1992

An expert fuzzy logic controller employing adaptive learning for servo systems

Zong-Mu Yeh
Iowa State University

Follow this and additional works at: https://lib.dr.iastate.edu/rtd

Part of the Artificial Intelligence and Robotics Commons, and the Industrial Engineering Commons

Recommended Citation
https://lib.dr.iastate.edu/rtd/10392

This Dissertation is brought to you for free and open access by the Iowa State University Capstones, Theses and Dissertations at Iowa State University Digital Repository. It has been accepted for inclusion in Retrospective Theses and Dissertations by an authorized administrator of Iowa State University Digital Repository. For more information, please contact digirep@iastate.edu.
INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.
An expert fuzzy logic controller employing adaptive learning for servo systems

Yeh, Zong-Mu, Ph.D.

Iowa State University, 1992
An expert fuzzy logic controller employing adaptive learning
for servo systems

by

Zong-Mu Yeh

A Dissertation Submitted to the
Graduate Faculty in Partial Fulfillment of the
Requirements for the Degree of
DOCTOR OF PHILOSOPHY

Major: Industrial Education and Technology

Approved

Signature was redacted for privacy.

In Charge of Major Work

Signature was redacted for privacy.

For the Major Department

Signature was redacted for privacy.

For the Graduate College

Iowa State University
Ames, Iowa

1992

Copyright © Zong-Mu Yeh, 1992. All rights reserved.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>CHAPTER I. INTRODUCTION</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Background of the Study</td>
<td>1</td>
</tr>
<tr>
<td>Statement of the Problem</td>
<td>6</td>
</tr>
<tr>
<td>Purposes of the Study</td>
<td>7</td>
</tr>
<tr>
<td>Objectives of the Study</td>
<td>8</td>
</tr>
<tr>
<td>Assumptions of the Study</td>
<td>8</td>
</tr>
<tr>
<td>Delimitations of the Study</td>
<td>9</td>
</tr>
<tr>
<td>Procedures of the Study</td>
<td>9</td>
</tr>
<tr>
<td>Definition of Terms</td>
<td>11</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CHAPTER II. LITERATURE REVIEW</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>15</td>
</tr>
<tr>
<td>Fuzzy Sets and Fuzzy Logic</td>
<td>19</td>
</tr>
<tr>
<td>Fuzzy sets and terminology</td>
<td>20</td>
</tr>
<tr>
<td>Set-theoretic operations</td>
<td>21</td>
</tr>
<tr>
<td>Fuzzy numbers and linguistic variables</td>
<td>23</td>
</tr>
<tr>
<td>Fuzzy logic</td>
<td>26</td>
</tr>
<tr>
<td>Approximate reasoning</td>
<td>26</td>
</tr>
<tr>
<td>Fuzzy Logic Control</td>
<td>30</td>
</tr>
<tr>
<td>Fuzzification</td>
<td>31</td>
</tr>
<tr>
<td>Decision-making logic and rule base</td>
<td>35</td>
</tr>
<tr>
<td>Defuzzification</td>
<td>40</td>
</tr>
<tr>
<td>Fuzzy processing</td>
<td>42</td>
</tr>
</tbody>
</table>
Summary of the Review

CHAPTER III. METHODOLOGY

The Design Procedure
Design Factors of the Fuzzy Logic Controller
Derivation of Fuzzy Linguistic Rules
The Design of an Expert Fuzzy Logic Controller (FLC) with Adaptive Learning
   The parametric function method
   The choice of membership function
   Decision-making algorithm
   An adaptive learning algorithm for auto-tuning scaling factors
Implementation of the expert FLC

CHAPTER IV. SIMULATIONS AND RESULTS

Introduction
Performance Indices
Simulation of First-Order Plants
Simulation of Second-Order Plants with Dead-Zone and Saturation
Time-Varying Case for Second-Order Plant with Dead-Zone and Saturation
   Plant gain variation
   Plant time constant and plant gain variation
Simulation of Third-Order Plants with Dead-Zone and Saturation
Simulation of a Second-Order Oscillating Model
CHAPTER V. A NOVEL APPROACH OF MODEL-REFERENCE ADAPTIVE CONTROL

Introduction

A Novel Approach for Model-Reference Adaptive Control

Simulation and Results

Simulation result of a first-order plant with dead-zone and saturation

Simulation results of second-order plants with dead-zone and saturation

Simulation results of third-order plants with dead-zone and saturation

Summary and Discussion

CHAPTER VI. REAL-TIME APPLICATION AND RESULTS

Introduction

Real-Time Results

Summary and Discussion

CHAPTER VII. SUMMARY, CONCLUSIONS, AND RECOMMENDATIONS

Summary

Research background

Research design

Research results and finding

Conclusions

Recommendations
**LIST OF FIGURES**

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Membership function</td>
<td>2</td>
</tr>
<tr>
<td>1.2</td>
<td>A fuzzy logic control system</td>
<td>4</td>
</tr>
<tr>
<td>2.1</td>
<td>Stages of fuzzy processing</td>
<td>43</td>
</tr>
<tr>
<td>3.1</td>
<td>System step response for output C, error E and change in error CE</td>
<td>53</td>
</tr>
<tr>
<td>3.2</td>
<td>Rule justification by using system step response</td>
<td>54</td>
</tr>
<tr>
<td>3.3</td>
<td>Membership function for E, CE and U</td>
<td>57</td>
</tr>
<tr>
<td>3.4</td>
<td>Generating look-up table algorithm</td>
<td>60</td>
</tr>
<tr>
<td>3.5</td>
<td>Adaptive learning algorithm for auto-tuning scaling factors ( C_1=3, C_2=10 )</td>
<td>66</td>
</tr>
<tr>
<td>3.6</td>
<td>Implementation algorithm</td>
<td>67</td>
</tr>
<tr>
<td>3.7</td>
<td>An expert fuzzy logic controller with adaptive learning</td>
<td>68</td>
</tr>
<tr>
<td>4.1</td>
<td>Dead-zone and saturation</td>
<td>69</td>
</tr>
<tr>
<td>4.2</td>
<td>Transient response specifications</td>
<td>72</td>
</tr>
<tr>
<td>4.3</td>
<td>Comparison of the performance of the proposed controller on the plant gain=1 and the plant gain=0.8</td>
<td>76</td>
</tr>
<tr>
<td>4.4</td>
<td>Comparison of the performance of a PID controller on the plant gain=1 and the plant gain=0.8, where ( K_p=4, K_i=0.5 ) and ( K_d=0 )</td>
<td>77</td>
</tr>
<tr>
<td>4.5</td>
<td>Comparison of the performance of the expert FLC and a PID controller</td>
<td>78</td>
</tr>
<tr>
<td>4.6</td>
<td>A closed-loop control system</td>
<td>78</td>
</tr>
<tr>
<td>4.7</td>
<td>Comparison of the performance of the expert FLC and a PID controller for a second-order plant with dead-zone and saturation</td>
<td>80</td>
</tr>
</tbody>
</table>
Figure 4.8. Simulation of load change by deceasing the plant gain instantly at time=1 sec.

Figure 4.9. Simulation of load change by deceasing the plant gain instantly at time=1 sec., where Kp=38, Ki=500 and Kd=0.3

Figure 4.10. Comparison of the performance of the proposed controller and a PID controller on the plant gain change

Figure 4.11. Comparison of the performance of the expert FLC and a PID controller on time variant case

Figure 4.12. Comparison of the performance of the expert FLC and a PID controller on time-varying parameter case

Figure 4.13. Comparison of the performance of the proposed controller between the plant gain=10 and the plant gain=8

Figure 4.14. Comparison of the performance of a PID controller between the plant gain=10 and the plant gain=8

Figure 4.15. Comparison of the performance of the proposed controller and a PID controller on the plant gain changes

Figure 4.16. Simulation result of the proposed controller when the gains are set to be too high

Figure 4.17. The step response of a second-order oscillating model controlled by the expert FLC controller

Figure 4.18. The step response of a second-order damping model

Figure 5.1. A model reference control system

Figure 5.2. Adaptive learning algorithm for adapting the control action, where the constant factors can be chosen as k1=5 and k2=10

Figure 5.3. Simulation of a first-order plant with dead-zone and saturation
Figure 5.4. Simulation of a second-order plant with dead-zone and saturation 105
Figure 5.5. Simulation of a second-order plant with a time-varying plant gain 106
Figure 5.6. Simulation of a third-order plant with dead-zone and saturation 107
Figure 5.7. Simulation of a third-order plant with a time-varying plant gain 108
Figure 5.8. Simulation of a model-reference fuzzy control system 109
Figure 6.1. A closed-loop fuzzy control system 113
Figure 6.2. Hardware devices used in the experiment 114
Figure 6.3. Transient response of a servo system which is controlled by the expert FLC controller 117
Figure 6.4. Transient response of a PID control servo system with $K_p=3$, $K_i=0.1$ & $K_d=0$ 118
Figure 6.5. Transient response of a servo system with high overshoot 119
### LIST OF TABLES

| Table 2.1. | The difference between possibility and probability | 27 |
| Table 2.2. | Membership matrix table | 35 |
| Table 3.1. | Fuzzy rules | 55 |
| Table 3.2. | Membership matrix table | 58 |
| Table 3.3. | Lookup table | 61 |
| Table 4.1. | Comparison of the performance for different C, in the adaptive learning algorithm for Plant=1/(S+10) | 74 |
| Table 4.2. | Comparison of the performance of the proposed controller and the PID controller for different plants | 75 |
| Table 4.3. | Comparison of the performance between the proposed controller and the PID controller based on the same rise time, tr. | 79 |
| Table 4.4. | Comparison of the performance between the proposed controller and the PID controller for the time-varying case | 87 |
| Table 4.5. | The comparison of the performance between the proposed controller and the PID controller when the plant gain is changed instantly at the time=2.5 sec | 88 |
| Table 4.6. | Comparison of the expert FLC with another controller | 93 |
| Table 6.1. | Comparison of the performance between the proposed controller and the PID controller, where Kp=3, Ki=0.1, Kd=0 and the sample time=5 msec | 121 |
CHAPTER I. INTRODUCTION

Background of the Study

There is no doubt that society is undergoing a significant transformation from an industrial to an information society. This transition is strongly connected with the emergence and development of computer technology and the associated new fields such as artificial intelligence and systems science. The advances in computer technology have been steadily extending our capabilities for coping with systems of an increasingly broad range. While the level of system complexity which can be managed continues to increase, fundamental limits exist and it is increasingly difficult to describe any real system. Hence, one begins to consider the range of possibilities and to simplify the complexity of such systems.

