

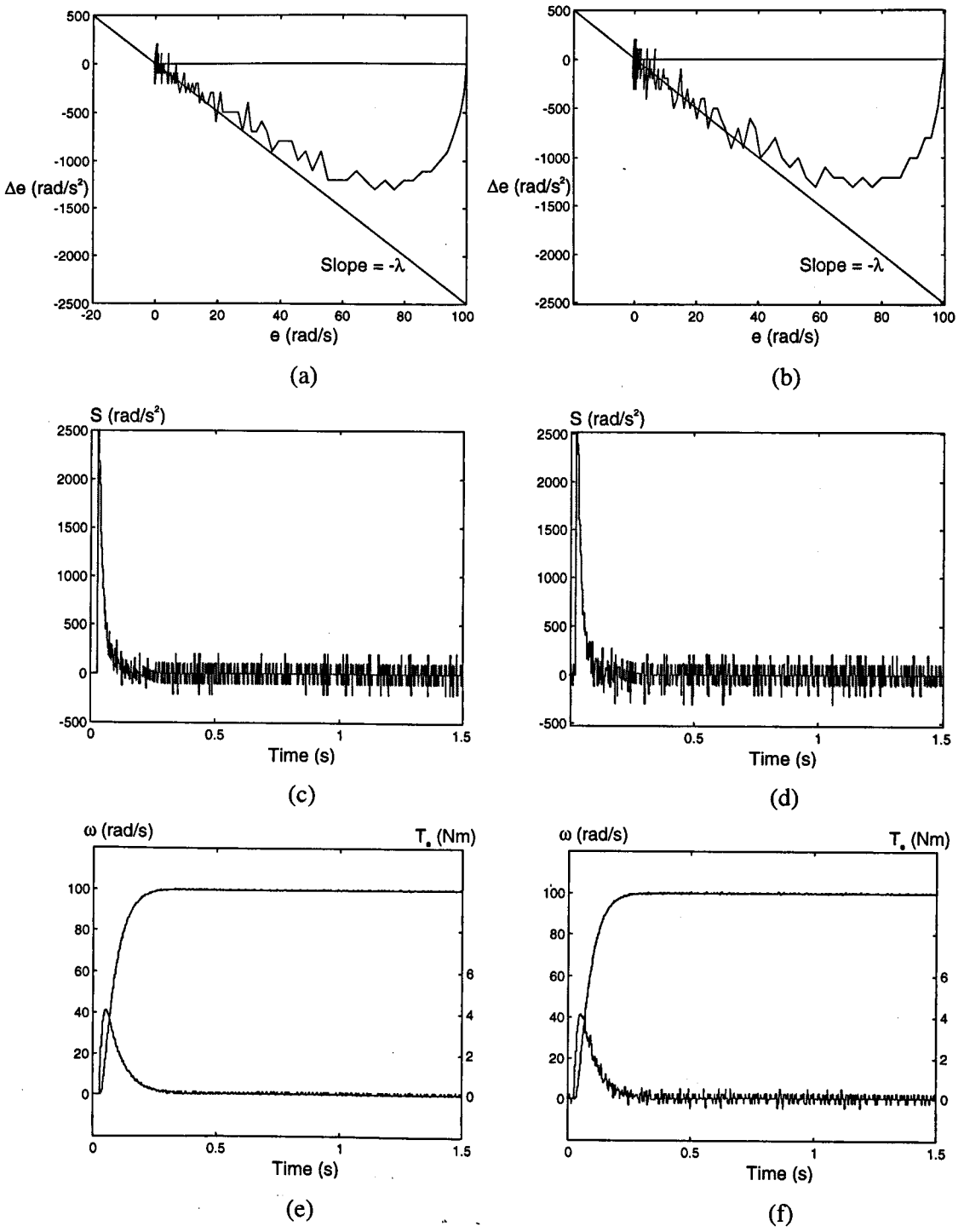


Figure 5.19 Experimental results for $J = J_n$ and $B = B_n$

(a), (c) and (e) : RLC

(b), (d) and (f) : MRRLC

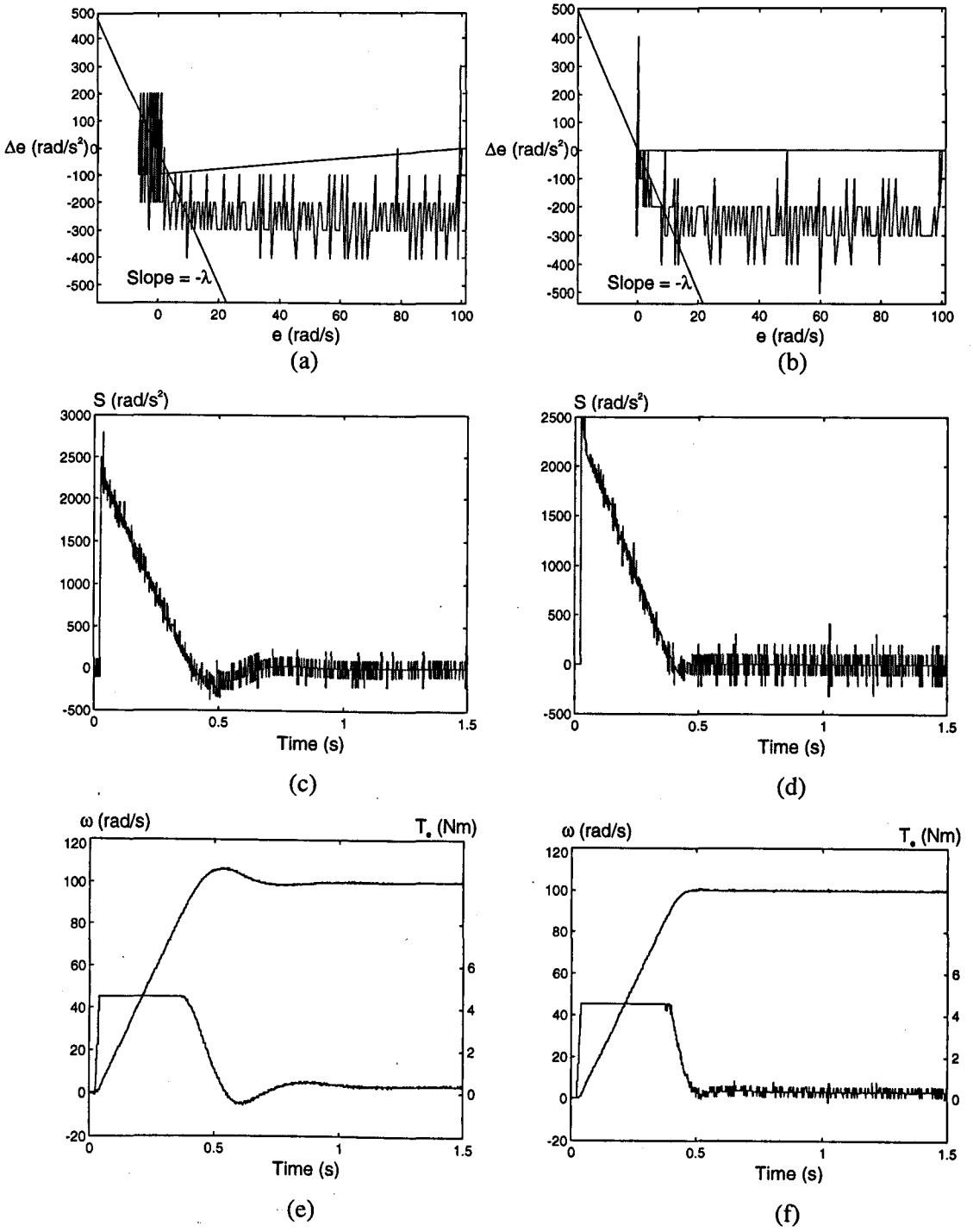


Figure 5.20 Experimental results for $J = 5J_n$ and $B = 5B_n$

(a), (c) and (e) : RLC

(b), (d) and (f) : MRRLC

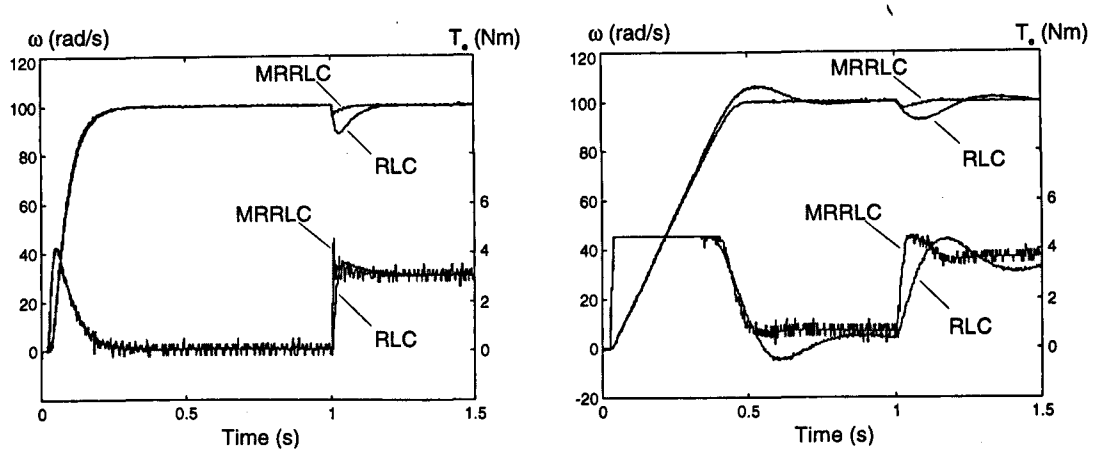


Figure 5.21 Experimental speed responses and electrical torque demands
(A step load torque, 3 Nm, is applied at t = 1s)
(a) $J = J_n$ and $B = B_n$
(b) $J = 5J_n$ and $B = B_n$

In this section, a reference S trajectory (derived using the reaching law designed for the nominal system parameters) is compared with the real S trajectory in order to get information about the system parameter variations and external disturbances. The error between the real and reference S is simply multiplied by a gain (K_e) and added to the controller output to illustrate that this error can be used in the compensation of the parameter variations and external disturbances. It should be noted that the MRRLC approach provides only a relative robustness depending on the selection of the value of K: in Section 5.5.3, it will be shown that $K+K_e$ is limited by the encoder resolution noise, i.e., $K+K_e \leq K_{nlim}$. If K is set to K_{nlim}, the MRRLC approach can not provide more robust performance than the RLC method since K_e will be zero due to the noise limitation. However, in Section 5.6, it will be shown that a controller designed using the MRRLC approach together with the FLC approach provides more robust performance than the RLC method in which K is set to the maximum possible value (K_{nlim}) determined by the noise. It should be remembered that one of the main objectives of this chapter was to combine the SMC (RLC) and FLC approaches in a common framework in order to use the advantages of both methods for the robust control design. In the final robust controller design procedure presented in Section 5.6, the MRRLC approach is basically used to provide information about the parameter variations and external disturbances for the FC to take an appropriate control action by considering the noise limitation.

5.5.3 Noise Limitation

In practical speed control systems, the ripples on the torque demand usually appear due to the speed encoder resolution. There are also other factors causing the noise such as drive torque harmonics, feed through onto the speed, A/D resolution and the closed loop controllers. As it can be seen from Fig.5.19f, the ripples on the torque demand have been increased compared to Fig.5.19e because there is an additional gain K_e in the loop path and this gain directly increases the ripples. If the magnitude of these ripples is too large, the unmodelled high frequency mechanical dynamics may be excited [9,35] and this usually appears as mechanical vibration on the shaft of the rig. This type of vibrations is of course undesirable for most practical applications. Actually, the magnitude of the ripples can be estimated approximately by simply considering the speed encoder resolution in the steady state. if the speed is calculated by counting pulses over the sampling time T_s, then the speed encoder resolution is given by

$$\omega_{res} = \frac{2\pi}{NT_s} \tag{5.61}$$

where N is the number of pulses in per revolution produced by the speed encoder. In the experimental system N = 10000 ppr and T_s = 2.5 ms, thus ω_{res} = 0.25 rad/s (2.4 rpm). The encoder resolution physically means that the speed is measured by the encoder with ±0.25 rad/s error. The change in the torque demand can be written as

$$\delta T_e(k) = K_T T_s \left(\left(\lambda e(k) + \frac{e(k) - e(k-1)}{T_s} \right) (K + K_e) + \frac{e(k) - e(k-1)}{T_s} K_{eq} \right) \tag{5.62}$$

In the steady state, consider the worst case that $e(k) = \omega_{res}$, $e(k-1) = -\omega_{res}$. If the numerical values used in the experimental implementation for Fig.5.19 are substituted in (5.62) (λ = 25 s^{-1} and K_{eq} = 0.0195 which is calculated using (5.44) for the nominal parameters), then (5.62) becomes

$$\delta T_e = 2.1546(K + K_e) + 0.0411 \tag{5.63}$$

Fig.5.22a and b show the ripples on the torque demand of Fig.5.19e and f respectively. The experimental results shown in Fig.5.22 validates the equation (5.63). The calculated magnitudes of the ripples (peak to peak) for Fig.5.22a and b are 0.109 Nm and 0.4526 Nm corresponding to

2.4% and 9.98% of the rated torque. The experimental magnitudes are approximately 0.11 Nm and 0.45 Nm as seen in Fig5.22a and b respectively.