One way of simplifying a very complex system is to allow some degree of uncertainty in its description. This means that statements obtained from this simplified system are less precise, but their relevance to the original system is fully maintained. Fuzziness can be used to describe event uncertainty. It measures the degree to which an event occurs, not whether it occurs. Whether an event occurs is "random." To what degree it occurs is fuzzy.
A fuzzy (vague) logic is a many-valued logic where the truth space is the set of the fuzzy numbers on the real interval \([0,1]\). The true value of a proposition is a fuzzy number whose support is included in \([0,1]\). For instance, such a fuzzy number may model linguistic temperature values whose names are "very low," "low," "medium-low," "medium," etc.

![Membership function](image)

Figure 1.1. Membership function

What differentiates fuzzy logic from traditional logic is that in fuzzy logic one can deal with fuzzy quantifiers like "very low," "low," "high" and "very high" to describe the temperature of a heater. In Figure 1.1, it is seen that the temperature 190\(^\circ\)F referred to as "medium" is true to the
degree 0.5. In traditional logic, one can only use either "low" or "high" to specify the temperature. It implies that all classes are assumed to have sharply defined boundaries in traditional logic, so either an object is a member of a class or it is not a member of a class.

Fuzzy logic can serve as a basis for reasoning with common sense knowledge. It is now finding wider and wider applications in a broad range of problem solving, from industrial process control, pattern recognition and decision analysis to weather prediction, medical diagnosis and other application areas in which the underlying information is imprecise (Zadeh, 1984).

Fuzzy logic control is one of the very important applications of fuzzy set theory (Zadeh, 1965). It is used to model human experience and knowledge. A simple fuzzy logic controller is shown in Figure 1.2. The controller basically consists of four components: fuzzifier, fuzzy inference, control rules and defuzzifier. The fuzzifier converts real-number input values into fuzzy values. The fuzzy inference is used to infer the fuzzy outputs of the controller. These fuzzy outputs are converted into real numbers by the defuzzifier. In the control rules, one of them could be "if position error is positive big or positive medium, then if change in position error is negative small, then servomotor input change is negative medium," where "positive big" and
"positive medium" are fuzzy sets on a discrete universe of position error values; similarly, "negative small" is a fuzzy set, but not on the same universe.

Figure 1.2. A fuzzy logic control system (e: error; ce: change in error; u: control action)

During the past several years, fuzzy logic control has emerged as one of the most active and fruitful areas for research in the application of fuzzy set theory (Yagishita, 1985; Yasunobu, 1987). In Japan, new products ranging from auto-focusing cameras to industrial assembly controllers are flooding the country. The pioneering research of Mamdani and
Assilian (1975) on fuzzy logic control was motivated by Zadeh’s seminal papers on the linguistic approach and system analysis based on the theory of fuzzy sets. Recent applications of fuzzy logic control in many areas have pointed a way for an effective utilization of fuzzy control in the context of complex ill-defined processes that can be controlled by a skilled human operator without knowledge of their underlying dynamics.

In conventional control system design, mathematical system modeling plays a very important role. However, the cost of obtaining a mathematical model is computationally expensive and it is also difficult to obtain the exact mathematical model of a plant. In servo systems, there are many component characteristics or variables that cannot be measured or determined accurately. Load disturbances and the change of plant parameters also cause environment variation. The classical approach is to adopt proportional integral-derivative (PID) controllers to control servo systems. These controllers will only be effective enough if the speed and accuracy requirements of the control system are not critical. The usual way to optimize the control action is to tune the PID coefficients. But this control cannot cope with varying control environments because of load disturbances and process nonlinearities.

For the above type of problems, research results have
shown that fuzzy logic controllers performed better than or at least as well as PID controllers (Mamdani, 1974; Kickert & Lemke, 1976). However, most of the experiments were concerned with slow control systems or chemical processes. For example, the industrial applications of fuzzy logic controllers in the cement industry have only considered non-learning controllers with sampling times more than one second. For fast response servo systems, whether the fuzzy logic controllers are potentially effective enough is worthy of investigation.

Statement of the Problem

Fuzzy logic control provides an algorithm that can convert human experience and knowledge into an automatic control strategy. The simple fuzzy logic controller has been successfully implemented in many test cases and actual industrial applications (Sugeno, ed., 1985). However, systematic procedures are still lacking in designing a fuzzy logic controller (Lee, 1990). The reason is that one must figure out both a large number of parameters for membership functions and control decision rules. Evaluation of these parameters for fuzzy set has been an important issue in fuzzy control. Suggestive methods have been reported in the past, but all of these approaches have drawbacks. For instance, membership functions are typically determined heuristically
and they requires many trial-and-error evaluations which are time consuming.

One key problem is that although there are many research results describing various applications of fuzzy logic controllers by using approximate reasoning, the fuzzy decision rules in such controllers were obtained either from verbal expression of domain experts or observations of human operator control action. The process of transferring expert knowledge or skilled operator control actions into a usable knowledge base is tedious and unsystematic.

Another problem is that although many fuzzy logic controllers with approximate reasoning have been developed to emulate human decision-making behavior, few focused on an important aspect of human learning which is the ability to modify fuzzy decision rules or scaling factors on the basis of experience. Thus developing a fuzzy logic controller by employing adaptive learning and adopting a more systematic way to define fuzzy rules is a challenging goal.

**Purposes of the Study**

The purpose of this study is to design and develop an expert fuzzy logic controller employing adaptive learning that can be applied to control microcomputer-based servo systems and improve the transient and steady-state responses of
control systems.

Objectives of the Study

1. Investigate and design an effective fuzzy control algorithm that can automatically control servo systems on a microcomputer.
2. Develop the necessary software package that can be used to design and simulate fuzzy control systems.
3. Simulate and evaluate the system transient responses (rise time, settling time, overshoot, steady-state error and delay time).
4. Compare the performance between the expert fuzzy logic controller and conventional PID controllers.
5. Develop a novel approach for model-reference control and evaluate this approach.
6. Evaluate the real-time application of the expert fuzzy logic controller.

Assumptions of the Study

In this research the following assumptions were made:

1. The hardware components used are stable and the values are as specified by the manufacturer.
2. Any errors introduced due to the computer are
3. The computer used to control the servo system is a true IBM compatible PC.

**Delimitations of the Study**

1. The software is designed to run only on IBM or Compatible PC.
2. The software is designed to run with the IBM interfacing board.
3. The simulation software is developed by using Turbo C++ and MATLAB software package.

**Procedures of the Study**

In this research, a model of a fuzzy logic controller employing adaptive learning is developed. An IBM compatible PC is interfaced to this servo system. The computer simulation is done by using an IBM PC/AT. The following procedures were used in conducting this study:

1. Identified the research problem.
2. Reviewed related literature devoted to fuzzy set theory and fuzzy logic control.
3. Wrote a proposal and sought committee approval of the proposal.
4. Identified and selected a microcomputer.
5. Developed some necessary hardware drivers for a servo system.
6. Developed an interface board that can connect the computer with the drivers.
7. Defined the necessary variables in the premises and in the consequences for the fuzzy decision rules, that is, determined which states of the process shall be observed and which control actions are to be considered.
8. Defined fuzzy subsets for the fuzzy variables and chosen the operating range and the discretization of the state variables.
9. Designed a Rule-Base for the fuzzy logic controller.
10. Translated fuzzy control statements into crisp control actions.
11. Made a performance analysis of the fuzzy model.
12. Modified the model.
13. Developed the necessary software to control the system.
14. Simulated the responses of the system.
15. Analyzed the results.
16. Compared the results to the results obtained by conventional approaches.
17. Wrote a final report which included summaries, conclusions and recommendations based on the findings.
Definition of Terms

The key terms of this study are defined as follows:

1. Fuzzy logic:
   A many-valued logic where the truth space is a set of the fuzzy numbers on the real interval [0,1]. It is a kind of logic using graded or qualified statements rather than ones that are strictly true or false. The results of fuzzy reasoning are not as definite as those derived by strict logic, but they cover a larger field of discourse (Zadeh, 1984).

2. Fuzzy logic controllers:
   A controller that uses production rules to capture the expert’s rule of thumb for control and allow knowledge to be expressed imprecisely.

3. Expert fuzzy logic controllers:
   Expert fuzzy controllers contain more complex knowledge about process control and use this knowledge in more complex ways (Tong, 1984).

4. Fuzzy sets:
   Sets that do not have a crisply defined membership, but rather allow objects to have grades of membership varying from 0 to 1 (Zadeh, 1984).

5. Expert system:
   An information system that can pose and answer questions
relating to information borrowed from human experts and stored in the system's knowledge base. It is a combination of knowledge base and inference engine, with supporting interface components that allows human being to interact with the system. An expert system contains knowledge in the form of facts and rules, and an inference engine for specifying how the facts and rules are to be used to reach conclusions (Edmunds, 1988).

6. Membership function:
A mapping function that maps from a fuzzy subset of X to [0,1].

7. Linguistic variables:
Ordinary-language terms that are used to represent a particular fuzzy set in a given problem such as "large", "small", "medium" or "very small."

8. Inference engine:
The inference engine is a mechanism that can manipulate the encoded knowledge from the knowledge base and to form inferences and draw conclusions. Two popular control strategies that are used to direct input and output and select which rules to evaluate are "forward chaining" and "backward chaining." In the first one, data-driven rules are evaluated for which the conditional parts are satisfied. The later one selects a special rule for evaluation. The goal is to satisfy the conditional part
9. Fuzzy reasoning:
A method of dealing with inexact or imprecise information by making it of some value in determining an outcome. Techniques of avoiding complexities when dealing with subjective information or poorly understood processes. A method of determining an adequate solution from imprecise information (Frenzel, 1987).

10. Heuristic:
Anything that helps a human or computer to discover or learn. The use of empirical knowledge to aid in problem-solving. Rules of thumb, tricks, procedural tips, and other information that help to guide, limit, and speed up the search process.

11. IF-THEN:
The form of the rules used in many artificial intelligence (AI) systems and expert systems. A conditional rule in which a certain action is taken only if some condition is satisfied. Decision-making tests that initiate an action if a specific condition is met.

12. Production rule:
An IF_THEN rule.

13. Algorithm:
A step-by-step procedure for solving a problem. A precisely defined group of rules or processes that leads
to a desired output from a given set of inputs.

14. Adaptive controllers:
An adaptive controller can change its behavior in response to changes in the dynamics of the process and the disturbances.

15. Learning control systems:
A control system is called learning if the information pertaining to the unknown features of a process or its environment is acquired by the system, and the obtained experience is used to control a process with unknown features (Fu, 1970).

16. PID controllers:
The PID controller is by far the most common control algorithm. It can be expressed as the following form:

\[ u(t) = K_p e(t) + K_i \int e(t) \, dt + K_d \frac{de(t)}{dt} , \]

where \( u \) is the control variable and \( e \) is the control error \((e=r-y)\), which is the difference between set point \( r \) and measured value \( y \). The controller parameters are proportional gain \( K_p \), integral gain \( K_i \) and derivative gain \( K_d \).

17. Plant:
A generalized complex of objects to be manipulated in a control system.
CHAPTER II. LITERATURE REVIEW

Introduction

The fuzzy set theory which was founded by Lotfi Zadeh (1965) can provide a means to handle inexact, vague data and to simplify very complex systems for the modeling and analysis of qualitative processes. The development of fuzzy set theory since its introduction in 1965 has been dramatic. More than 5,000 papers are available in the theory and applications of fuzzy sets. Since it is difficult to present a comprehensive survey of the wide variety of applications that are available, only the important literature in fuzzy logic control is subjectively reviewed.

In 1965 system scientist Lotfi Zadeh published the paper "Fuzzy sets" which formally developed multivalued logic theory, intended to generalize the classical notion of a set, introduced fuzziness into the technical literature, and inaugurated the waves of interest in fuzzy logic control from systems to commercial products. Zadeh wrote: "The notion of a fuzzy provides a convenient point of departure for the construction of a conceptual framework which parallels in many respects the framework used in the case of ordinary sets, but is more general than the latter and, potentially, may prove to have a much wider scope of applicability, particularly in the
fields of pattern classification and information processing. Essentially, such a framework provides a natural way of dealing with problems in which the source of imprecision is the absence of sharply defined criteria of class membership rather than the presence of random variables" (p. 339).

From 1960s to 1980s, the most active researchers in the development of fuzzy set theory have been Zadeh (1965, 1973), Dubois and Parde (1988), Negoita (1985), and Kaufmann and Gupta (1985). The important milestones for developing fuzzy control can be summarized as follows.

In 1972, Zadeh discussed the "over-mathematization" of control theory and gave an outline of how fuzzy sets might be used to make complex real world control problems tractable. But in the 1960s and the early 1970s, most researchers were still engaged in fundamental mathematical aspects of the theory. Until 1974, Mamdani and Assilian first demonstrated that a fuzzy logic controller which was implemented in the laboratory steam engine at Queen Mary College performed more suitably than a human controller in tasks where control rules are very difficult to formulate using classical mathematics.

In the late seventies and eighties, the theory behind much of the early work was formalized and extended. Research has focused on algorithm completeness, industrial applications, interaction, justification of fuzzy control rules, rule completeness, controller stability, fuzzy controller hardware
system and approximate reasoning.

In 1976, Braae and Rutherford discussed the use of fuzzy relations in controller analysis. They also presented a detailed analysis of various defuzzification strategies which included center of gravity method and mean of maximum method and concluded that the center of gravity method is to be preferred and the "gains" should be adjusted first to give approximately the behavior desired (Braae & Rutherford, 1978). However, the mean of maximum method strategy yields a better transient response while the "center of gravity" strategy yields a better steady-state response (Scharf & Mandic, 1985).

In 1977, Procyk introduced the self-organizing (or learning) controller. Some other researchers have also used this concept (Shao, 1988). These controllers are essentially adaptive controllers which improve the control strategy by generating or modifying control rules automatically. In order to do this, the controller must be able to assess its own performance. This assessment has usually been made by comparing the actual to the desired closed loop response. Ostergaard (1977) introduced one of the early applications of the fuzzy controller paradigm in the paper "Fuzzy logic control of a heat exchange process."

In 1980, Willaeys discussed a mathematical problem of obtaining an optimal controller design from a fuzzy
description of the system. Tong (1980) discussed the modelling problem and focussed on some practical issues of model construction and gave a detailed example of the application of the fuzzy rule-based paradigm to a complex industrial process.

In 1983, Hirox and Pedrycz designed an identification procedure to determine the parameters of a fuzzy relation or a model by using probabilistic sets. This approach could be used to deal with the identification of fuzzy systems. For developing a fuzzy controller, there are two important problems which are the defects of the reasoning algorithm and the method to acquire control rules. Takagi and Sugeno (1983) proposed a realistic fuzzy reasoning algorithm and a method to identify control rules from the human operator’s actual control actions.

In 1984, Sugeno and Nishida introduced parking control of a model car in which fuzzy control rules are derived by modelling an expert’s driving actions. They found that a method to derive fuzzy control rules from an experienced operator’s control actions is very useful to design a fuzzy controller if the operator’s knowledge and experience can be expressed in words (Sugeno & Nishida, 1985). However, if a system is too complex to be controllable by a human expert, then a fuzzy model of the system is built, and the control rules are derived theoretically. This approach requires the
development of fuzzy identification (Takagi & Sugeno, 1985) and fuzzy controller design based on a model (Sugeno & Kang, 1988).

The stability of a fuzzy control system is difficult to analyze. The main reason is that fuzzy controllers have nonlinear characteristics. Since stability is a very important concept of control systems, it is required to analyze a fuzzy control system. Kiszka, Gupta et al. (1985) proposed an energy 'measure' and designed 'an energy function' to determine the stability of a fuzzy dynamic system. A dynamic system is determined to be stable if its total energy decreases monotonously until a state of equilibrium is reached. Recent work in the stability analysis of fuzzy control systems was to adopt the fuzzy block diagram approach (Tanaka & Sugeno, 1992).

Togai and Watanabe (1986) designed a chip for real-time approximate reasoning based on the "max-min operation" of fuzzy set theory. Other research results about hardware deal with fuzzy controller hardware systems (Yamakawa, 1986 & 1987) and fuzzy memory devices (Watanabe, 1988; Yamakawa, 1987).

Fuzzy Sets and Fuzzy Logic

In this section, some of the basic concepts of fuzzy set theory and fuzzy logic which are based on Zadeh's papers

Fuzzy sets and terminology

a) A fuzzy set $B$ in a universe of discourse $U$ is a set of ordered pairs that is characterized by a membership function $\mu_B(u)$ which maps $U$ to the membership space $M$ in the real interval $[0,1]$, namely, $\mu_B : U \rightarrow [0,1]$. Where $U$ is a collection of objects denoted generically by $\{u\}$ and $B = \{(u, \mu_B(u)) | u \in U\}$. When $U$ is continuous, a fuzzy set $B$ can be expressed concisely as

$$B = \mu_B(u)/u.$$  

When $U$ is discrete, a fuzzy set $B$ is represented as

$$B = \mu_B(u_1)/u_1 + \mu_B(u_2)/u_2 + \ldots + \mu_B(u_n)/u_n.$$  

b) Two fuzzy sets $A$ and $B$ are said to be equal (denoted $A = B$) iff $\forall u \in U, \mu_A(u) = \mu_B(u)$.

c) The support of a fuzzy set $B$ is the crisp set of all $u$ in $U$ such that $\mu_B(u) > 0$.

d) The crossover points of $B$ are the elements of $x$ such that $\mu_B(x) = \frac{1}{2}$.

e) A fuzzy set $B$ is said to be normalized iff $\exists x \in U, \mu_B(x) = 1$. 
Set-theoretic operations

The basic operations of union, intersection, complement and others for fuzzy sets which are defined via their membership functions are introduced in this section. Let A and B be two fuzzy sets in U with membership functions $\mu_A$ and $\mu_B$.

a) Complement:

The membership function of the complement of a fuzzy set A is pointwise defined by

$$\overline{\mu_A}(x) = 1 - \mu_A(x), \forall x \in U.$$ 

b) Union:

The membership function $\mu_{A \cup B}$ of the union $A \cup B$ is pointwise defined by

$$\mu_{A \cup B}(x) = \max \{\mu_A(x), \mu_B(x)\}, \forall x \in U.$$ 

c) Intersection:

The membership function $\mu_{A \cap B}$ of the intersection $A \cap B$ is pointwise defined by

$$\mu_{A \cap B} = \min \{\mu_A(x), \mu_B(x)\}, \forall x \in U.$$ 

d) Power:

The membership function $\mu_p(x)$ of the ith power of a fuzzy set A is defined by

$$\mu_p(x) = [\mu_A(x)]^i, \forall x \in U.$$
e) Concentration:

The membership function $\mu_{\text{conc}(A)}(x)$ of the concentration of a fuzzy set $A$ is defined by

$$\mu_{\text{conc}(A)}(x) = [\mu_A(x)]^2, \ \forall x \in U.$$ 

f) Dilation:

The membership function $\mu_{\text{dil}(A)}(x)$ of the dilation of a fuzzy set $A$ is defined by

$$\mu_{\text{dil}(A)}(x) = [\mu_A(x)]^{\frac{1}{\gamma}}, \ \forall x \in U.$$ 

g) Sum:

The membership function $\mu_{A+B}(x)$ of the sum of two fuzzy sets $A$ and $B$ is defined by

$$\mu_{A+B}(x) = \mu_A(x) + \mu_B(x), \ \forall x \in U.$$ 

h) Algebraic product:

The membership function $\mu_{A \cdot B}(x)$ of the algebraic product of two fuzzy sets $A$ and $B$ is defined by

$$\mu_{A \cdot B}(x) = \mu_A(x) \cdot \mu_B(x), \ \forall x \in U.$$ 

i) Fuzzy Relation:

Let $U_1, \ldots, U_n$ be $n$ universes of discourse. A $n$-ary fuzzy relation is a fuzzy set in $U_1 \times \ldots \times U_n$ and is expressed as

$$R_{U_1 \times \ldots \times U_n} = \{(u_1, \ldots, u_n),$$

$$\mu_R(u_1, \ldots, u_n) | (u_1, \ldots, u_n) \in U_1 \times \ldots \times U_n \}.$$ 

For example, $n=2$, $U_1 = U_2 = \mathbb{R}^+$, $R =$"very greater than" may be defined by

$$\mu_R(u_1, u_2) = 0 \ \text{iff} \ u_1 \leq u_2,$$
\[(u_1 - u_2)/10u_2 \text{ iff } u_2 < u_1 \leq 11u_2 = 1 \text{ iff } u_1 > 11u_2.\]

j) Max-star Composition:
If R and S are fuzzy relation in \(U \times V\) and \(V \times W\), respectively, the composition of R and S is a fuzzy relation denoted by \(R \circ S\) and is defined by:
\[
R \circ S = \{(u, w), \max_v(\mu_R(u, v) \ast \mu_S(v, w))\},
\]
where * could be any operator in the class of triangular norms, namely, minimum, algebraic product, and bounded product.

k) Linguistic modifier:
A linguistic modifier is an operation that modifies the meaning of a term or a fuzzy set. If B is a fuzzy set then the modifier m generates the (composite) term \(C = m(B)\). For example,

very \(B = \text{con}(B)\)
more or less \(B = \text{dil}(B)\).