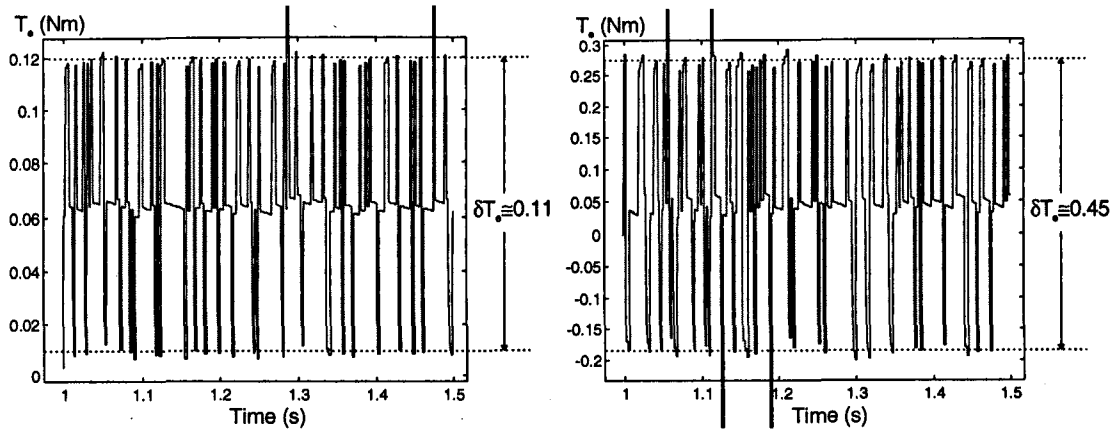


Figure 5.22 The ripples on the torque demand of
(a) Fig.5.19e (b) Fig.5.19f

Although an experimental result is presented in Fig 5.22b which has the torque ripples corresponding to approximately 10% of the rated torque, in most of the practical applications, this may be considered too large. In the experimental rig of this project, the noise percentage allowed by the system is usually around 4-5% before mechanical vibration becomes noticeable. If the noise is 10%, the system can not generally operate. Externally sourced noise also contributes to torque ripple. These noise sources are due to conducted EM emissions through the supply cables. These emissions can be significant due to the prevalence of other power electronic equipment being used in the vicinity. It should be noted that the MRRLC results of Fig.5.19 - 5.21 were taken during a quiet holiday period (when other equipment was not in use); these results thus are not generally repeatable. However, the results of Fig.5.19 to 5.21 can be obtained in practice by the use of the FLC approach discussed in Section 5.6. With 5% rated noise, δT_e of (5.63) is 0.2267Nm giving $K+K_e \leq 0.086$ or $K+K_e \leq 0.27K_m$.

In order to reduce the magnitude of the torque ripples, a simple way is to filter the error e_s (See Fig.5.15) using a low-pass filter, but the use of excessive filtering degrades the control quality. Therefore, as far as the noise is concerned, it is desirable to use lower gains in the steady state. What we need is a mechanism which should be able to choose a control law having a low enough effective gain to avoid the noise problem when $e_s \approx 0$ (no error between S and S_{ref} and

thus system is in the nominal condition) which includes the steady state. This mechanism should also be able to increase the effective controller gain in order to increase the robustness of the controller when there is an error between S and S_{ref}. As it has been shown in Chapter 4, this can be achieved by using the fuzzy logic that can easily implement a mechanism to interpolate between different control laws.

Until this point, the choice of λ has not been discussed in order to reduce the complexity of the discussion. λ is the control bandwidth and it is of course good to have a λ as high as possible. However, in mechanical systems, λ is typically limited by three factors [60]. These are the unmodelled structural resonant frequency, the largest unmodelled time delay in the system and the sampling rate. Therefore, in practice, the maximum available λ depends on the individual applications. In this study, optimisation of λ is not investigated since the main purpose is to develop a robust control method and explain the principles rather than developing an optimal controller. Thus, λ is fixed at a safe value which does not cause noise problem. However, the effect of λ on the torque ripples can be seen by rewriting (5.62) as

$$\delta T_e(k) = K_T T_s \left(\lambda(K + K_e)e(k) + \left(K + K_e + \frac{\lambda J_n - (1+\lambda T_s)B_n}{(1+\lambda T_s)K_T} \right) \Delta e(k) \right) \tag{5.64}$$

which is obtained by using the approximation

$$P_n = \exp(-B_n T_s / J_n) \cong 1 - \frac{B_n T_s}{J_n} \quad \Rightarrow \quad C_n = \frac{T_s}{J_n}$$

so that K_{eq} of (5.44) becomes

$$K_{eq} = \frac{\lambda J_n - (1+\lambda T_s)B_n}{(1+\lambda T_s)K_T} \tag{5.65}$$

From (5.64), λ is seen to be a coefficient of proportionality between e(k) and $\delta T_e(k)$, and since $B_n \approx 0$, also between $\Delta e(k)$ and $\delta T_e(k)$ as well.

In this study, the nominal inertia and viscous friction are chosen naturally as the nominal inertia and viscous friction of the experimental rig (corresponding to the minimum bounds of the

inertia and friction).This is because all the drive motors have their own nominal inertia and friction before connecting to a mechanical load. In addition, let us assume that the controller is designed with a given noise percentage limit for an nominal inertia value higher than the minimum bound (i.e. the nominal inertia is assumed to be a value between the minimum and the maximum inertia bounds). If the inertia of the plant is reduced to the minimum bound, the plant gain will be higher than the nominal plant gain and thus the torque ripples will be multiplied by a higher plant gain and fed back through the speed loop as an input to the controller. This more noisy input to the controller will be multiplied by the controller gains and seen as higher ripples on the output of the controller. Therefore, the torque ripple limit will be exceeded when an inertia less than the nominal value is considered in the plant. Hence, as far as the noise problem is concerned, the minimum inertia and friction should be considered because the plant has its maximum gain when the inertia and friction have their minimum values (see (5.39)).

In the following section, the MRRLC strategy is implemented together with the FLC approach in order to develop a practical robust controller design procedure by considering the noise problem.

5.6 Robust Controller Design using the MRRLC and FLC Methods

In this section, a practical robust controller design procedure based on the MRRLC and FLC approaches is given. The experimental results validating the design procedure are presented. A Sugeno type controller is preferred due to the simplicity of experimental implementation comparing to the Mamdani type controllers. However, a similar design procedure can be derived for Mamdani type controllers using the design approach explained in Chapter 4.

As mentioned in Section 5.5.3, the fuzzy logic will be used to interpolate smoothly between two control laws. Assume that these control laws are u_0 and u_1. The criteria for choosing the control law is the error between S and S_{ref} (e_s) because the existence and the size of this error directly depends on the variation of the plant parameters and the external disturbance. When e_s is around zero, which actually means either there is no perturbation or the system is in steady state, higher controller gains are not required. On the other hand, high controller gains (proportional with the size of e_s) are required in the case of the perturbations. The control law

u_0 is used when there is no error between S and S_{ref} while u_1 is used when the error is large. Thus, the duty of the fuzzy logic is to interpolate between u_0 and u_1 according to the error e_s.

u_0 is chosen by considering the system noise discussed in Section 5.5.3. However, the gains of u_1 can be chosen higher than the noise limit since it is used when the system is away from the steady state.

In this manner, a Sugeno type fuzzy MRRL speed controller design procedure, which can be easily implemented in practice, is summarised as follows:

1) Chose a safe λ (e.g. $\lambda \leq 100\ s^{-1}$) by taking into consideration of the unmodelled mechanical dynamics, maximum time delay in the system and the sampling time [60].

2) Define the control laws u_0 and u_1 as

$$u_0(k) = K_0 S(k) + K_{eq0}\Delta e(k) \tag{5.66}$$

$$u_1(k) = K_1 S(k) + K_{eq1}\Delta e(k) + K_e e_s(k) \tag{5.67}$$

if ω_{ref} is not a step demand, add $\Delta\omega_{ref}$ terms to the control laws as shown in (5.41).

a) Calculate K_{eq0} using (5.44) with the nominal parameters.

b) Determine the value of K_0 using (5.62) by considering the torque demand ripples that can be tolerated by the system. For most of the speed control applications, up to 3-6% ripples are usually acceptable. Note that $K+K_e$ in (5.62) should be replaced by K_0 since u_0 is the control law designed for the case $e_s = 0$ and thus K_e does not exist in u_0. In other words, K_0 corresponds to the maximum possible value (K_{nlim}) determined by the noise limitation.

c) Calculate the parameters of u_1 as follows:

$$K_1 = mK_0,\quad K_{eq1} = mK_{eq0} \text{ and } K_e = mK_0$$

which actually means

$$u_1(k) = m\left(u_0(k) + K_0 e_s(k)\right) \tag{5.68}$$

where m can be determined as follows:

It is assumed that the plant inertia and the viscous friction are limited as

$$J_n \le J \le J_{\max} \quad \text{and} \quad B_n \le B \le B_{\max}$$

where $J_{\max} = mJ_n$ and $B_{\max} = nB_n$

For practical servo machine drives, the transient response is generally dominated by the inertia and therefore, only m is used in order to design the parameters of u_1. This is because when there is an error between S and S_{ref}, the size of e_s is dominantly determined by the increase in the plant inertia and any increase in the viscous friction has a negligible effect on e_s.

3) Define the membership functions of e_s as shown in Fig.5.23. It should be remembered that there are three inputs (S, Δe and e_s) to the fuzzy controller. However, the criteria for choosing the control laws is only e_s. Therefore, the other inputs (S and Δe) do not need to be fuzzified since they are only used in the calculations and they do not determine the control law. However, if a full fuzzy logic controller is desired, they can be easily fuzzified and involved in the fuzzy system without any difficulty, but this will only increase the calculation time of the controller output in the microprocessor implementation.

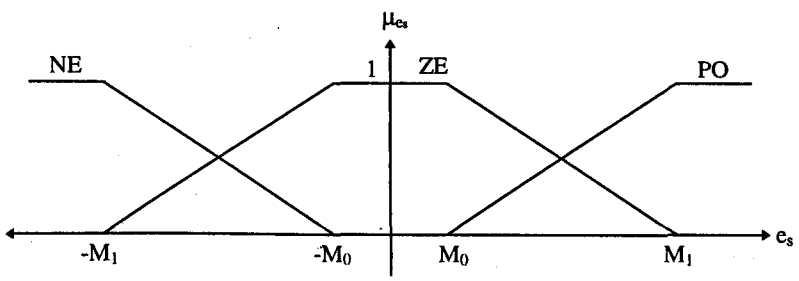


Figure 5.23 Membership functions of e_s

a) Determine the M_0 and M_1 as follows:

The choice of M_0 directly depends on the resolution of e_s. If e_s is not filtered, M_0 can be calculated as

$$M_0 = \lambda\omega_{res} + \frac{\omega_{res}}{T_s} \tag{5.69}$$

since $e_s = S - S_{ref}$ and $S = \lambda e + \Delta e$. However, if a carefully designed filter, which will not spoil the information e_s, is used then M_0 can be reduced in order to increase the sensitivity of the controller to the error e_s. Increasing the sensitivity actually means increasing the robustness of the controller.

M_1 can be chosen as

$$M_1 \cong mM_0 \tag{5.70}$$

b) The fuzzy rules are defined as

If e_s is PO then $u = u_1$
If e_s is ZE then $u = u_0$
If e_s is NE then $u = u_1$

4) If the control performance is not very satisfactory, a good performance can be obtained by simply tuning M_0 and M_1 only. For example, if the robustness is not satisfactory then reduce M_1 (and M_0 if necessary). if the controller suffers from the noise, increase M_0 or apply filtering to the error e_s. it should be note that excessive filtering will destroy the information (e_s) about the parameter variations and external disturbances.

A design example implemented in the experimental rig is given below:

1) λ is chosen as 25 s^{-1}.
2) a) K_{eq0} is calculated as 0.0195.
   b) 4% ripple is found acceptable for the experimental system and δT_e is calculated as $0.04*T_{rated} = 0.18136$ Nm and then K_0 is calculated as 0.065 which corresponds to $0.2K_m$.
   c) The inertia and the friction of the plant are assumed as
   $J_n \le J \le 5J_n$ and $B_n \le B \le 5B_n$
   since m = 5, $K_1 = K_e = 0.325$ and $K_{eq1} = 0.0975$.
3) a) M_0 and M_1 are chosen as 32 and 160 respectively.