**Fuzzy numbers and linguistic variables**
Zadeh stated that "In retreating from precision in the face of overpowering complexity, it is natural to explore the use of what might be called linguistic variables, that is, variables whose values are not numbers but words or sentences in a natural or artificial languages."
The motivation for the use of the words or sentences rather than numbers is that linguistic characterizations are, in general, less specific than numerical ones" (Zadeh 1973a, p3). This quotation presents the motivation for the use of linguistic variables that can provide a basis for a systematic way for the manipulation of vague and imprecise concepts. In this section, linguistic variables are defined as follows.

a) $\alpha$-cut:

The set $B_{\alpha}$ of elements that belong to the fuzzy set $B$ at least to the degree $\alpha$ is defined by

$$B_{\alpha} = \{ u \in U | \mu_B(u) \geq \alpha \}$$

b) Convex:

A fuzzy set $B$ is convex iff its $\alpha$-cuts are convex.

It also can be defined by

$$\mu_B(\lambda x_1 + (1-\lambda)x_2) \geq \min(\mu_B(x_1), \mu_B(x_2)),$$

$$\forall x_1 \in U, \forall x_2 \in U, \forall \lambda \in [0,1].$$

c) Fuzzy Number:

A convex normalized fuzzy set defined on a real line whose membership function is piecewise continuous is called a fuzzy number.

d) Linguistic Variables:

A linguistic variable is characterized by a quintuple $(z, T(z), U, G, M)$ in which $z$ is the name of the variable; $T(z)$ indicates the term set of $z$, that is, the set of names of linguistic values of $z,
with each value being a fuzzy number and ranging over a universe of discourse \( U \) which is associated with the base variable \( u \); \( G \) is a syntactic rule for generating the name, \( Z \), of values of \( z \); and \( M \) is a semantic rule for associating with each \( Z \) its meaning, \( M(Z) \) which is a fuzzy subset of \( U \). For example, let \( Z \) be a linguistic variable with the label "temperature." Its value is also called "temperature" with \( U = [100, 350] \). Its term set may be expressed as

\[
T(\text{temperature}) = \{ \text{very low, low, medium-low, medium, high} \},
\]

where each term in \( T(\text{temperature}) \) is characterized by a fuzzy set in a universe of discourse \( U = [100,350] \) and \( G(z) \) is a rule which generates the labels of terms in the term set. The base-variable \( u \) is the degree in temperature. \( M(Z) \) is the rule that assigns a meaning, that is, a fuzzy set, to the terms. In the Figure 1.1, We might interpret "low" as the temperature close to 125°F and its fuzzy set and membership function are expressed as

\[
M(\text{low}) = \{(u, \mu_{\text{low}}(u)) \mid u \in [100,350]\}
\]

where \( \mu_{\text{low}}(u) = 0 \) if \( u < 100 \) or \( u > 150 \)

\[
= (u-100)/(125-100) \text{ if } 100 < u < 125
\]

\[
= (u-150)/(125-150) \text{ if } 125 < u < 150.
\]
Fuzzy logic

Every proposition is either true or false in classical logic (or two-valued logic) has been questioned since Aristotle. The classical two-valued logic can be extended into three-valued logic in various ways and each one is well established now. It is common in these three-valued logic to indicate the truth, falsity, and indeterminacy by 1, 0, and \( \frac{1}{2} \). Once the various three-valued logics were as meaningful and useful, it became desirable to explore generalization into n-valued logic for an arbitrary number. The true values of these n-valued logics are labeled by rational numbers in the unit interval \([0, 1]\). The set \( T_n \) of true values which are obtained by evenly dividing the interval \([0, 1]\) is defined as

\[
T_n = \{0, 1/(n-1), 2/(n-2), \ldots, (n-2)/(n-1), 1\}.
\]

A fuzzy logic (Zadeh 1973, p. 101) is a many-valued logic where the true space is the set of the fuzzy numbers on the real interval \([0, 1]\). In other words, it is an extension of set theoretic multivalued logic in which the truth values are linguistic variables. Its ultimate goal is to provide foundations for approximate reasoning with imprecise propositions using fuzzy set theory as the principal tool.

Approximate reasoning

A logic system for reasoning can be distinguished essentially by three context-independent items: truth values,
operators, and reasoning procedure (tautologies). In Fuzzy logic, truth values can be fuzzy numbers or linguistic variables and operators, like \( \land, \lor, \neg \) and \( \cdot \), and can be defined by using truth tables or possibility theory. The reasoning procedure is based on two important tautologies which are called the generalized modus tollens and the generalized modus ponens.

Note that "possibility" is different from "probability." For example, consider the statement "A construction company built \( X \) buildings in two years." \( X = \{1, 2, 3, \ldots\} \). A possibility distribution as well as a probability distribution may be associated with \( X \). The possibility distribution \( Q_X(u) \) can be interpreted as the degree of ease with which the company can complete \( u \) buildings in two years while the probability distribution \( P_X(u) \) might be determined by observing a certain number of buildings completed in two given years. The possibility will decrease as the number of buildings increases. The values of \( Q_X(u) \) and \( P_X(u) \) might be shown in Table 2.1.

<table>
<thead>
<tr>
<th>( u )</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>&gt;8</th>
</tr>
</thead>
<tbody>
<tr>
<td>( Q_X )</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.8</td>
<td>0.6</td>
<td>0.4</td>
<td>0.2</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>( P_X )</td>
<td>0.1</td>
<td>0.2</td>
<td>0.4</td>
<td>0.2</td>
<td>0.1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
"Informally, by approximate or, equivalently, fuzzy reasoning we mean the process or processes by which a possibly imprecise conclusion is deduced from a collection of imprecise premises. Such reasoning is, for the most part, qualitative rather than quantitative in nature and almost all of it falls outside of the domain of applicability of classical logic" (Zadeh, 1979, p149).

In classical logic theory, the statement of modus ponens is expressed as \((A \land (A \rightarrow B)) \rightarrow B\) or

- **Premise**: \(A\) is true,
- **Implication**: If \(A\) then \(B\),
- **Consequence**: \(B\) is true.

This could be interpreted as: "If \(A\) is true and if the statement 'If \(A\) is true then \(B\) is true' is also true then \(B\) is true." The statement of modus tollens could be denoted as \((A \rightarrow B) \land \neg B) \rightarrow \neg A\). In approximation reasoning, the above modus tollens and modus ponens are generalized to allow statements that are characterized by fuzzy sets, and to relax (slightly) the identity of the "A’s" and "B’s" in the implication and the conclusion. These fuzzy implication inference rules are named the generalized modus ponens (GMP) and the generalized modus tollens (GMT) respectively (Zadeh, 1973). For example, let \(A\), \(A'\), \(B\), \(B'\) be fuzzy statements, then the GMP could be expressed as

- **Premise**: \(x\) is \(A'\), (This apple is very red,)
- **Implication**: If \(x\) is \(A\) then \(y\) is \(B\),
  
  (If an apple is red then the apple is ripe,)
- **Consequence**: \(y\) is \(B'\). (this apple is very ripe.)
and the GMT could be expressed as

Premise: y is B′,

Implication: If x is A then y is B,

Consequence: x is A′.

The GMP is closely related to the forward data-driven inference. It is particularly useful in fuzzy logic control. The GMT is closely related to the backward data-driven inference and is particularly useful in fuzzy expert systems.

In Zadeh's compositional rule of inference (Zadeh, 1973a), the implication could be expressed by a fuzzy relation R which is written as

\[ R = A \rightarrow B = A \times B. \]

The R is a cross product of two fuzzy sets and its membership function \( \mu_R(x, y) \) could be obtained by a minimization operator,

\[ \mu_R(x, y) = \mu_A(x) \land \mu_B(y). \]

The relation R also can be denoted by a matrix in which

\[ r_{ij} = a_i \land b_j, \quad a_i \in A, \quad b_j \in B. \]

For fuzzy reasoning, the consequence B′ can be obtained by a fuzzy composition which is denoted as

\[ B' = A' \circ R, \]

and the membership function could be obtained by a maximization operator \( \lor \),

\[ \mu_{B'}(y) = \lor (\mu_{A'}(x) \land \mu_R(x, y)). \]

For example, if
A: (1, 0.9, 0.4),
B: (0.3, 1, 0.7),

then

\[
R = A \times B = \begin{bmatrix}
0.3 & 1 & 0.7 \\
0.3 & 0.9 & 0.7 \\
0.3 & 0.4 & 0.4
\end{bmatrix}
\]

and B can be obtained by

\[
B = A \circ R = \begin{bmatrix}
0.3 & 1 & 0.7 \\
0.3 & 0.9 & 0.7 \\
0.3 & 0.4 & 0.4
\end{bmatrix}
\]

In a two dimension condition, the implication can be denoted as (x is A and y is B) \(\Rightarrow\) z is C and its fuzzy relation R can be expressed as \(R = A \times B \times C\). For fuzzy reasoning, the consequence can be obtained by the fuzzy composition

\[C' = (A' \times B') \circ R.\]

**Fuzzy Logic Control**

Fuzzy logic control systems can be used to emulate human experience and decision-making behavior. Ostergaard stated that "Certain complex industrial plants, for example, a cement kiln, can be controlled with better results by an experienced operator than by conventional automatic controllers. The
control strategies employed by an operator can often be formulated as a number of rules that are simple to carry out manually but difficult to implement by using conventional algorithms. This difficulty is because human beings use qualitative rather than quantitative terms when describing various decisions to be taken as a function of different states of the process. It is this qualitative or fuzzy nature of man's way of making decisions that has encouraged control engineers to try to apply fuzzy logic to process control" (Ostergaard, 1977).

The concept of fuzzy logic control has gained wide popularity since a conceptual framework was introduced by Zadeh (1973a) and the first application of fuzzy set theory to the control of systems was introduced by Mamdani and Assilian (1975) who reported on the control of a laboratory model steam engine. Nowadays, Japan has become the most active country in this area. The basic idea of fuzzy logic control is to model human experience and human decision-making behavior. In this section, the main concepts and configuration as shown in Figure 1.2 of a fuzzy logic controller are introduced.

**Fuzzification**

Fuzzification plays an important role in dealing with uncertain information which might be objective or subjective
in nature. It could be defined as a mapping from an observed input space to fuzzy sets in a certain input universe of discourse. In a fuzzy control system, since its inputs most often are non-fuzzy values (crisp data) and its data manipulation is based on fuzzy logics, it needs fuzzification to transform data into fuzzy sets. The functions of the fuzzification process include:

a) measures the values of input variables,

b) performs a scale mapping that transfers the range of values of input variables into corresponding discretized universes of discourse. The function of the scale mapping can be either linear or nonlinear. The choice depends on some prior knowledge.

c) performs the function of fuzzification that transforms nonfuzzy input data into suitable fuzzy sets.

A fuzzification operator (fuzzifier), which takes in the real inputs and matches them to different fuzzy variables to find corresponding membership values, can be denoted as

\[ x = \text{fuzzifier}(x_0), \]

where \( x_0 \) is a nonfuzzy input value and \( x \) is a fuzzy set.

 Basically, there are two types of fuzzifier operators: continuous and discrete. The continuous type is to obtain the degree of membership in fuzzy sets by calculating the explicit equations which denote the membership functions. The shapes of membership functions are quite arbitrary and depend on the
designer’s preference. In general three kinds of shapes of membership functions are often used, but for the sake of computational efficiency and ease of data acquisition, triangular membership functions are most often used (Kaufmann & Gupta, 1985). If the observed data are disturbed by random noise, the fuzzification operator should convert the probabilistic data into fuzzy numbers (possibilistic data).

In this condition, computational efficiency is very important, so it is better to choose an isosceles triangle as the fuzzification function (Murayama & Terano, 1985). The shapes of fuzzification functions and their corresponding equations are denoted as follows:

a) Bell-shape: $\mu_a(x) = \exp[-(x-m_i)^2/b_i^2]$
where $m_i$ denotes the fuzzy means, while $b_i$ is responsible for the spread.

b) Trapezoidal:
$$
\begin{align*}
\mu_a(x) &= 0 & x < a, \\
&= (x-a)/(b-a) & a \leq x < b, \\
&= 1 & b \leq x < c, \\
&= (d-x)/(d-c) & c \leq x \leq d, \\
&= 0 & x > d.
\end{align*}
$$

c) Triangular:
$$
\begin{align*}
\mu_a(x) &= 0 & x < a \text{ or } x > c, \\
&= (x-a)/(b-a) & a \leq x \leq b, \\
&= (c-x)/(c-b) & b \leq x \leq c.
\end{align*}
$$
Since fuzzy logic controllers are implemented by using digital computer, the continuous universe of discourse needs to be converted into a discrete universe. The discretization of a universe of discourse is frequently referred to as quantization. The function of quantization is to discretize a universe into a certain number of segments. Each segment is labeled as a generic element and forms a discrete universe of discourse \( U = \{x_1, \ldots, x_n\} \). The fuzzy set then can be defined by assigning grade of membership values to each of the new discrete universe. The fuzzy set can be denoted as

\[
A = \mu_1/x_1 + \mu_2/x_2 + \ldots + \mu_n/x_n,
\]

where \( x_i \) is an element of the support of fuzzy set \( A \), \( \mu_i \) is its grade of membership in \( A \) and "+" is to denote union. For example,

\[
U = \{1, 2, 3, \ldots, 5, 8, 9, 10, 11\}.
\]

approximately \( 5 = .2/3 + .7/4 + 1/5 + .7/6 + .2/7 \).

In fuzzy control applications, the large number of quantization levels can provide an adequate approximation while the small number can save memory storage but have coarse resolution. The choice of quantization levels has an important influence on how fine a control can be obtained. A good approach for expressing the quantization and fuzzification of a discrete type is to adopt the membership matrix table. An example is expressed in Table 2.2.
Table 2.2: Membership matrix table

<table>
<thead>
<tr>
<th>Level No.</th>
<th>Range</th>
<th>NB</th>
<th>NM</th>
<th>NS</th>
<th>ZO</th>
<th>PS</th>
<th>PM</th>
<th>PB</th>
</tr>
</thead>
<tbody>
<tr>
<td>-6</td>
<td>x ≤ -4.0</td>
<td>1.0</td>
<td>0</td>
<td>0.3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-5</td>
<td>-4.0 &lt; x ≤ -2.0</td>
<td>0.7</td>
<td>0</td>
<td>0.7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-4</td>
<td>-2.0 &lt; x ≤ -1.0</td>
<td>0.3</td>
<td>1</td>
<td>0.3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-3</td>
<td>-1.0 &lt; x ≤ -0.5</td>
<td>0</td>
<td>0.7</td>
<td>0.7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-2</td>
<td>-0.5 &lt; x ≤ -0.25</td>
<td>0</td>
<td>0.3</td>
<td>1</td>
<td>0.3</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-1</td>
<td>-0.25 &lt; x ≤ -0.125</td>
<td>0</td>
<td>0</td>
<td>0.7</td>
<td>0.7</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>-0.125 &lt; x ≤ 0.125</td>
<td>0</td>
<td>0</td>
<td>0.3</td>
<td>1</td>
<td>0.3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0.125 &lt; x ≤ 0.25</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.7</td>
<td>0.7</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0.25 &lt; x ≤ 0.5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.3</td>
<td>1</td>
<td>0.3</td>
</tr>
<tr>
<td>3</td>
<td>0.5 &lt; x ≤ 1.0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.7</td>
<td>0.7</td>
</tr>
<tr>
<td>4</td>
<td>1.0 &lt; x ≤ 2.0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.3</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>2.0 &lt; x ≤ 4.0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.7</td>
</tr>
<tr>
<td>6</td>
<td>4.0 &lt; x</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.3</td>
</tr>
</tbody>
</table>

Where PB = Positive Big
PM = Positive Medium
PS = Positive Small
ZO = Zero
NS = Negative Small
NM = Negative Medium
NB = Negative Big

U = { -6, -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6}.

From the above table, if the measured input is x=-2.1 then the quantized level is -5 and the possibilistic fuzzy numbers are obtained to be {(NB, 0.7), (NM, 0.7)} by looking up the table.

**Decision-making logic and rule base**

The decision-making logic is the kernel of a fuzzy control system. It can determine the values of control variables for a given set of input fuzzy values by simulating
human decision-making based on fuzzy set concepts and inferring fuzzy control actions based on approximate reasoning. Its functions consist of fuzzy implication, fuzzy composition, interpretation of the sentence connective 'and' and 'or', and inference mechanisms.

In general, control decision rules in a fuzzy controller are fuzzy relations which are expressed as fuzzy implications: IF <condition> THEN <action>. A control decision rule is a fuzzy conditional statement in which the antecedent is a condition in its application domain and the consequence is a control action for a control system. For example, in a two-input-single-output fuzzy control system, control decision rules have the form:

\[ R_i: \text{IF } e \text{ is } E_i \text{ and } ce \text{ is } CE_i \text{ THEN } c \text{ is } C_i, \quad \text{(also)} \]
\[ R_2: \text{IF } e \text{ is } E_2 \text{ and } ce \text{ is } CE_2 \text{ THEN } c \text{ is } C_2, \quad \text{(also)} \]
\[ \ldots \]
\[ R_n: \text{IF } e \text{ is } E_n \text{ and } ce \text{ is } CE_n \text{ THEN } c \text{ is } C_n, \]

where \( e \) and \( ce \) are measured input linguistic variables, \( c \) is a linguistic control variable; \( E_i, CE_i, \) and \( C_i \) are linguistic values of the linguistic variables \( e, ce, \) and \( c \) in the universes of discourse \( U, V, \) and \( W \) respectively, with \( i = 1,2, \ldots, n; \) the 'and' operator is used to bind the fuzzy conditional statements; and the implicit operator 'also' is a union that links the rules into a rule base as
R = also( R₁, R₂, ..., Rᵢ, ..., Rₙ).

The "and" operator is usually implemented as fuzzy variable conjunction in a cartesian product space in which the variables take values in different universe of discourse. For example, the fuzzy relation (implication) of ith rule can be denoted as

\[ Rᵢ = (Eᵢ \text{ and } C.Eᵢ) \rightarrow Cᵢ = (Eᵢ \times C.Eᵢ) \times Cᵢ. \]

Its membership function can be expressed as

\[ \mu_{Rᵢ}(e, ce, c) = \mu_{EᵢCEᵢCᵢ} = \mu_{Eᵢ}(e) \land \mu_{CEᵢ}(ce) \land \mu_{Cᵢ}(c) \]

or

\[ \mu_{Rᵢ}(e, ce, c) = \mu_{EᵢCEᵢCᵢ} = \mu_{Eᵢ}(e) \cdot \mu_{CEᵢ}(ce) \cdot \mu_{Cᵢ}(c) \]

where \( \land \) is a minimization operation and \( \cdot \) is a product operation. There are also many other ways in which the sentence connective operators "also", and fuzzy implications are defined and discussed (Kiszka, 1985; Stachowicz & Kochanska, 1987), but the connective "also" as the union operator appear to be better suited for constructing fuzzy models than other methods in fuzzy control applications.

For compositional operators, there are four kinds of compositional operators which can be used in approximate reasoning such as max-min operation (Zadeh, 1973), max-product operation (Kaufmann, 1975), max-bounded-product operation (Mizumoto, 1981) and max-drastic-product operation (Mizumoto, 1981). In fuzzy control applications, the max-min and max-product operations are the most frequently used when
computational efficiency is considered. However the other two methods with different implications may get better results than max-min operator (Mizumoto & Zimmermann, 1982). For an example of max-min operation, if the input fuzzy variables are $E_0$ and $CE_0$ then an output of the inference, $C_0$, can be obtained as

$$C_0 = (E_0 \times CE_0) \circ R.$$ 

If non-fuzzy values, $e$ and $ce$, are given as inputs, then the membership function of an output is expressed as

$$\mu_{c_0}(c) = \max(\mu_{c_1}(e, ce, c), \mu_{c_2}(e, ce, c), \ldots, \mu_{c_n}(e, ce, c))$$

or

$$\mu_{c_0}(c) = \mu_{c_1}(e, ce, c) \lor \mu_{c_2}(e, ce, c) \lor \ldots \lor \mu_{c_n}(e, ce, c) = [K_1 \land \mu_{c_1}(c)] \lor [K_2 \land \mu_{c_2}(c)] \lor \ldots \lor [K_n \land \mu_{c_n}(c)],$$

where $\lor$ is a maximization operator and

$$K_i = \mu_{E_1}(e) \land \mu_{CE_1}(ce), \quad i = 1, 2, \ldots, n.$$ 

The inference mechanisms can manipulate the encoded knowledge base to form inferences and draw conclusions. The conclusions can be deduced in a number of ways which depend on the structure of the inference mechanism and the method used to present the knowledge. In fuzzy control applications, the inference mechanisms are based on one-level forward data-driven inference which is much simpler than those used in a typical expert system. The reason is that the control outputs are obtained from the union of the consequent of rules and the
consequent of a rule is not applied to the antecedent of another rule. In a typical expert system, the consequent can be applied to another rule to deduce a new consequent like a chaining or recursive reference.