Fig.5.24, 5.25 and 5.26 shows the experimental results of the controller designed above in comparison with the RLC method for $J = J_n$, $J = 3J_n$ and $J=5J_n$ ($B = B_n$ for all cases) respectively. The RLC is designed for the nominal parameters with 4% maximum acceptable ripple and thus *the control law of the RLC method is* u_0. In this manner, the RLC design can be interpreted as *the best possible or the most robust controller design using the RLC method* for $\lambda = 25s^{-1}$ with 4% torque ripple limitation. As seen in Fig.5.24, both Fuzzy MRRLC (FMRRLC) and RLC methods give identical responses to a 100rad/s step demand as expected because there is no error between S and S_{ref} (the plant has the nominal parameters). The robustness improvement of FMRRLC method can be seen in Fig.5.25 and 5.26 which are the parameter variations cases.

Fig.5.27a and b show the speeds and the electrical torque demands in the case of an external step load torque (3Nm) for $J = J_n$ and $J = 5J_n$ ($B = B_n$ for both cases). Once again, the robustness improvement of the FMRRLC method is seen in Fig.5.27.

Fig.5.28 shows the experimental speed responses to a triangular speed demand (peak to peak ±10rad/s) for both FMRRLC and RLC methods. Same controllers designed above are used, however, $\Delta\omega_{ref}$ terms (see (5.41)) are included in the control laws of both methods since ω_{ref} is not a step demand anymore. In Fig.5.28a, the plant inertia is increased from J_n to $5J_n$ at t = 1.525s. It is seen that both control methods gives exactly same speed responses until t = 1.525s because the plant inertia has its nominal value until that time. However, when the inertia is increased to 5 times of the nominal inertia, the FMRRLC method shows better tracking performance as clearly seen in Fig.5.28b which is the shaded are of Fig.5.28a (zoomed) to have a better view. Fig 5.28c and d show the speed responses in the case of the external step load disturbances with the magnitudes 3Nm and -3Nm respectively applied at t = 0.775s. The better tracking performances of the proposed method are again seen in Fig.5.28c and d.

Finally, the same controllers designed above are also tested for two non-linear loads emulated in the experimental rig. The first non-linear load is an speed dependent inertial load (see Section 3.6.2) which has the torque-speed equation as

$$T_e = \left(J_n + K_j\omega^2\right)\dot{\omega} + B_n\omega \tag{5.71}$$

where $K_j = 1*10^{-6}$ which is chosen so that the total effective inertia ($J_n + K_j\omega^2$) becomes approximately $4J_n$ when the speed reaches the steady state value of 100rad/s. Fig.5.29a shows the experimental speed responses and the electrical torque demands for both controllers, where the reference input is a step demand (100rad/s).

The second non-linear load is a Watt governor which is explained in Section 3.6.5 and has the torque-speed equation as

$$T_e = J_{ef}\dot{\omega} + B_{ef}\omega \tag{5.72}$$

where $J_{ef} = J_n + 2m\ell^2 \sin^2\theta$ and $B_{ef} = B_n + 2m\ell^2\dot{\theta}\sin(2\theta)$ are the effective inertia and friction seen by the motor electrical torque. Fig.5.29b shows the speed responses and electrical torque demands for both controllers when a 50rad/s step reference input is applied. The parameters of the Watt governor are m = 0.1kg, ℓ = 0.15m and B_o = 0.15Nm (see Section 3.6.5 for details). The performance improvements of the FMRRLC over the RLC method can be seen in Fig.5.29 for both non-linear loads. It should be remembered that both controllers are initially designed for the nominal values J_n and B_n to give exactly same response (see Fig.5.24) and then non-linearity is introduced to compare the performances of the controllers.