In fuzzy reasoning, the consequent in a control decision rule could be a function of input linguistic variables. Takagi and Sugeno (1983, 1985) proposed a method in which the \( i \)th control rule is of the form

\[
R_i: \text{IF} \ (x_1 \text{ is } A_{i1} \text{ and } \ldots \text{ and } x_n \text{ is } A_{in}) \ \text{THEN} \ y = f_i(x_1, \ldots, x_n),
\]

where \( x_1, \ldots, x_n \) are input linguistic variables and \( y \) is control linguistic variable; \( A_{i1}, \ldots, A_{in} \) are linguistic values and \( f_i \) is a function of the input variables. This method has been applied to guide a model car smoothly along a crank-shaped track (Sugeno & Nishida, 1985) and to park a car in a garage (Sugeno & Murakami, 1985). For a simple example, only two control rules are considered,

\[
\begin{align*}
R_1: & \ \text{IF} \ x_1 \text{ is } A_{11} \text{ and } x_2 \text{ is } A_{12} \ \text{THEN} \ y = f_1(x_1, x_2) \\
R_2: & \ \text{IF} \ x_1 \text{ is } A_{21} \text{ and } x_2 \text{ is } A_{22} \ \text{THEN} \ y = f_2(x_1, x_2),
\end{align*}
\]

where the functions \( f_1 \) and \( f_2 \) may be defined as linear functions

\[
y = ax_1 + bx_2 + c.
\]

For the inputs \( x_0 \) and \( z_0 \), which are crisp values, the firing strengths \( w_1 \) and \( w_2 \) may be expressed as

\[
\begin{align*}
w_1 &= \mu_{A_{11}}(x_0) \cdot \mu_{A_{12}}(z_0), \\
w_2 &= \mu_{A_{21}}(x_0) \cdot \mu_{A_{22}}(z_0),
\end{align*}
\]
or
\[
\begin{align*}
  w_1 &= \mu_{A_{11}}(x_0) \land \mu_{A_{12}}(z_0), \\
  w_2 &= \mu_{A_{21}}(x_0) \land \mu_{A_{22}}(z_0).
\end{align*}
\]

The inferred values of the control action from the first rule and the second rule are \( w_1 f_1(x_0, z_0) \) and \( w_2 f_2(x_0, z_0) \) respectively. The crisp control action can be obtained by
\[
Y_0 = \frac{[w_1 f_1(x_0, z_0) + w_2 f_2(x_0, z_0)]}{(w_1 + w_2)}.
\]

**Defuzzification**

The output from decision-making logic is a fuzzy set of control, but a process usually requires a nonfuzzy value. In other words, the output must be transformed into a crisp value, so a defuzzification stage is needed. The defuzzification can be defined as a mapping from a space of fuzzy control actions over an output universe of discourse into a space of nonfuzzy (crisp) control actions.

The purpose of a defuzzification strategy is to generate a crisp control action that best represents the possibility distribution of an inferred fuzzy control action. Although there are several ways of tackling the defuzzification problem, there is still no systematic procedure for choosing a defuzzification. The often used strategies are the max-procedure, the center of gravity procedure and the mean of maximum procedure which are described as follows:

a) The max-procedure
This method simply chooses the control value $y_0$ for which the membership functions of control action reach the maximum which can be expressed as
\[ \max_{x \in U} \mu(y) = \mu(y_0). \]

b) The mean of maximum procedure

When there are more than one element of membership functions that possess this maximal value, it is difficult to determine $x_0$. The mean-maxima approach may be used to tackle this problem by calculating the average value of those maximum points. In case of a discrete universe, this method can be denoted as
\[ y_0 = \frac{\sum_{i=1}^{k} y_i}{k}, \]
where $y_i$ is the $i$th support value at which its membership function reaches maximum value $\mu(y_i)$, and the $k$ is the total number of such support values.

c) The center of gravity procedure

The center of gravity is the most popular method which produces the center of gravity of the possibility distribution of a control action.
\[ y_0 = \frac{\sum_{i=1}^{k} \mu_y(y_i) \cdot y_i}{\sum_{i=1}^{k} \mu_y(y_i)} \]

**Fuzzy processing**

The stages of fuzzy processing are illustrated in Figure
2.1, for a simple case of two input variables $e$ and $ce$ combined by means of only three fuzzy logic rules to form the control action $u$. Input $e$ is taken from a sensor and $ce$ is calculated from $e$. Both $e$ and $ce$ are compared to membership functions and the lower of the two conditions is selected (taking the minimum). The output of all rules is combined in a logic sum. Then, in a defuzzification process, a crisp control action is generated from the logic sum by using the center of gravity procedure.

**Summary of the Review**

In summarizing the review, some key points can be stated:

1. Fuzzy sets provide a basis for a systematic way for the manipulation of vague and imprecise concepts.

2. A fuzzy number can be described by linguistic terms ('high', 'medium', 'low', etc.) whose fuzziness provides many degrees of freedom in dealing with uncertainty by using non-uniform possibility distributions.

3. Approximate reasoning provides a way that a possibly imprecise conclusion is deduced from a collection of imprecise premises. This reasoning is qualitative rather than quantitative in nature.

4. The basic idea of fuzzy logic control is to model
INFERENDE: If (Taking the minimum) APM (Adjustment) Rule 1 1.0 zo and 0.6

Rule 2 PS PS 0.6 ce

Rule 3 PS e NS ce 0.3 PS

SENSOR INPUT INPUT

Logical sum (Taking the maximum)

PS PM 0.6 ce

DSFUZZIFICATION

center of gravity location

Figure 2.1. Stages of fuzzy processing.
The configuration of a fuzzy logic controller includes fuzzification, rule base, decision-making logic and defuzzification.

For complex industrial processes, fuzzy logic control offers an effective means of automating applications that cannot be done by using conventional control.

Recently, more and more hardware designs based on the idea of fuzzy logic have been developed and successfully applied in some industrial systems and consumer products.
CHAPTER III. METHODOLOGY

The method and procedure for designing the expert Fuzzy logic controller are described in this chapter. The algorithms for the implementation of the fuzzy logic controller are also expressed here. Computer simulations are used to compare the performance between the proposed controller and conventional PID controllers. The simulation results are shown in Chapter IV. The method and procedure of designing a novel approach for model reference control is described in Chapter V. The real-time application is discussed in Chapter VI.

The Design Procedure

The procedure for developing a fuzzy logic controller includes a number of steps. The essential design procedure can be listed as follows:

1) Determine the input (decision) variables which are observed from the states of a process.

2) Determine the control variables in which control actions are to be considered.

3) Define the fuzzy subsets for both the input (decision) and control variables.

4) Establish the operating range and the discretization
of the state variables.

5) Define the fuzzy membership functions.

6) Design the rule base in which all possible conditions in the rules could exist in the problem environment.

7) Design the computational unit, that is, supply algorithms to perform fuzzy computations.

8) Compare the set of conditions existing in the problem environment at a given time to the rules and calculate the deterministic value of the output of the fuzzy logic controller from the consequence parts of rules.

9) Modify the model for obtaining a satisfactory performance of the model. This modification may include:
   a) changing rules,
   b) adjusting the number of variables, and
   c) changing implication operators.

Design Factors of the Fuzzy Logic Controller

During the design of the fuzzy logic controller, it is necessary to consider the following factors:

1) the fuzzification strategies and the proper choice of discretization (the fuzzification operators).
2) the normalization/discretization of universe of discourse.

3) the choice of the membership function of a primary fuzzy set.

4) the derivation of fuzzy control rules.

5) the function of the mathematical definition of the fuzzy implication operator and the fuzzy connective operator.

6) the definition of the fuzzy composition operator.

7) the inference mechanism.

8) the consistency and completeness of fuzzy control rules.

9) the defuzzification strategies and the proper choice of a defuzzification operator.

The Derivation of Fuzzy Linguistic Rules

A fuzzy control system is characterized by a set of linguistic rules. How to derive fuzzy linguistic rules is one essential task in the design of a fuzzy logic controller. Basically, the fuzzy linguistic control rules could be derived by four methods. These methods are not mutually exclusive. Two of them may be combined to derive a set of effective fuzzy control rules. These methods are described as follows:
1) Based on expert's experience and knowledge
In nature, an expert may make a decision based on linguistic information rather than numerical data. Thus fuzzy linguistic rules which are in the form of "IF-THEN" rules provide a natural framework to acquire the operator knowledge. This approach could be said to be a convenient way to express an expert's domain knowledge. However, some aspects of this method to obtain fuzzy control rules are based on heuristic, trial and error (Takagi & Sugeno, 1983).

2) Based on the operator's control action
Deriving fuzzy control rules by this method is the same as making a fuzzy model of operators control. When the industrial man-machine control systems are too complex to be expressed as mathematical models, the conventional control theory cannot be applied to simulate and control the systems. However skilled operators can control such systems quite successfully without having any quantitative models. In other words, the skilled workers employ consciously or subconsciously qualitative control rules to control such systems. These qualitative control rules could be deduced from the observation of human controller's actions in terms of the input-output operation data and the rules can be expressed as a set of fuzzy control rules (Sugeno & Murakami, 1985).
3) Based on the fuzzy model of the process
For fuzzy models, the fuzzy relational equations and a state-space methodology are used to describe a fuzzy control system. This method is somewhat more complicated than other methods, but it may be used to generate a set of fuzzy control rules for attaining optimal performance of a dynamic system (Nola et al., 1991).

4) Based on learning algorithms
The learning approach is to create and modify fuzzy control rules based on emulating human decision-making behavior or human learning. The first learning fuzzy controller, called self-organizing controller (SOC), was proposed by Procyk and Mamdani (1979). The SOC has a hierarchical structure which consists of two rule bases. The first one is the general rule base of a fuzzy controller. The second one is the performance index which is used to create and modify the general rule base. Recently, further studies relating to the learning approach can be found in the literature (Shao, 1988; Wu, 1992).

The Design of an Expert Fuzzy Logic Controller (FLC) with Adaptive Learning

From the previous section, we know that designing a FLC is based on a skilled operator's or an expert's experience and
knowledge. However for a complicated process, it is very difficult to derive the control rules by synthesizing and analyzing the skilled operator's experience and there still lacks a systematic procedure to design a FLC. In this section, we proposed a more systematic way which combined the parametric function method and an adaptive learning algorithm by auto-tuning the scale factors to design an expert FLC.

The parametric function method

In the conventional decision rule set \( R \), the fuzzy control rule \( R_i \) is expressed as the fuzzy conditional statement of the form:

\[
R_i : \text{IF } (x_1 \text{ is } A_{i1} \land \ldots \land x_n \text{ is } A_{in}) \text{ then } y \text{ is } B_i.
\]

where \( x_1, \ldots, x_n \) & \( y \) are the linguistic variables which are representing the process state variables, \( A_{i1}, \ldots, A_{in} \) & \( B_i \) are the linguistic values in the universe of discourse, \( U_1, \ldots, U_n \text{ and } V \), respectively, \( i=1,2,\ldots,n \). In a more general type, the consequent is expressed as a function of the process state variables, \( x_1, \ldots, x_n \), such as

\[
R_i : \text{IF } (x_1 \text{ is } A_{i1} \land \ldots \land x_n \text{ is } A_{in}) \text{ then } y = f_i(x_1, \ldots, x_n).
\]

In these fuzzy control rules, the process states (e.g., state, state error, state integral, state error sum) at time \( t \) are evaluated and then a fuzzy control action at time \( t \) is computed as a function of \( (x_1, \ldots, x_n) \).