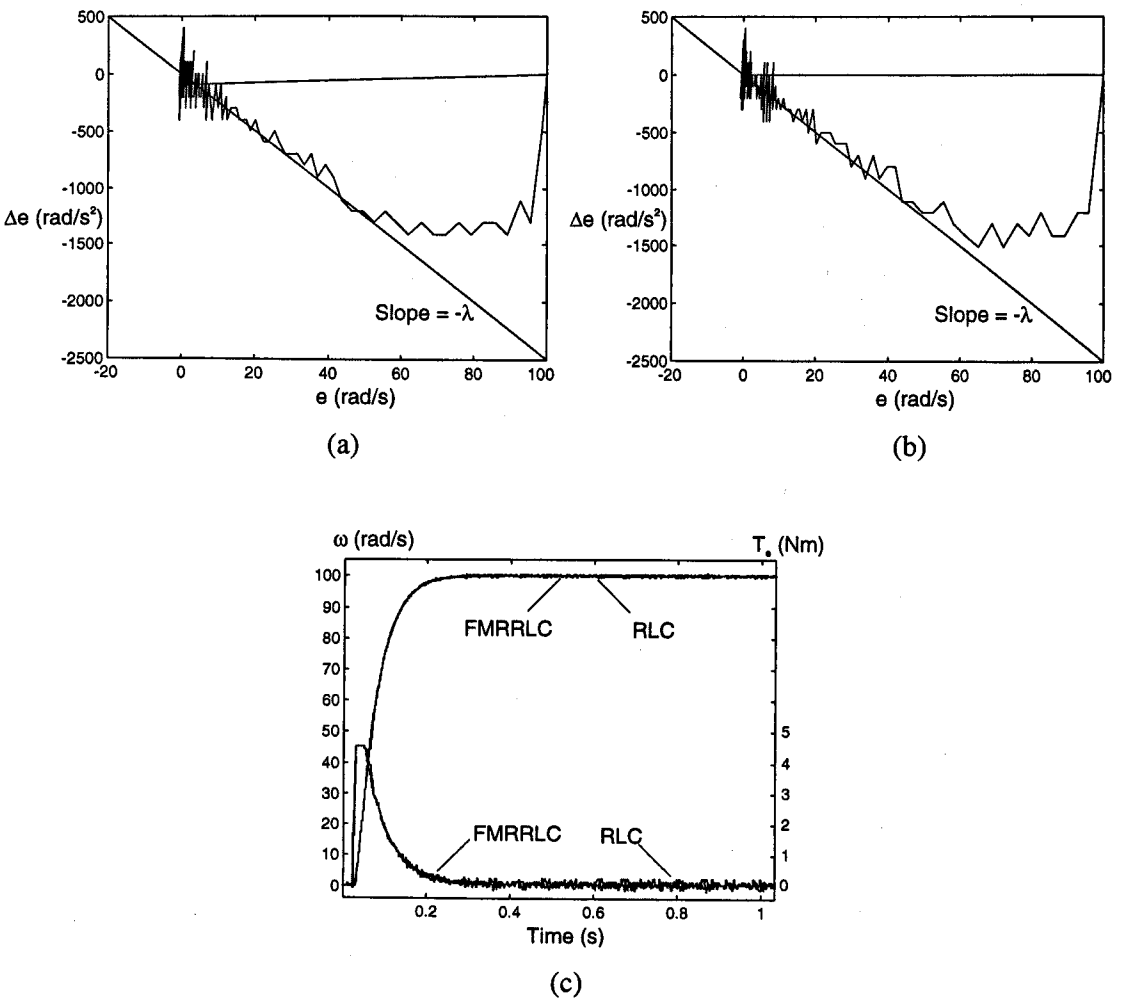


Figure 5.24 Experimental results for $J = J_n$ and $B = B_n$

(a) Phase plane for RLC

(b) Phase plane for FMRRLC

(c) Speed response and electrical torque demands for both RLC and FMRRLC

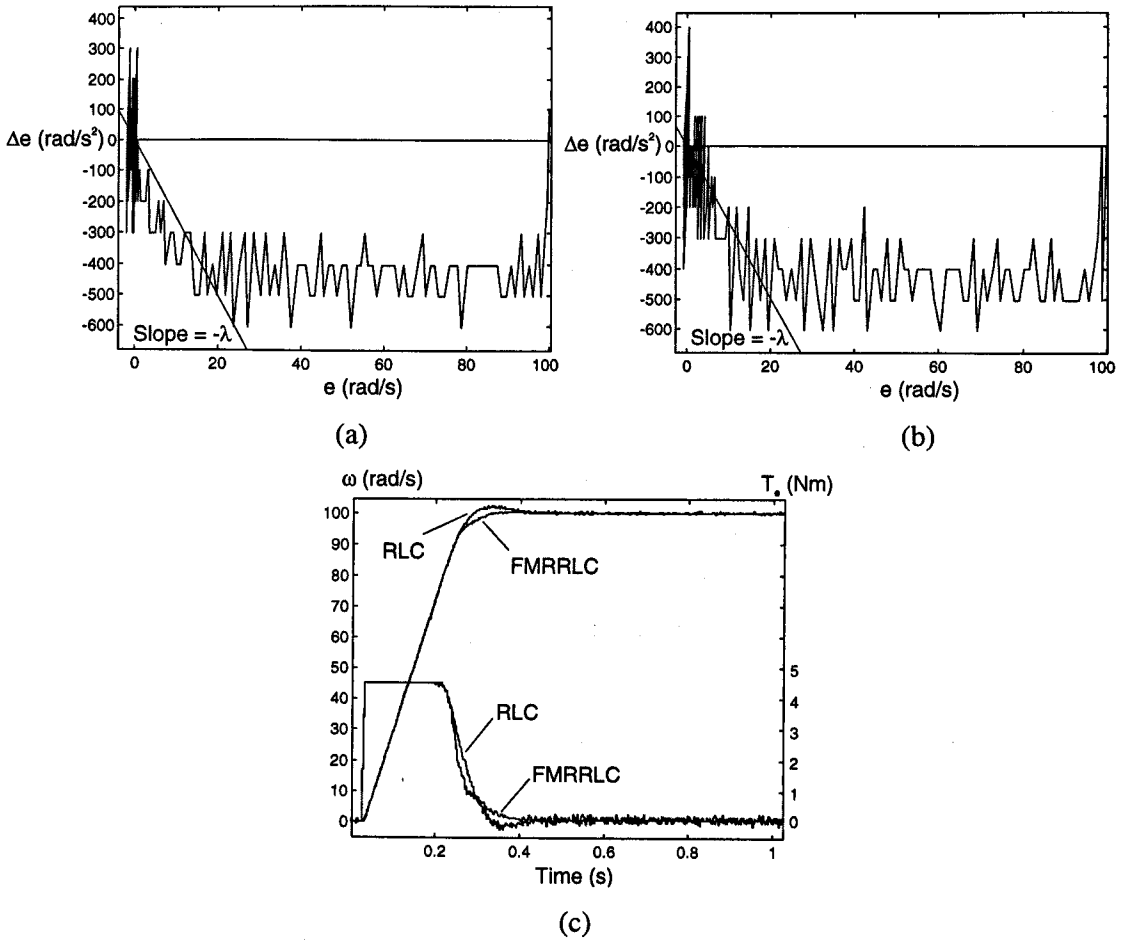


Figure 5.25 Experimental results for $J = 3J_n$ and $B = B_n$

(a) Phase plane for RLC

(b) Phase plane for FMRRLC

(c) Speed response and electrical torque demands for both RLC and FMRRLC

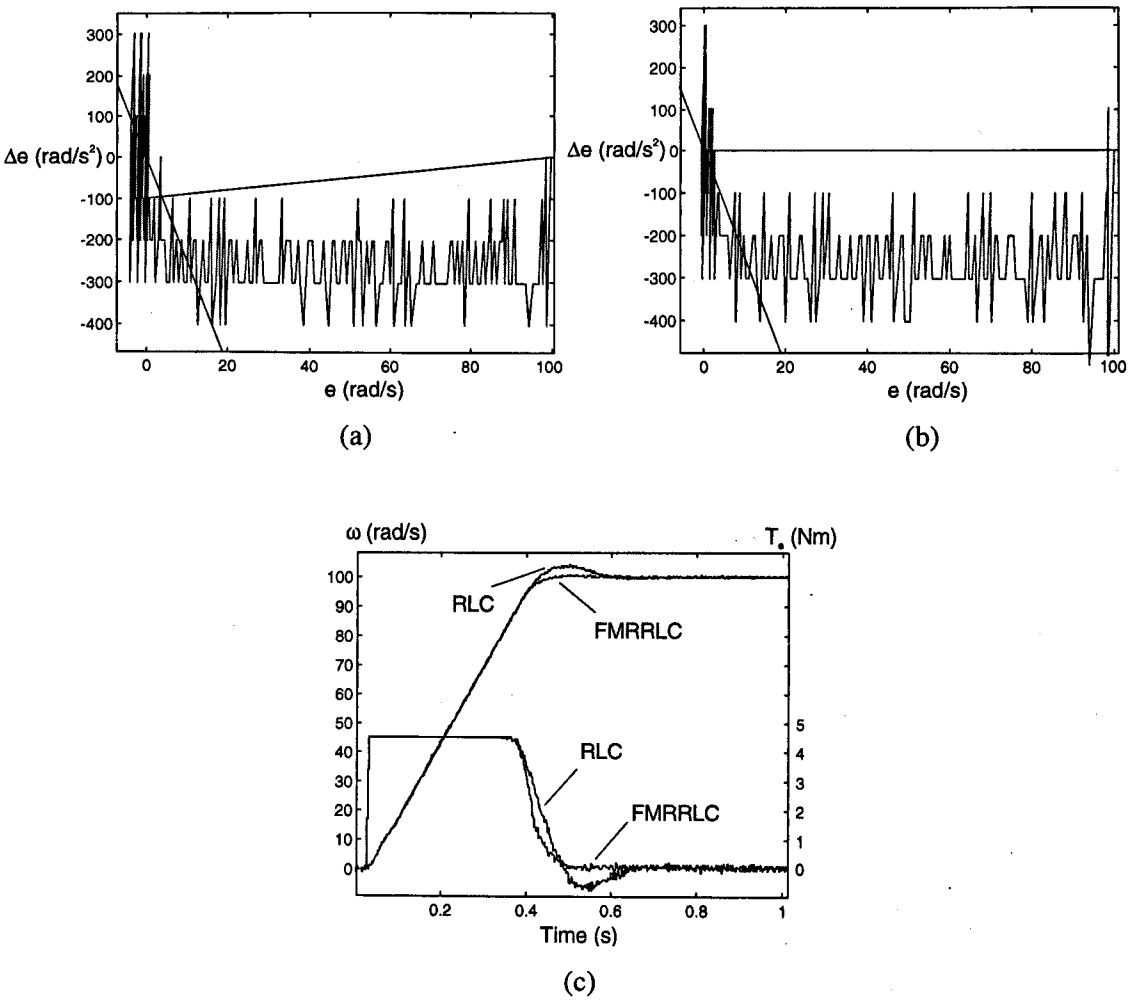


Figure 5.26 Experimental results for $J = 5J_n$ and $B = B_n$

(a) Phase plane for RLC

(b) Phase plane for FMRRLC

(c) Speed response and electrical torque demands for both RLC and FMRRLC

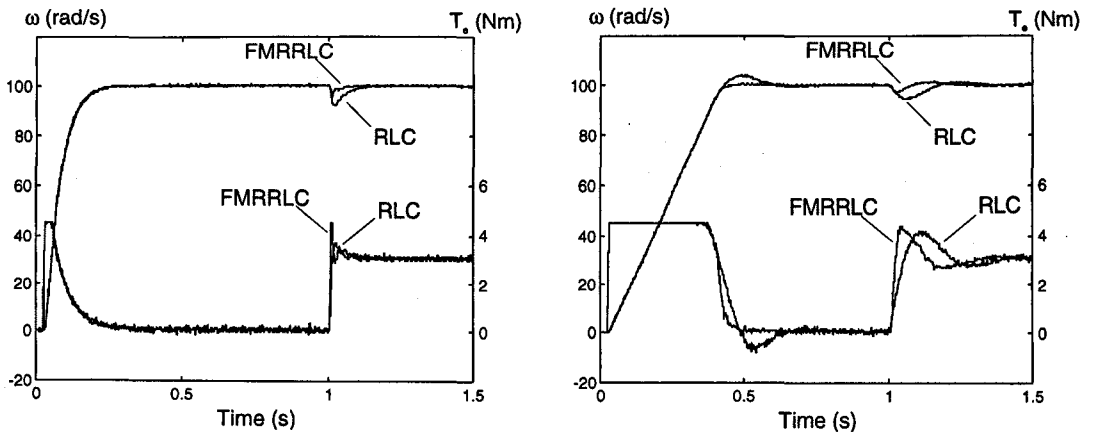


Figure 5.27 Experimental speed responses and electrical torque demands

(A step load torque, 3 Nm, is applied at t = 1s)

(a) $J = J_n$ and $B = B_n$

(b) $J = 5J_n$ and $B = B_n$

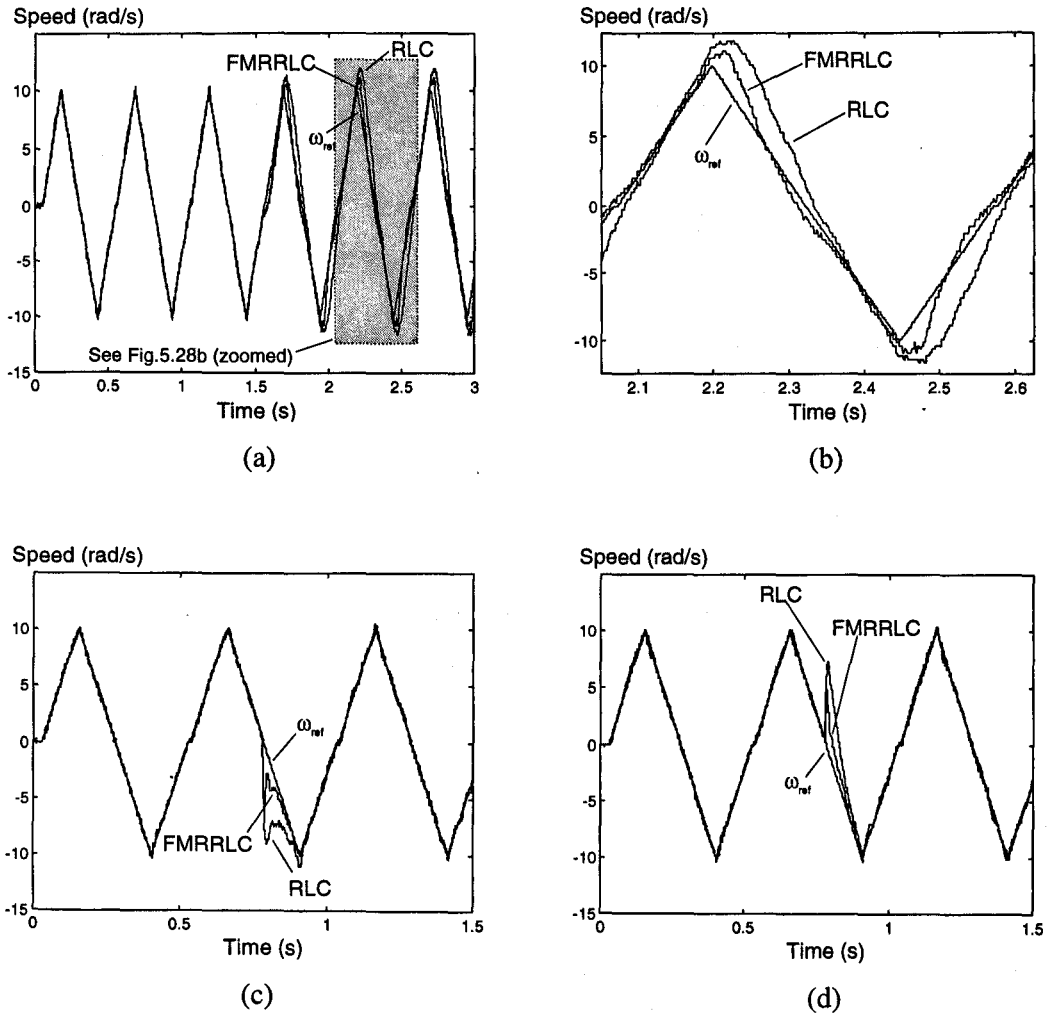


Figure 5.28 Experimental speed responses to an triangular speed demand for the cases

(a) The plant inertia is increased from J_n to $5J_n$ at t = 1.525 s (B = B_n)

(b) Shaded area of Fig.5.28a (zoomed)

(c) An external 3Nm step load torque is applied at t = 0.775 s (J = J_n, B = B_n)

(d) An external -3Nm step load torque is applied at t = 0.775 s (J = J_n, B = B_n)

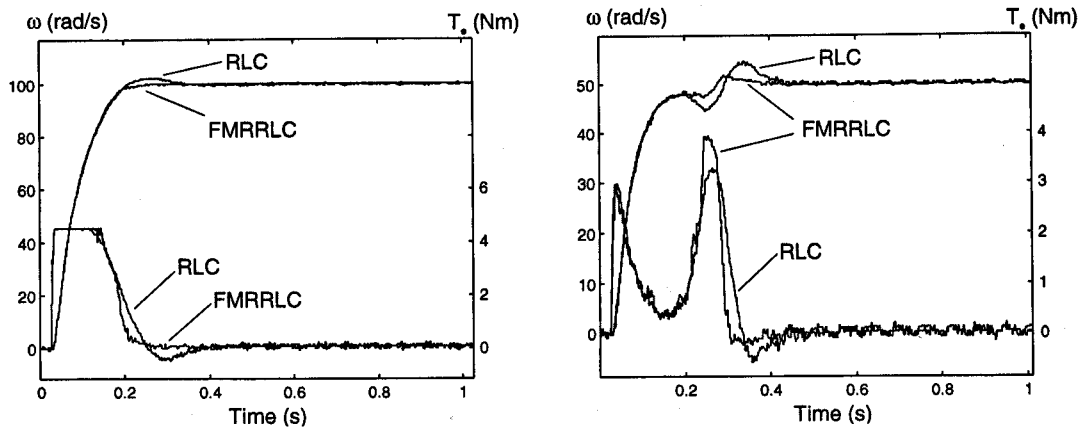


Figure 5.29 Experimental speed responses and electrical torque demands for non-linear loads
(a) Speed dependent inertial load
(b) Watt governor

In this section, the experimental results obtained by using the proposed control method (FMRRLC) have been presented in comparison with the RLC method. The controller based on the RLC method was designed according to a practical noise criteria which basically restricts the controller gains. Since PI controllers are the most widely used controllers in the speed control applications, it may be meaningful to show the *equivalence* between a PI controller and a controller designed using the RLC method. This equivalence is derived under the assumption that the speed demand is a step function (i.e. $\Delta\omega_{ref} = 0$). Otherwise, such an equivalence can not be obtained since the control law of the RLC method will contain some $\Delta\omega_{ref}$ terms which do not exist in a PI controller. If the transfer function of a PI controller is given by

$$G_{PI}(z) = \frac{I_q(z)}{e(z)} = k_p + \frac{zT_s}{z-1}k_i \tag{5.73}$$

where k_p and k_i are the proportional and the integral gain constants respectively, then the output of the PI controller can be written as

$$I_q(k) = I_q(k-1) + T_s\Delta I_q(k) \tag{5.74}$$

where

$$\Delta I_q(k) = k_i e(k) + k_p \Delta e(k) \tag{5.75}$$

and Δ operator is defined by (5.34). On the other hand, for the RLC design, the output of the integrator (see Fig.5.8) can be written as

$$I_q(k) = I_q(k-1) + T_s u(k) \tag{5.76}$$

By equating (5.74) to (5.76) and using (5.75) and (5.42), the PI controller gains can be expressed in terms of the RLC design parameters as

$$\begin{aligned} k_p &= K + K_{eq} \\ k_i &= K\lambda \end{aligned} \tag{5.77}$$

where K and K_{eq} are given by (5.43) and (5.44) respectively. Hence, in this manner, the controller designed using the RLC method becomes equivalent to a PI controller. However, it should be remembered that this equivalence is only valid when the ω_{ref} is a step demand.

5.7 Conclusions

In this chapter, the Sliding Mode Control (SMC) strategy has been investigated with the purpose of developing a robust speed controller design procedure. For the speed control applications, a SMC design approach called Reaching Law Control (RLC) method was found more appropriate than the classical SMC with BL design technique. The digital implementation of the control strategy has been considered since it is more convenient for the practical applications.

A new method called MRRLC has been developed in order to use in the robust control strategy. It has been seen that the robustness is restricted by the system noise in the practical applications. This restriction appears as a limitation on the controller gains. A best possible RLC can be designed with the known gain limitation for a fixed λ. In this case, the fuzzy logic has been employed in the implementation of the MRRLC approach and a further improvement has been obtained over the RLC method. A design procedure has been presented for the robust controller based on the MRRLC and the FLC approaches. The main aim in the design

procedure was to keep the controller design as simple and as algorithmic as possible in order to allow an easy practical implementation.

It should be noted that the main objective of this chapter was to investigate the SMC approach for the speed control systems and to develop a robust controller design procedure based on the SMC and the FLC approaches. The optimisation of λ was beyond the scope of this study. A further work can be done to optimise the whole controller performance.

Chapter 6

Conclusions

6.1 An Overview of the Project Results and Discussions

The first objective of the project was to develop and implement a high performance dynamometer control strategy in order to provide desired linear and non-linear mechanical loads for the experimental validations of the motor drive control methods. The discrete time implementation of the conventional inverse model approach was analysed and it was shown that this method suffers from the stability and noise problems. It was noted that the system might be stabilised using a digital filter, but this would violate the dynamic structure of the desired mechanical load and thus the emulation would give totally erroneous results if used in a closed loop control system. Hence, a new dynamometer control strategy, based on speed tracking and torque feed-forward compensation, was developed and successfully implemented in the experimental system. The emulation was placed in a closed loop speed control system and the experimental results were compared with the corresponding ideal simulated results for the experimental validation of the dynamometer control strategy. The comparisons have shown very good agreements for a variety of linear and non-linear mechanical load models.

The emulation bandwidth is basically limited by the inner current control loops and the system encoder resolution noise. The closed loop bandwidth of the current control loops was approximately 200Hz and thus this limits the emulation bandwidth to approximately 50Hz. This means that the emulation will be effective for closed loop speed bandwidth of, say, 25Hz. PI speed controllers have been evaluated experimentally with a closed loop natural frequency of 10-15Hz (noise and delay effects limit the closed loop speed bandwidths greater than 15Hz). Comparisons between the experimental and simulated systems have shown good agreement for this bandwidth. Therefore it can be assumed that the bandwidth of the emulation is much higher than 15Hz.

The emulation required the drive motor torque reference signal. This was not a restriction given the aim of this work to provide a test-bed for motor drive control strategies. If a torque reference signal is not available, it is recommended, where possible, that the motor drive voltages and currents be measured and fed to an electrical torque observer based on the model equations of the drive machine. It is felt that errors in the estimated torque would be less problematic than the discretization and noise effects arising from the inverse dynamics.

It should be noted that in the previous dynamic load emulation studies [23-28], only simulation results of the continuous time systems have been presented. It is believed that this thesis and the papers [64-67] arising from this study are the only studies reporting such a high performance *experimental* load emulation results.

The second objective of the project was to investigate the Fuzzy Logic Control (FLC) and the Sliding Mode Control (SMC) approaches in order to develop a simple, algorithmic and practical robust control design procedure for industrial drive control systems. The target was to combine the FLC and SMC methods in a common framework in order to use the advantages of both methods while keeping the complexity of the control structure as small as possible.

Since the classical linear controllers are still the most widely used controllers in industrial drive control applications, the equivalence between fuzzy and classical linear controllers was first derived in order to establish a bridge between fuzzy and classical controller design approaches. It was also noted that the equivalence generates an automatic design procedure for Fuzzy Controllers (FCs) and helps to obtain fair comparisons between fuzzy and linear controllers. More importantly, it was shown that the equivalence can be used to design robust FCs for a class of non-linear deterministic systems. However, it was concluded that this design approach can not be directly used for non-deterministic systems since it requires information about the system non-linearity and parameter variations. For example, consider a speed drive control system with inertial and frictional variations (this is one of the most common problems in drive control applications). If the values of the inertia and friction are known accurately, a simple PI controller can be designed to control the system satisfactorily. However, in most cases, these parameters either are unknown, not measurable or change during the operation. For speed control systems, the usual inputs to the FC are the speed error and the change of error. Unfortunately, these input variables do not provide sufficient information about the inertia and friction variations to choose an appropriate control law. It should be noted that a FC can provide robust control only if its decision making mechanism works properly. This means that

the inputs of the FC are very important since the decision, i.e. the control action, is taken according to the values of the input variables. Therefore, a method was required to provide an input variable which can be used in the decision making mechanism to take an appropriate control action in the case of parameter variations and external disturbances. For this reason, and due to the well known robust characteristics of the SMC approach, the robust investigation was moved to the area of the SMC and its variants.

The Reaching Law Control (RLC) approach, which is a new SMC design technique, was investigated and found more appropriate than the classical SMC approach for the speed control applications in which the torque demand limitation is required. More importantly, in order to provide useful information about the system parameter variations and external disturbances, a reference switching function trajectory was derived using the reaching law designed for the nominal parameters of the system. This reference trajectory was compared with the real switching trajectory during the operation and the error was interpreted by the decision making system of the FC which changes the control actions appropriately in the case of parameter variations and disturbances. The noise problem (speed encoder resolutions) had to be taken into account in the design of the final controller in order to have a realistic controller for the practical systems. Finally, a simple and algorithmic robust controller design procedure based on the RLC and FLC approaches was given. The robustness of the proposed control approach was tested for a variety of linear and non-linear mechanical loads provided by the dynamometer. Good output responses were obtained for large parameter variations and external disturbances.

In this study, the value of λ (the slope of the switching line in the phase plane) was kept constant. This is because the main purpose of the robust control approach based on the SMC technique is to force the system to follow a constant switching line determined by λ whatever the plant dynamics are. Therefore, the optimisation or changing the value of λ during operation constitutes a further control problem which can form the basis of further work.

6.2 Further Works

The emulation of further mechanical load dynamics (e.g., resonant loads, backlash, etc.,) was beyond the scope of this study. However, if a compliant load is being emulated, then shaft position can be used as the tracking variable instead of speed and thus the emulation of vibrational loads

may be investigated. In this case, the speed tracking controller (see Chapter 3) may be a PD or P controller since there is already an integrator in the real dynamics (i.e., $G(s) = 1/s(Js+B)$). Of course, a new compensation transfer function should be derived using the same technique illustrated in Chapter 3. Naturally, the frequencies of emulated vibrations will be limited by the bandwidth of the tracking controller.

As far as the robust control research is concerned, further work may be directed to the investigation of on-line tuning of the λ, K and K_{eq} parameters. In other words, an adaptive control structure may be introduced to update these controller parameters using an error minimisation technique for the error between the reference and the real switching functions. Alternatively, in order not to increase the complexity of the control, one simple approach may be to investigate time varying sliding lines by analysing the state trajectories during a controlled degree of chattering. However, there will probably be a compromise between the maximum slope of the sliding line and the degree of allowable noise, and this is likely to make an algorithmic approach difficult.

6.3 Publications

The research carried out resulted in an IEEE journal paper [64], four conference papers [65-68] and a pending journal paper.

Appendix-A

Dynamic Equations for Induction Machines and Field Oriented Control

Field Oriented Control (FOC) or Vector Control effectively "transforms" the AC machine into a "DC machine equivalent" in which a torque producing and field producing current may be defined. Torque and flux can thus be independently controlled as in a DC machine. If we are to describe the alternating current as "DC", we should therefore describe the rotating flux as "stationary". A rotating quantity is only "stationary" in a rotating reference frame. We therefore choose a reference frame called *synchronous frame* which rotates at the excitation frequency ω_e and hence write down the dynamic equations of the machine in the synchronous frame. The real and imaginary axis of the synchronous frame are denoted by the suffixes "d" and "q" respectively. The generalised d-q axis dynamic model of the induction motor in the synchronous rotating frame of reference is give below [1-3] :

$$v_{sd} = \left(R_s + \sigma L_s \frac{d}{dt}\right) i_{sd} - \omega_e \sigma L_s i_{sq} + \frac{M}{L_R}\frac{d}{dt}\phi_{rd} - \omega_e \frac{M}{L_R}\phi_{rq} \tag{A.1}$$

$$v_{sq} = \left(R_s + \sigma L_s \frac{d}{dt}\right) i_{sq} + \omega_e \sigma L_s i_{sd} + \frac{M}{L_R}\frac{d}{dt}\phi_{rq} + \omega_e \frac{M}{L_R}\phi_{rd} \tag{A.2}$$

$$0 = -\frac{M}{L_R} R_R i_{sd} + \left(\frac{R_R}{L_R} + \frac{d}{dt}\right)\phi_{rd} - \omega_{sl}\phi_{rq} \tag{A.3}$$

$$0 = -\frac{M}{L_R} R_R i_{sq} + \left(\frac{R_R}{L_R} + \frac{d}{dt}\right)\phi_{rq} + \omega_{sl}\phi_{rd} \tag{A.4}$$

The symbols represent :

v_{sd}, v_{sq}	d-q axis stator voltages,
i_{sd}, i_{sq}	d-q axis stator currents,
ϕ_{rd}, ϕ_{rq}	d-q axis rotor fluxes,
R_s, R_R	stator and rotor resistance,

L_s, L_R stator and rotor self inductance,

M magnetising inductance,

σ total leakage factor,

ω_e stator angular frequency,

ω_{sl} slip angular frequency.

The torque equation of this system is

$$T_e = \left(\frac{3pM}{L_R}\right)\left(i_{sq}\phi_{rd} - i_{sd}\phi_{rq}\right) \tag{A.5}$$

where p is the number of pole pairs. For field orientation, the d-axis of the rotating frame is aligned with the rotor flux so that $\phi_{rq} = 0$, giving the field orientation equations

$$v_{sd} = \left(R_s + \sigma L_s \frac{d}{dt}\right) i_{sd} - \omega_e \sigma L_s i_{sq} + \frac{M}{L_R}\frac{d}{dt}\phi_{rd} \tag{A.6}$$

$$v_{sq} = \left(R_s + \sigma L_s \frac{d}{dt}\right) i_{sq} + \omega_e \sigma L_s i_{sd} + \omega_e \frac{M}{L_R}\phi_{rd} \tag{A.7}$$

$$0 = -\frac{M}{L_R} R_R i_{sd} + \left(\frac{R_R}{L_R} + \frac{d}{dt}\right)\phi_{rd} \tag{A.8}$$

$$0 = -\frac{M}{L_R} R_R i_{sq} + \omega_{sl}\phi_{rd} \tag{A.9}$$

$$T_e = \left(\frac{3pM}{L_R}\right) i_{sq}\phi_{rd} \tag{A.10}$$

Usually, the rotor flux is written as

$$\phi_{rd} = M i_{mrd} \tag{A.11}$$

where i_{mrd} is the rotor flux magnetising or field current. Introducing the rotor time constant $\tau_R = L_R / R_R$ and substituting (A.11) in (A.8), (A.9) and (A.10) yields

$$\tau_R \frac{di_{mrd}}{dt} + i_{mrd} = i_{sd} \tag{A.12}$$

$$\omega_{sl} = \frac{i_{sq}}{\tau_R i_{mrd}} \tag{A.13}$$

$$T_e = \left(\frac{3pM^2}{L_R}\right) i_{mrd} i_{sq} \tag{A.14}$$

Equations (A.12)-(A.14) are the fundamental equations for vector control. Equation (A.12) is directly analogous to the field equation $v_f = R_f i_f + L_f (di_f / dt)$ of a DC machine. Thus, i_{sd} is the *steady-state field current* and analogous to v_f / R_f. It is directly proportional to the rotor flux level in the induction machine. For this reason, i_{sd} is called "field component" of the stator current. The rotor flux magnetising current i_{mrd} is analogous to the DC motor field current i_f. Obviously, in steady-state $i_{sd} = i_{mrd}$. For 'normal' or 'constant torque' operation of the induction motor, ϕ_{rd} is controlled constant and hence $i_{sd} = i_{mrd}$ is controlled constant. In fact (A.12) is only relevant in field weakening control. Equation (A.14) corresponds to the torque equation $T_{dc} = K i_f i_a$ of the DC machine, where i_a is the armature current of the DC machine. Thus, i_{sq} is the torque producing component of the stator current.

For the implementation of vector control, the rotor flux angle θ_e (defining the position of the rotor flux vector in space) must be known at all times. The rotor flux angle can be obtained either *directly* or *indirectly*. In direct vector control, the rotor flux angle is either measured or derived via an rotor flux observer. An alternative and simpler approach is to employ a technique known as *Indirect Vector Control* where the rotor flux is not directly measured or derived. This strategy imposes vector control in a feed-forward manner. It implicitly aligns the d-axis to the rotor flux vector, by controlling the slip angular velocity, according to (A.13), using the controller reference value of i_{sq} and ϕ_{rd} rather than the actual machine values. It thus relies upon the precise control of the d and q axis currents using fast stator current controllers. A general scheme for such an indirect vector controller employs two quadrature current

controllers for i_{sd} and i_{sq} and controls the inverter frequency according to (A.13). The controller imposes a rotor flux angle θ_e which is implicitly aligned to the d-axis such that

$$\theta_e = \theta_r + \theta_{sl} \tag{A.15}$$

$$\omega_{sl} = \frac{i_{sq}^{ref}}{\tau_R i_{sd}^{ref}} \tag{A.16}$$

where θ_r is the absolute position of the rotor. The torque current reference is derived from the speed controller, whereas the flux current reference under base speed is maintained constant at just under saturation level. The general schematic for the indirect vector control, often called Indirect Rotor Flux Orientation (IRFO), algorithm employed is illustrated in Fig.A.1, where '*' refers to the reference values.