When the number of the process state variables is large
it is difficult to design a FLC. Raju, Zhou & Kisner (1991) proposed an approach, hierarchical fuzzy control, to reduce the complexity for designing a FLC. However, the derivation of fuzzy rules is still very complicated. In order to derive fuzzy rules in a systematic way, the following parametric function method is adopted:

For the sake of simplicity, two input fuzzy variables e (error), ce (change in error) and one control action variable u are used. Let the form of rules be "If e is PB and ce is ZO (zero) then u is PB," and let S be a linguistic value set, where S={sₙ, sₙ₋₁, . . ., s₁, s₀, s₁, . . ., sₙ}.

The fuzzy rules can be considered as a mapping

\[ \hat{\Phi}: S \times S \rightarrow S. \]

The mapping can be expressed as

\[ \hat{\Phi}(S_i, S_j) = S_{f(i,j)}, \]

where f is a function

\[ f: \{-n, \ldots, -1, 0, 1, \ldots, n\} \times \{-n, \ldots, -1, 0, 1, \ldots, n\} \rightarrow \{-n, \ldots, -1, 0, 1, \ldots, n\}. \]

The function f may be linear or nonlinear. For simplicity, the following linear functions are adopted:

\[ f(i,j) = \begin{cases} \text{Max}(-n, A*i+B*j+C), & \text{if both } i \text{ & } j \text{ are negative,} \\ \text{Min}(n, A*i+B*j+C), & \text{if both } i \text{ & } j \text{ are positive,} \\ A*i + B*j + C, & \text{otherwise,} \end{cases} \]

where A, B & C are constant parameters and can be determined
by using phase-plane trajectory or system step response. For example, assume that the fuzzy term set of input/output variables have the same cardinality, 7, with a common fuzzy term

\[ S = \{ \text{NB, NM, NS, ZO, PS, PM, PB} \}. \]

The indices of each fuzzy set in \( S \) are expressed as

\[ s_3 = \text{NB}, \ s_2 = \text{NM}, \ s_1 = \text{NS}, \ s_0 = \text{ZO}, \ s_i = \text{PS}, \ s_2 = \text{PM}, \ s_3 = \text{PB}. \]

From Figures 3.1 and 3.2, the rule justification can be done by referring the system step response and the function parameters can be determined by the following four or more rules which are obtained from the step response analysis:

\[ a^1: \text{IF } e \text{ is PB and } ce \text{ is ZO THEN } u \text{ is PB}, \]
\[ b^1: \text{IF } e \text{ is ZO and } ce \text{ is NB THEN } u \text{ is NB}, \]
\[ c^1: \text{IF } e \text{ is NB and } ce \text{ is ZO THEN } u \text{ is NB}, \]
\[ d^1: \text{IF } e \text{ is ZO and } ce \text{ is PB THEN } u \text{ is PB}. \]

According to the above rules, the parametric function can be indicated as

\[ a^1: f(3, 0) = A*3 + B*0 + C = 3A + C = 3, \]
\[ b^1: f(0, -3) = A*0 + B*(-3) + C = -3B + C = -3, \]
\[ c^1: f(-3, 0) = A*(-3)+B*0+ C = -3A+ C =-3, \]
\[ d^1: f(0, 3) = A*0 + B*3 + C = 3B+ C =3, \]

The parameters can be obtained as \( A=1, \ B=1, \) and \( C=0. \) Once the parameters \( A, \ B \) and \( C \) are determined, the complete fuzzy control rules can be determined by the parametric function \( f(i,j) \) and are expressed in Table 3.1.
Figure 3.1. System step response for output C, error E and change in error CE
Figure 3.2. Rule Justification by using system step response
Table 3.1. Fuzzy rules

<table>
<thead>
<tr>
<th>e</th>
<th>NB</th>
<th>NB</th>
<th>NB</th>
<th>NB</th>
<th>NM</th>
<th>NS</th>
<th>PS</th>
<th>PM</th>
<th>PB</th>
</tr>
</thead>
<tbody>
<tr>
<td>NB-3</td>
<td>NB</td>
<td>NB</td>
<td>NB</td>
<td>NB</td>
<td>NM</td>
<td>NS</td>
<td>ZO</td>
<td>PS</td>
<td>PM</td>
</tr>
<tr>
<td>NM-2</td>
<td>NB</td>
<td>NB</td>
<td>NB</td>
<td>NM</td>
<td>NS</td>
<td>ZO</td>
<td>PS</td>
<td>PM</td>
<td>PS</td>
</tr>
<tr>
<td>NS-1</td>
<td>NB</td>
<td>NB</td>
<td>NM</td>
<td>NS</td>
<td>ZO</td>
<td>PS</td>
<td>PM</td>
<td>PS</td>
<td>PS</td>
</tr>
<tr>
<td>ZO0</td>
<td>NB</td>
<td>NM</td>
<td>NS</td>
<td>ZO</td>
<td>PS</td>
<td>PM</td>
<td>PB</td>
<td>PB</td>
<td>PB</td>
</tr>
<tr>
<td>PS1</td>
<td>NM</td>
<td>NS</td>
<td>ZO</td>
<td>PS</td>
<td>PM</td>
<td>PB</td>
<td>PB</td>
<td>PB</td>
<td>PB</td>
</tr>
<tr>
<td>PM2</td>
<td>NS</td>
<td>ZO</td>
<td>PS</td>
<td>PM</td>
<td>PB</td>
<td>PB</td>
<td>PB</td>
<td>PB</td>
<td>PB</td>
</tr>
<tr>
<td>PB3</td>
<td>ZO</td>
<td>PS</td>
<td>PM</td>
<td>PB</td>
<td>PB</td>
<td>PB</td>
<td>PB</td>
<td>PB</td>
<td>PB</td>
</tr>
</tbody>
</table>

The choice of membership function

After formulating the fuzzy control rules, the next step is to define the membership functions of the linguistic sets, i.e., positive big (PB), positive medium (PM), etc. For each membership function the following issues must be determined:

a) whether the universe of discourse is continuous or discrete.

b) whether the shape of the membership function is triangle-shaped, trapezoid-shaped or bell-shaped.

c) whether the membership function will remain fixed or adjusted in time.

In practical applications, the universe of discourse is discretized since it is inevitable to adopt the analogue-to-digital and digital-to-analogue converters to get the data of a control system. Considering the computation and simplicity, it is helpful to choose the isosceles triangle as the fuzzification function which is used to convert a crisp value
into a fuzzy singleton within a certain universe of discourse. These membership functions of isosceles triangle are fixed during the application of the fuzzy logic controller.

Figure 3.3 shows the membership functions of the isosceles triangle and Table 3.2 is an example of the membership matrix table. It could be used for the error, change in error and control action variables. Each table consists of seven sets, including PB, PM, PS, ZO, NS, NM and NB, and each set consists of thirty-one levels, i.e., -15, -14, . . . , -1, 0, 1, . . . , 15. All the values of error, change in error, and control action variables are quantified to these thirty-one levels.

**Decision-making algorithm**

For real-time control, computational efficiency is very important. In order to shorten the running time of the FLC, a multidimensional look-up table (inference matrix) based on discrete universes, which define the output of a FLC for all possible combinations of the input signals, can be implemented by off-line processing. For approximate reasoning and defuzzification, the max-min compositional operators and the center of gravity method (COG) are adopted. The COG can yield a better steady-state performance (Scharf & Mandic, 1985). For a fuzzy control system with two inputs
Figure 3.3. Membership function for $E$, $Ce$ and $U$
Table 3.2. Membership matrix table

<table>
<thead>
<tr>
<th></th>
<th>NB</th>
<th>NM</th>
<th>NS</th>
<th>ZO</th>
<th>PS</th>
<th>PM</th>
<th>PB</th>
</tr>
</thead>
<tbody>
<tr>
<td>-15</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-14</td>
<td>0.8</td>
<td>0.2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-13</td>
<td>0.6</td>
<td>0.4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-12</td>
<td>0.4</td>
<td>0.6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-11</td>
<td>0.2</td>
<td>0.8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-10</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-9</td>
<td>0</td>
<td>0.8</td>
<td>0.2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-8</td>
<td>0</td>
<td>0.6</td>
<td>0.4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-7</td>
<td>0</td>
<td>0.4</td>
<td>0.6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-6</td>
<td>0</td>
<td>0.2</td>
<td>0.8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-5</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-4</td>
<td>0</td>
<td>0</td>
<td>0.8</td>
<td>0.2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-3</td>
<td>0</td>
<td>0</td>
<td>0.6</td>
<td>0.4</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-2</td>
<td>0</td>
<td>0</td>
<td>0.4</td>
<td>0.6</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-1</td>
<td>0</td>
<td>0</td>
<td>0.2</td>
<td>0.8</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Table 3.2. (continued)

<table>
<thead>
<tr>
<th></th>
<th>NB</th>
<th>NM</th>
<th>NS</th>
<th>NO</th>
<th>PS</th>
<th>PM</th>
<th>PB</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.8</td>
<td>0.2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.6</td>
<td>0.4</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.4</td>
<td>0.6</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.2</td>
<td>0.8</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.8</td>
<td>0.2</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.6</td>
<td>0.4</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.4</td>
<td>0.6</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.2</td>
<td>0.8</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.8</td>
<td>0.2</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.6</td>
<td>0.4</td>
</tr>
<tr>
<td>13</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.4</td>
<td>0.6</td>
</tr>
<tr>
<td>14</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.2</td>
<td>0.8</td>
</tr>
<tr>
<td>15</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.2</td>
<td>0.8</td>
</tr>
</tbody>
</table>
e, ce and one output u, the inference matrix u=Φ(e, ce) with dimension n₁ X n₂, where n₁ and n₂ are the number of quantization levels of e and ce respectively, can be determined by the following algorithm in Figure 3.4. The look-up table generated by the algorithm is shown in Table 3.3.