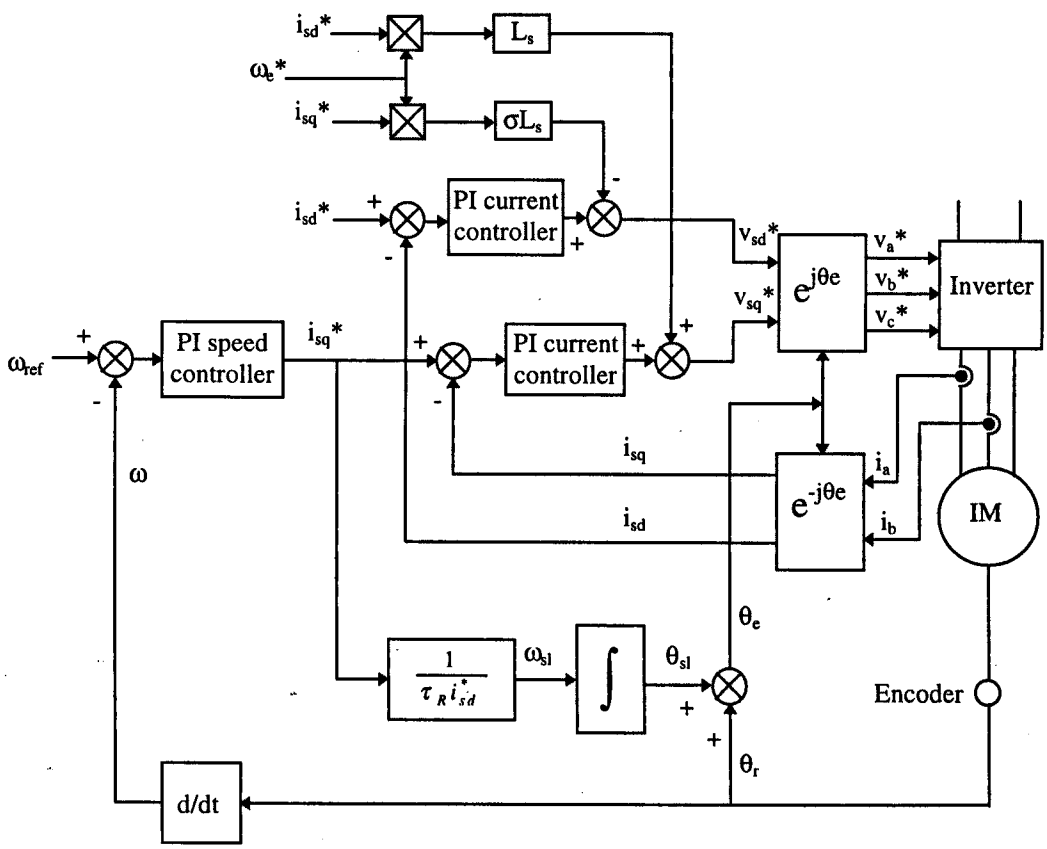


Figure A.1 Speed control of an induction machine using IRFO method

The instantaneous rotor flux position θ_e is defined at any instant by (A.15), where θ_r is the measured rotor position and θ_{sl} is the slip angle demand. The rotor flux angle is then used to transform the instantaneous measured line currents to the field oriented frame of reference, using a 3-2 phase transformation (demodulation) as follows :

$$i_{s\alpha} = \frac{1}{\sqrt{2}} i_a \tag{A.17a}$$

$$i_{s\beta} = \frac{1}{\sqrt{6}} (i_b - i_c) \tag{A.17b}$$

$$i_{sd} = i_{s\alpha} \cos\theta_e + i_{s\beta} \sin\theta_e \tag{A.17c}$$

$$i_{sq} = i_{s\beta} \cos\theta_e - i_{s\alpha} \sin\theta_e \tag{A.17d}$$

The absolute values of these transformed currents reflect the phase rms quantities in the motor and are consistent with the torque equation used. This transformation is abbreviated for the schematic diagram by use of the equivalent complex operator $e^{-j\theta_e}$. The corresponding modulation routines (i.e. transformation of d and q axis values to instantaneous stator reference values) are represented by the complex operator $e^{j\theta_e}$ and give by

$$v_{s\alpha} = v_{sd}^* \cos\theta_e - v_{sq}^* \sin\theta_e \tag{A.18a}$$

$$v_{s\beta} = v_{sd}^* \sin\theta_e + v_{sq}^* \cos\theta_e \tag{A.18b}$$

$$v_a^* = \sqrt{2} v_{s\alpha} \tag{A.18c}$$

$$v_b^* = -\frac{1}{\sqrt{2}} v_{s\alpha} + \sqrt{\frac{3}{2}} v_{s\beta} \tag{A.18d}$$

$$v_c^* = -\frac{1}{\sqrt{2}} v_{s\alpha} - \sqrt{\frac{3}{2}} v_{s\beta} \tag{A.18e}$$

The precise control of the stator currents is achieved using current feedback to provide closed-loop control of d and q axis currents, and is based on the stator dynamic equations (A.6) and (A.7). The current controllers are designed on the basis of the following equations :

$$v_{sd} = \left(R_s + \sigma L_s \frac{d}{dt} \right) i_{sd} - \omega_e \sigma L_s i_{sq} \tag{A.19}$$

$$v_{sq} = \left(R_s + \sigma L_s \frac{d}{dt} \right) i_{sq} + \omega_e L_s i_{sd} \qquad \text{(A.20)}$$

which are derived from (A.6) and (A.7) respectively (assuming $i_{sd} = i_{mrd}$). It can be seen that the right hand side of (A.19) and (A.20) constitute additional coupling terms which must be compensated for at the output of each current controller (see Fig.A.1).

The three-phase voltage demands are derived using (A.18a)-(A.18e). These are then used to produce PWM signals for the inverter (see Chapter 2).

Appendix-B

Interface Circuits

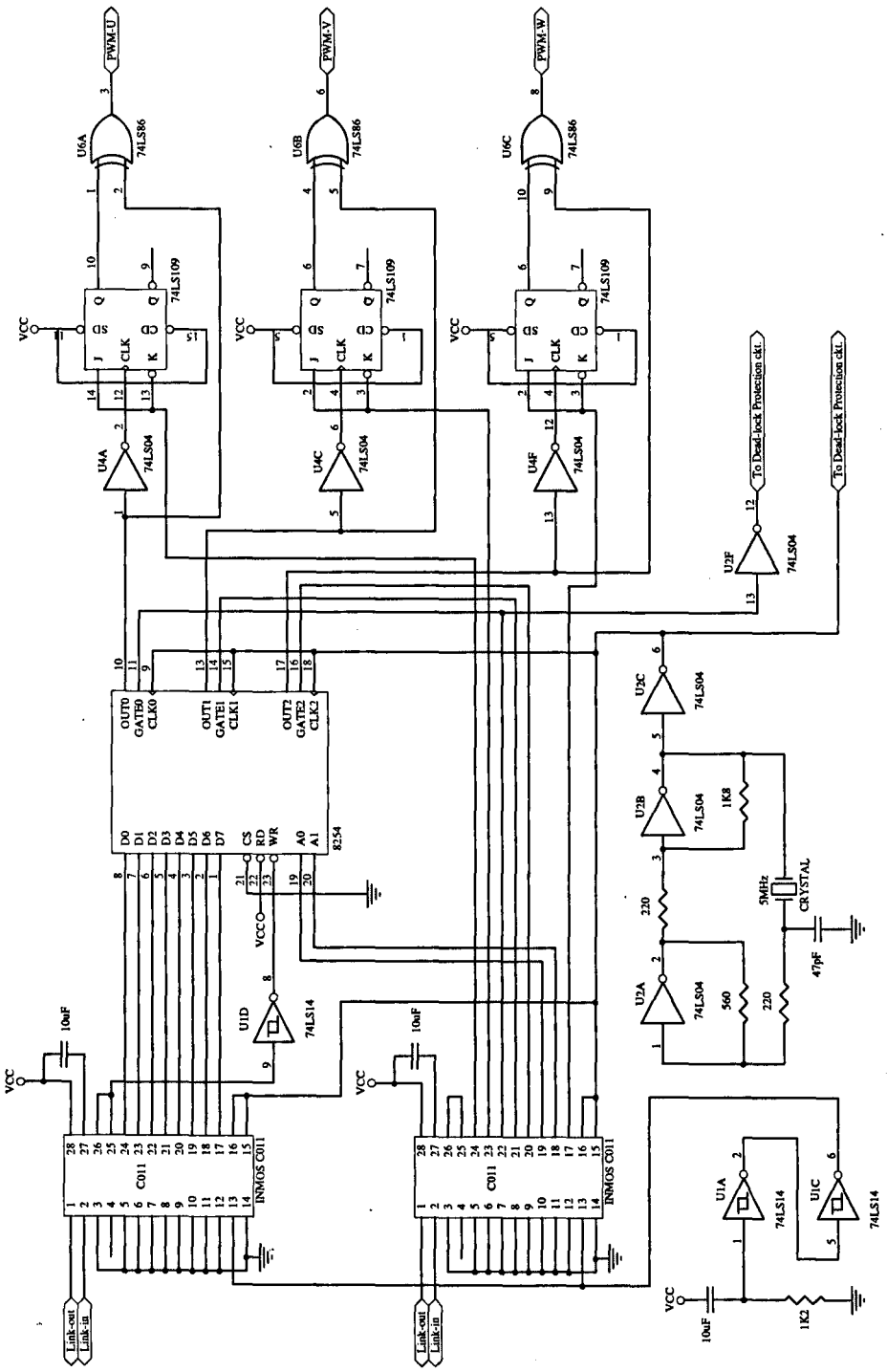


Figure B.1 PWM generation (counter) circuit

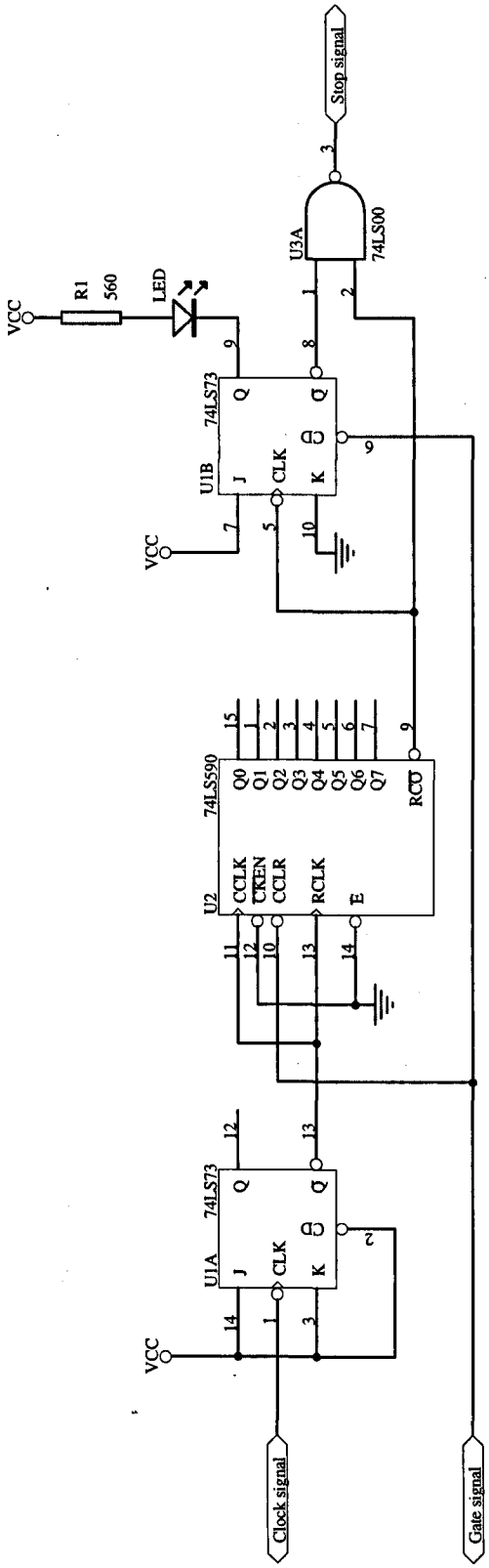


Figure B.2 Dead-lock protection circuit

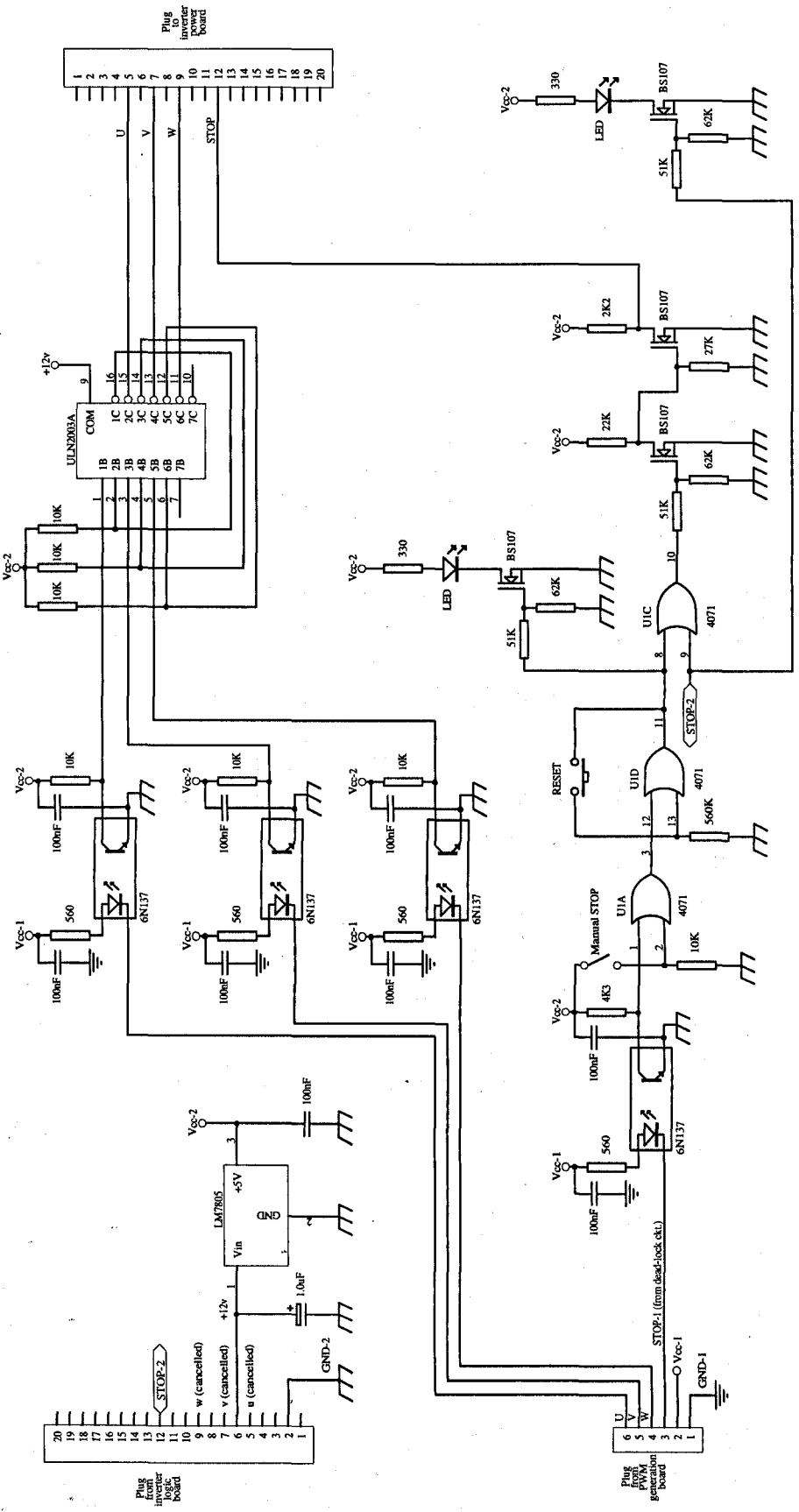


Figure B.3 Inverter interface circuit

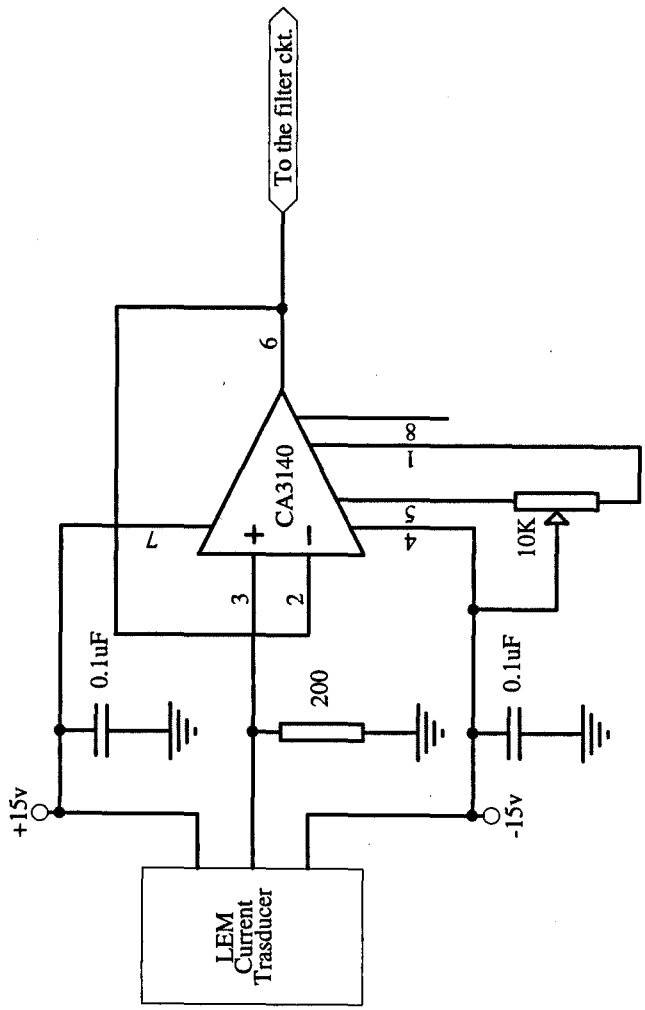


Figure B.4 Current measurement circuit

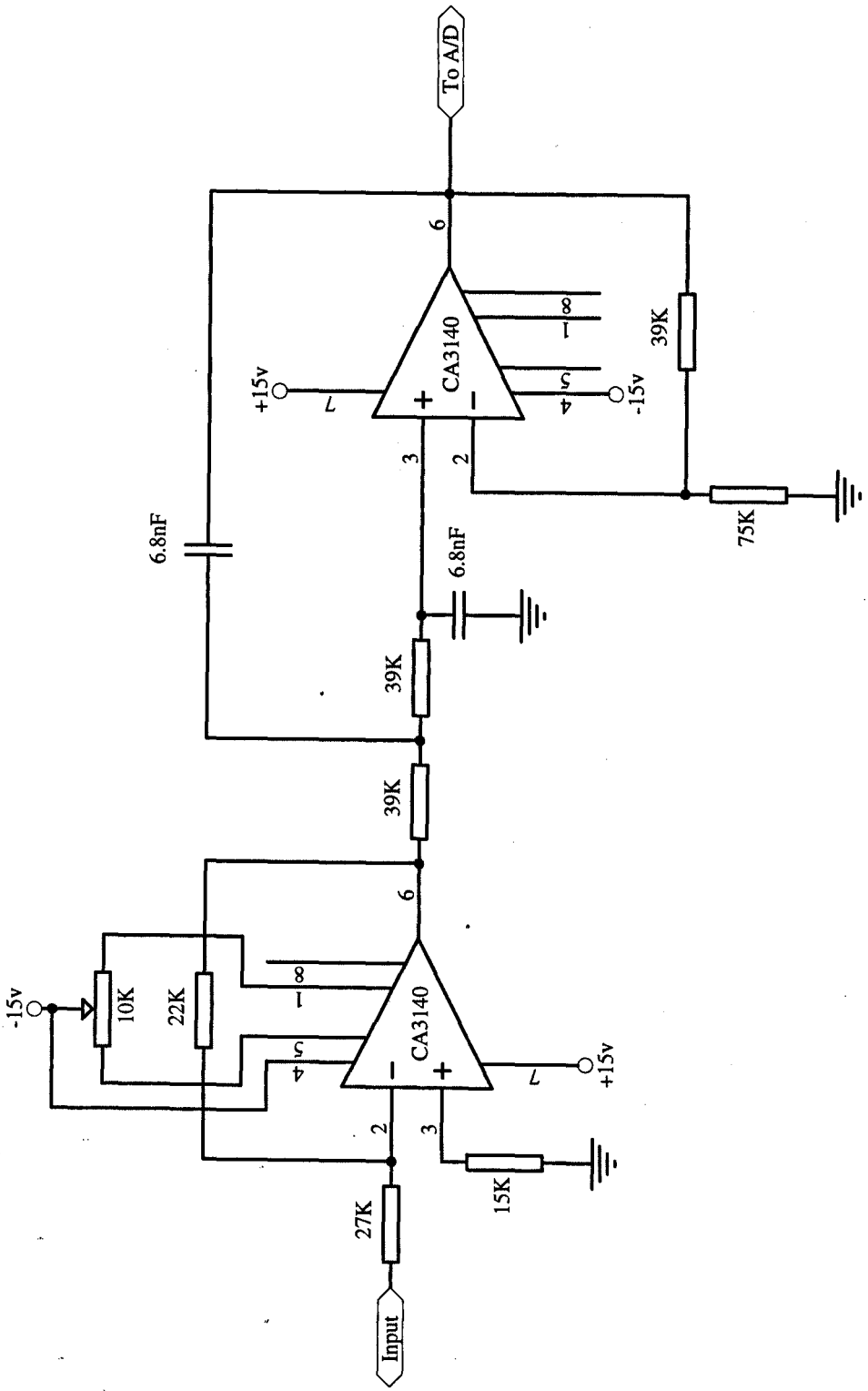


Figure B.5 Filter circuit

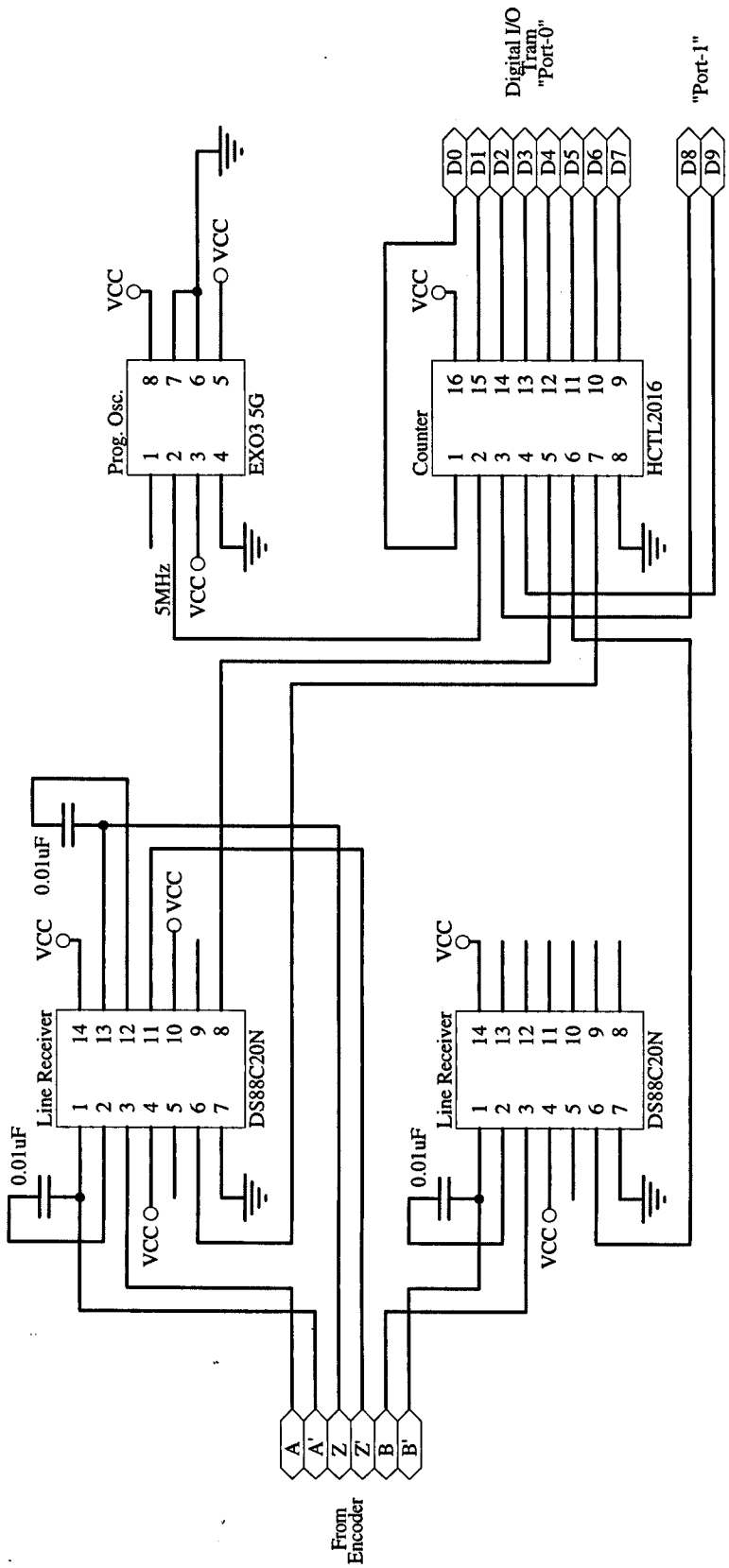


Figure B.6 Encoder interface circuit

Bibliography

[1] W. Leonhard, *Control of Electrical Drives*. Springer-Verlag, 1996.

[2] P. Vas, *Vector Control of AC Machines*. Oxford University Press, 1992.

[3] B.K. Bose, *Power Electronics and AC Drives*. Prentice-Hall, 1986.

[4] B. Shahian and M. Hassul, *Control System Design using MATLAB*. Prentice-Hall, 1993.

[5] G.F. Franklin, J.D. Powell and M.L. Workman, *Digital Control of Dynamic Systems*. Addison-Wesley, 1990.

[6] M. Iwasaki and N. Matsui, "Robust speed control of IM with torque feed-forward control," *IEEE Trans. Ind. Electron.*, vol. 40, no. 6, pp. 553-560, 1993.

[7] C.M. Liaw and F.J. Lin, "A robust speed controller for induction motor drives," *IEEE Trans. Ind. Electron.*, vol. 41, no. 3, pp. 308-315, 1993.

[8] F.J. Lin, C.M. Liaw, Y.S. Shieh, R.J. Guey and M.S. Hwang, "Robust two-degrees-of-freedom control for induction motor servo-drive," *IEE Proc. Electr. Power Appl.*, vol. 142, no. 2, pp. 79-86, 1995.

[9] E.Y.Y. Ho and P.C. Sen, "Control dynamics of speed drive systems using sliding mode controllers with integral compensation," *IEEE Trans. Ind. Appl.*, vol. 27, no. 5, pp. 883-892, 1991.

[10] S.K. Chung, J.H. Lee, J.S. Ko and M.J. Youn, "Robust speed control of brushless direct-drive motor using integral variable structure control," *IEE Proc. Electr. Power Appl.*, vol. 142, no. 6, pp. 361-370, 1995.

[11] T.L. Chern, C.W. Chuang and R.L. Jiang, "Design of discrete integral variable structure control systems and application to a brushless DC motor control," *Automatica.*, vol. 32, no. 5, pp. 773-779, 1996.

[12] K.K. Shyu and H.J. Shieh, "A new switching surface sliding mode speed control for induction motor drive systems," *IEEE Trans. Power Electron.*, vol. 11, no. 4, pp. 660-667, 1996.

[13] F.J. Lin and S.L. Chiu, "Robust PM synchronous motor servo drive with variable structure model-output-following control," *IEE Proc. Electr. Power Appl.*, vol. 144, no. 5, pp. 317-324, 1997.

[14] T.L. Chern, C.S. Liu, C.F. Jong and G.M. Yan, "Discrete integral variable structure model following control for induction motor drives," *IEE Proc. Electr. Power Appl.*, vol. 143, no. 6, pp. 467-474, 1996.

[15] I.C. Baik, K.H. Kim and M. J. Youn, "Robust nonlinear speed control of PM synchronous motor using adaptive and sliding mode control techniques," *IEE Proc. Electr. Power Appl.*, vol. 145, no. 4, pp. 369-376, 1998.

[16] F.J. Lin and S.L. Chiu, "Adaptive fuzzy sliding mode control for PM synchronous servo motor drives," *IEE Proc. Contr. Theory Appl.*, vol. 145, no. 1, pp. 63-72, 1998.

[17] G.C.D. Sousa and B.K. Bose, "A fuzzy set theory based control of a phase-controlled converter DC machine drive," *IEEE Trans. Ind. Appl.*, vol. 30, no. 1, pp. 34-44, 1994.

[18] J.T. Teeter, M.Y. Chow and J.J. Brickley, "A nowel fuzzy friction compensation approach to improve the performance of a DC motor control system," *IEEE Trans. Ind. Electron.*, vol. 43, no. 1, pp. 113-120, 1996.

[19] F.J. Lin, R.F. Fung and Y.C. Wang, "Sliding mode and fuzzy control of toggle mechanism using PM synchronous servomotor drive," *IEE Proc. Contr. Theory Appl.*, vol. 144, no. 5, pp. 393-402, 1997.

[20] C.R. Wasko, "A universal AC dynamometer for testing motor drive systems," in *Proc. IEEE-IAS Conf.*, vol. 1, 1987, pp. 409-412.

[21] A.C. Williamson and K.M.S. Al-Khalidi, "An improved engine testing dynamometer," in *Proc. 4th Int. Conf. Elec. Mach. And Drives*, vol. 1, 1989, pp. 374-378.

[22] P. Sandholdt, E. Ritchie and J.K. Pedersen, "An automatic test system with steady state load emulation for test of electrical drive systems," in *Proc. Int. Conf. Elec. Mach. (ICEM'98)*, vol. 3, 1998, pp. 2071-2076.

[23] E.R. Collins and Y. Huang, "A programmable dynamometer for testing rotating machinery using a three-phase induction machine," *IEEE Trans. Ener. Conv.*, vol. 9, no. 3, pp. 521-527, 1994.

[24] R. W. Newton, R.E. Betz and H.B. Penfold, "Emulating dynamic load characteristics using a dynamic dynamometer," in *Proc. Int. Conf. Power Electron. and Drive Systems*, vol. 1, 1995, pp. 465-470.

[25] R.E. Betz, H.B. Penfold and R.W. Newton, "Local vector control of an AC drive system load simulator," in *Proc. IEEE Conf. Contr. Appl.*, vol. 1, 1994, pp. 721-726.

[26] P. Sandholdt, E. Ritchie, J.K. Pedersen and R.E. Betz, "A dynamometer performing dynamical emulation of loads with non-linear friction," in *Proc. IEEE Int. Symp. Ind. Electron.*, vol. 2, 1996, pp. 873-878.

[27] G.C.D. Sousa and D.R. Errera, "A high performance dynamometer for drive systems testing," in *Proc. 23rd Int. Conf. Ind. Electron., Contr. Inst. (IEEE-IECON'97)*, vol. 2, 1997, pp. 500-504.

[28] J.J. Carrol, D.M. Dawson and E.R. Collins, "A non-linear control technique for the development of a computer controlled dynamometer", in *Proc. Dynamic Sys. and Contr. Division, American Soc. of Mech. Eng.*, vol. 53, 1993, pp. 31-36.

[29] L.A. Zadeh, "Fuzzy sets," *Informat. Contr.*, vol. 8, pp. 338-353, 1965.

[30] P. Vas, A.F. Stronach and M. Neuroth, "Full fuzzy control of a DSP-based high performance induction motor drive," *IEE Proc. Contr. Theory Appl.*, vol. 144, no. 5, pp. 361-368, 1997.

[31] Z. Ibrahim and E. Levi, "A detailed comparative analysis of fuzzy logic and PI speed control in high performance drives," in *Proc. IEE Conf. Pow. Electron. Var. Speed Drives*, 1998, pp. 638-643.

[32] W.G. da Silva and P.P. Acarnley, "Fuzzy logic controlled DC motor drive in the presence of load disturbance," in *Proc. EPE'97*, vol. 2, 1997, pp. 386-391.

[33] C. Elmas and O.F. Bay, "Modeling and operation of a nonlinear switched reluctance motor drive based on fuzzy logic," in *Proc. EPE'95*, vol. 3, 1995, pp. 592-597.

[34] C.M. Liaw and J.B. Wang, "Design and implementation of a fuzzy controller for a high performance induction motor drive," *IEEE Trans. Syst. Man Cybern.*, vol. 21, no. 4, pp. 921-929, 1991.

[35] J.Y. Hung, W.B. Gao and J.C. Hung, "Variable structure control : A survey," *IEEE Trans. Ind. Electron.*, vol. 40, no. 1, pp. 2-22, 1993.

[36] W.B. Gao and J.C. Hung, "Variable structure control of nonlinear systems : A new approach," *IEEE Trans. Ind. Electron.*, vol. 40, no. 1, pp. 45-55, 1993.

[37] D. Driankov, H. Hellendoorn and M. Reinfrank, *An Introduction to Fuzzy Control.* Springer-Verlag, 1993.

[38] R. Palm, D. Driankov and H. Hellendoorn, *Model Based Fuzzy Control.* Springer-Verlag, 1997.

[39] R. Palm, "Robust control by fuzzy sliding mode," *Automatica.*, vol. 30, no. 9, pp. 1429-1437, 1994.

[40] L.X. Wang, *A Course in Fuzzy Systems and Control.* Prentice-Hall, 1997.

[41] J.S. Glower and J. Munighan, "Designing fuzzy controllers from a variable structure standpoint," *IEEE Trans. Fuzzy Syst.*, vol. 5, no. 1, pp. 138-144, 1997.

[42] C.S. Chen and W.L. Chen, "Robust adaptive sliding mode control using fuzzy modeling for an inverted-pendulum system," *IEEE Trans. Ind. Electron.*, vol. 45, no. 2, pp. 297-306, 1998.

[43] Y.T. Kim, M.R. Akbarzadeh-T, D.W. Lee and S.R. Lee, "Adaptive fuzzy sliding mode control of a direct drive motor," in *Proc. IEEE Int. Conf. Syst. Man Cybern.*, vol. 2, 1997, pp. 1668-1673.

[44] D.H. Kim, H.S. Kim, J.M. Kim, C.Y. Won and S.C. Kim, "Induction motor servo system using variable structure control with fuzzy sliding surface," in *Proc. Int. Conf. Ind. Electron. (IEEE-IECON'96)*, vol. 2, 1996, pp. 977-982.

[45] G.M. Asher and M. Sumner, "Parallelism and the transputer for real-time high-performance control of AC induction motors," *IEE Proc. Pt. D*, vol. 137, no. 4, pp. 179-188, 1990.

[46] G. Harp, *Transputer Applications.* Pitman, 1989.

[47] S.R. Cok, *Parallel Programs for Transputers.* Prentice-Hall, 1991.

[48] R. Blasco-Gimenez, *High Performance Sensorless Vector Control of Induction Motor Drives*. Ph.D. Thesis, University of Nottingham, October 1996.

[49] J.Cilia, *Sensorless Speed and Position Control of Induction Motor Drives*. Ph.D. Thesis, University of Nottingham, October 1997.

[50] C.S. Staines, *Sensorless Position Estimation in Asymmetric Induction Machines*. Ph.D. Thesis, University of Nottingham, November 1998.

[51] J.R. Leigh, *Applied Digital Control*. Prentice-Hall, 1985.

[52] C.L. Phillips and H.T. Nagle, *Digital Control System Analysis and Design*. Prentice-Hall, 1990.

[53] J.L. Meriam and L.G. Kraige, *Engineering Mechanics : Dynamics*. John Wiley & Sons, Inc., 1993.

[54] C.C. Lee, "Fuzzy logic in control systems : Fuzzy logic controller, part I and II," *IEEE Trans. Syst. Man Cybern.*, vol. 20, no. 2, pp. 404-435, 1990.

[55] J.M. Mendel, "Fuzzy logic systems for engineering : A tutorial," *IEEE Proc.*, vol. 83, no. 3, pp. 345-377, 1995.

[56] S. Galichet and L. Foulloy, "Fuzzy controllers : Synthesis and equivalences," *IEEE Trans. Fuzzy Syst.*, vol. 3, no. 2, pp. 140-148, 1995.

[57] K.M. Passino and S. Yurkovich, *Fuzzy Control*. Addison-Wesley, 1998.

[58] G.F. Franklin, J.D. Powell and A.E. Naeini, *Feedback Control of Dynamic Systems*. Addison-Wesley, 1994.

[59] J.J.E. Slotine, "Sliding controller design for nonlinear systems," *Int. J. Contr.*, vol. 40, no. 2, pp. 421-434, 1984.

[60] J.J.E. Slotine and W. Li, *Applied Nonlinear Control*. Prentice-Hall, 1991.

[61] F.J. Lin, S.L. Chiu and K.K. Shyu, "Novel sliding mode controller for synchronous motor drive," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 34, no. 2, pp. 532-542, 1998.

[62] H. Hashimoto, H. Yamamoto, S. Yanagisawa and F. Harasima, "Brushless servo motor control using variable structure approach," *IEEE Trans. Ind. Electron.*, vol. 34, no. 1, pp. 160-170, 1988.

[63] S.Z. Sarpturk, Y. Istefanopulos and O. Kaynak, "On the stability of discrete-time sliding mode control systems," *IEEE Trans. Automat. Contr.*, vol. 32, no. 10, pp. 930-932, 1987.

[64] Z.H. Akpolat, G.M. Asher and J.C. Clare, "Dynamic emulation of mechanical loads using a vector controlled induction motor-generator set," *IEEE Trans. Ind. Electron.*, vol. 46, no. 2, pp. 370-379, 1999.

[65] Z.H. Akpolat, G.M. Asher and J.C. Clare, "Emulation of non-linear mechanical loads," in *Proc. Int. Conf. Elec. Mach. (ICEM'98)*, vol. 3, 1998, pp. 2007-2011.

[66] Z.H. Akpolat, G.M. Asher and J.C. Clare, "Emulation of high bandwidth mechanical loads using vector controlled AC dynamometer," in *Proc. 8th Int. Power Electron. and Motion Contr. Conf.(PEMC'98)*, vol. 5, 1998, pp. 133-138.

[67] Z.H. Akpolat, G.M. Asher and J.C. Clare, "Experimental dynamometer emulation of non-linear mechanical loads", in *Proc. 33rd IEEE-IAS Conf. (IAS'98)*, vol. 1, 1998, pp. 532-539.

[68] Z.H. Akpolat, G.M. Asher and J.C. Clare, "Equivalence of fuzzy and classical controllers : An approach to fuzzy control design", to be seen in *Proc. 8th European Conference on Power Electronics and Applications (EPE'99), Lausanne, Switzerland, September 1999.*

[69] J. Ramirez, J. Bastidas and C. Vinante, "Variable structure control with integral action," *Revista Tecnica de la Facultad de Ingenieria Universidad del Zulia*, (in Spanish), vol. 20, no. 2, pp. 133-140, 1997.

[70] D.P. An, K. Nezu and T. Akuto, "Design of longitudinal control system for brushless motor electric vehicle using VSTC control theory," *JSME Int. J. Series C*, vol. 38, no. 4, pp. 756-764, 1995.

[71] M. Sumner and G.M. Asher, "Autocommissioning for voltage-referenced voltage-fed vector-controlled induction motor drives," *IEE Proc. Pt. B*, vol. 140, no. 3, pp. 187-200, 1993.