Input: membership matrices e, ce and u, and fuzzy rules
Output: a look-up table

procedure:

Begin
for i=1 to n₁ do
  for j=2 to n₂ do
    Begin
      h:=0; u:=0;
      for each fuzzy set k of input variable e do
        Begin
          b:=membership_matrix_e[k, i];
          if b ≠ 0 then
            for each fuzzy set g of input variable ce do
              Begin
                c:= membership_matrix_ce[g, j];
                if c ≠ 0 then
                  Begin
                    for each fuzzy set t of control action variable do
                      if membership_matrix_u[ fuzzy_rule[k, g], t] =1 then
                        h:=h+min(b, c)*t;
                        u:=u+min(b, c);
                      end
                    end
                  end
                end
              end
            end
          end
        end
      end
    end
  end
Figure 3.4 Generating look-up table algorithm
Table 3.3. Lookup table

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>-15.0</td>
<td>1.4</td>
<td>1.0</td>
<td>0.7</td>
<td>0.0</td>
<td>0.0</td>
<td>2.1</td>
<td>4.0</td>
<td>5.0</td>
<td>5.7</td>
<td>6.4</td>
<td>7.1</td>
<td>8.0</td>
<td>9.0</td>
<td>10.0</td>
<td>11.0</td>
</tr>
<tr>
<td>-1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.7</td>
<td>1.4</td>
<td>2.1</td>
<td>4.0</td>
<td>5.0</td>
<td>5.7</td>
<td>6.4</td>
<td>7.1</td>
<td>8.0</td>
<td>9.0</td>
<td>10.0</td>
<td>10.7</td>
</tr>
<tr>
<td>0.0</td>
<td>1.0</td>
<td>2.0</td>
<td>3.0</td>
<td>4.0</td>
<td>5.0</td>
<td>6.0</td>
<td>7.0</td>
<td>8.0</td>
<td>9.0</td>
<td>10.0</td>
<td>11.0</td>
<td>12.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
</tr>
<tr>
<td>-14.3</td>
<td>-14.3</td>
<td>-14.0</td>
<td>-12.1</td>
<td>-11.4</td>
<td>-10.7</td>
<td>-10.0</td>
<td>-9.0</td>
<td>-7.1</td>
<td>-6.4</td>
<td>-5.7</td>
<td>-5.0</td>
<td>-4.0</td>
<td>-2.1</td>
<td>-1.4</td>
<td>-0.7</td>
</tr>
<tr>
<td>-13.9</td>
<td>-13.6</td>
<td>-13.0</td>
<td>-11.4</td>
<td>-10.6</td>
<td>-10.0</td>
<td>-9.3</td>
<td>-8.0</td>
<td>-6.4</td>
<td>-5.6</td>
<td>-5.0</td>
<td>-4.3</td>
<td>-3.0</td>
<td>-1.4</td>
<td>-0.6</td>
<td>0.0</td>
</tr>
<tr>
<td>-13.3</td>
<td>-12.9</td>
<td>-12.0</td>
<td>-10.7</td>
<td>-10.0</td>
<td>-9.4</td>
<td>-8.6</td>
<td>-7.0</td>
<td>-5.7</td>
<td>-5.0</td>
<td>-4.4</td>
<td>-3.6</td>
<td>-2.9</td>
<td>-1.0</td>
<td>0.0</td>
<td>1.0</td>
</tr>
<tr>
<td>-12.9</td>
<td>-12.1</td>
<td>-11.0</td>
<td>-10.0</td>
<td>-9.3</td>
<td>-8.6</td>
<td>-7.9</td>
<td>-6.0</td>
<td>-5.0</td>
<td>-4.3</td>
<td>-3.6</td>
<td>-2.9</td>
<td>-2.0</td>
<td>-0.7</td>
<td>0.0</td>
<td>0.6</td>
</tr>
<tr>
<td>-12.0</td>
<td>-11.0</td>
<td>-10.0</td>
<td>-9.0</td>
<td>-8.0</td>
<td>-7.0</td>
<td>-6.0</td>
<td>-5.0</td>
<td>-4.0</td>
<td>-3.0</td>
<td>-2.0</td>
<td>-1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.7</td>
<td>1.4</td>
</tr>
<tr>
<td>-10.7</td>
<td>-10.0</td>
<td>-9.0</td>
<td>-8.0</td>
<td>-7.1</td>
<td>-6.4</td>
<td>-5.7</td>
<td>-5.0</td>
<td>-4.0</td>
<td>-3.0</td>
<td>-2.0</td>
<td>-1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.7</td>
<td>1.0</td>
</tr>
<tr>
<td>-9.8</td>
<td>-8.6</td>
<td>-7.0</td>
<td>-6.4</td>
<td>-5.7</td>
<td>-5.0</td>
<td>-4.4</td>
<td>-3.6</td>
<td>-2.9</td>
<td>-2.0</td>
<td>-0.7</td>
<td>0.0</td>
<td>0.6</td>
<td>1.4</td>
<td>3.0</td>
<td>4.3</td>
</tr>
<tr>
<td>-8.6</td>
<td>-7.9</td>
<td>-6.0</td>
<td>-5.0</td>
<td>-4.3</td>
<td>-3.6</td>
<td>-2.9</td>
<td>-2.0</td>
<td>-1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.7</td>
<td>1.4</td>
<td>2.1</td>
<td>4.0</td>
<td>5.7</td>
</tr>
<tr>
<td>-7.0</td>
<td>-6.0</td>
<td>-5.0</td>
<td>-4.0</td>
<td>-3.0</td>
<td>-2.0</td>
<td>-1.0</td>
<td>0.0</td>
<td>1.0</td>
<td>2.0</td>
<td>3.0</td>
<td>4.0</td>
<td>5.0</td>
<td>6.0</td>
<td>7.0</td>
<td>8.0</td>
</tr>
<tr>
<td>-5.7</td>
<td>-5.0</td>
<td>-4.0</td>
<td>-2.1</td>
<td>-1.4</td>
<td>-0.7</td>
<td>0.0</td>
<td>1.0</td>
<td>2.0</td>
<td>3.0</td>
<td>4.0</td>
<td>5.0</td>
<td>6.0</td>
<td>7.0</td>
<td>8.0</td>
<td>9.0</td>
</tr>
<tr>
<td>-4.4</td>
<td>-3.6</td>
<td>-2.0</td>
<td>-0.7</td>
<td>0.0</td>
<td>0.6</td>
<td>1.4</td>
<td>3.0</td>
<td>4.3</td>
<td>5.0</td>
<td>6.0</td>
<td>7.0</td>
<td>8.0</td>
<td>9.0</td>
<td>10.0</td>
<td>11.0</td>
</tr>
<tr>
<td>-3.6</td>
<td>-2.9</td>
<td>-1.0</td>
<td>0.0</td>
<td>0.7</td>
<td>1.4</td>
<td>2.1</td>
<td>4.0</td>
<td>5.0</td>
<td>6.0</td>
<td>7.0</td>
<td>8.0</td>
<td>9.0</td>
<td>10.0</td>
<td>11.0</td>
<td>12.0</td>
</tr>
<tr>
<td>-2.0</td>
<td>-1.1</td>
<td>0.0</td>
<td>1.0</td>
<td>2.0</td>
<td>3.0</td>
<td>4.0</td>
<td>5.0</td>
<td>6.0</td>
<td>7.0</td>
<td>8.0</td>
<td>9.0</td>
<td>10.0</td>
<td>11.0</td>
<td>12.0</td>
<td>13.0</td>
</tr>
<tr>
<td>-0.7</td>
<td>0.0</td>
<td>1.0</td>
<td>2.9</td>
<td>3.6</td>
<td>4.3</td>
<td>5.0</td>
<td>6.0</td>
<td>7.0</td>
<td>8.0</td>
<td>9.0</td>
<td>10.0</td>
<td>11.0</td>
<td>12.0</td>
<td>13.0</td>
<td>14.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.7</td>
<td>2.0</td>
<td>3.6</td>
<td>4.4</td>
<td>5.0</td>
<td>5.7</td>
<td>6.0</td>
<td>6.7</td>
<td>7.0</td>
<td>7.0</td>
<td>7.0</td>
<td>7.0</td>
<td>7.0</td>
<td>7.0</td>
<td>7.0</td>
</tr>
<tr>
<td>0.6</td>
<td>1.4</td>
<td>3.0</td>
<td>4.3</td>
<td>5.0</td>
<td>5.6</td>
<td>6.4</td>
<td>6.0</td>
<td>6.0</td>
<td>5.0</td>
<td>5.0</td>
<td>5.0</td>
<td>5.0</td>
<td>5.0</td>
<td>5.0</td>
<td>5.0</td>
</tr>
<tr>
<td>1.4</td>
<td>2.1</td>
<td>4.0</td>
<td>5.0</td>
<td>5.7</td>
<td>6.4</td>
<td>7.1</td>
<td>7.0</td>
<td>7.0</td>
<td>6.0</td>
<td>6.0</td>
<td>6.0</td>
<td>6.0</td>
<td>6.0</td>
<td>6.0</td>
<td>6.0</td>
</tr>
<tr>
<td>3.0</td>
<td>4.0</td>
<td>5.0</td>
<td>6.0</td>
<td>7.0</td>
<td>8.0</td>
<td>9.0</td>
<td>10.0</td>
<td>11.0</td>
<td>12.0</td>
<td>13.0</td>
<td>14.0</td>
<td>15.0</td>
<td>16.0</td>
<td>17.0</td>
<td>18.0</td>
</tr>
<tr>
<td>4.3</td>
<td>5.0</td>
<td>6.0</td>
<td>7.9</td>
<td>8.6</td>
<td>9.3</td>
<td>10.0</td>
<td>11.0</td>
<td>12.0</td>
<td>13.0</td>
<td>14.0</td>
<td>15.0</td>
<td>16.0</td>
<td>17.0</td>
<td>18.0</td>
<td>19.0</td>
</tr>
<tr>
<td>5.0</td>
<td>5.7</td>
<td>7.0</td>
<td>8.6</td>
<td>9.4</td>
<td>10.0</td>
<td>10.7</td>
<td>11.0</td>
<td>12.0</td>
<td>13.0</td>
<td>14.0</td>
<td>15.0</td>
<td>16.0</td>
<td>17.0</td>
<td>18.0</td>
<td>19.0</td>
</tr>
<tr>
<td>5.6</td>
<td>6.4</td>
<td>8.0</td>
<td>9.3</td>
<td>10.0</td>
<td>10.6</td>
<td>11.4</td>
<td>12.0</td>
<td>12.9</td>
<td>13.9</td>
<td>14.3</td>
<td>15.0</td>
<td>16.0</td>
<td>17.0</td>
<td>18.0</td>
<td>19.0</td>
</tr>
<tr>
<td>6.4</td>
<td>7.1</td>
<td>9.0</td>
<td>10.0</td>
<td>10.7</td>
<td>11.4</td>
<td>12.1</td>
<td>14.0</td>
<td>14.3</td>
<td>14.3</td>
<td>14.3</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
</tr>
<tr>
<td>8.0</td>
<td>9.0</td>
<td>10.0</td>
<td>11.0</td>
<td>12.0</td>
<td>13.0</td>
<td>14.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
</tr>
<tr>
<td>9.3</td>
<td>10.0</td>
<td>11.0</td>
<td>12.1</td>
<td>12.9</td>
<td>13.6</td>
<td>14.3</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
</tr>
<tr>
<td>10.7</td>
<td>10.7</td>
<td>12.0</td>
<td>12.9</td>
<td>13.6</td>
<td>14.3</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
</tr>
<tr>
<td>10.6</td>
<td>11.4</td>
<td>13.0</td>
<td>13.6</td>
<td>14.3</td>
<td>14.3</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
</tr>
<tr>
<td>11.4</td>
<td>12.1</td>
<td>14.0</td>
<td>14.3</td>
<td>14.3</td>
<td>14.3</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
</tr>
<tr>
<td>13.0</td>
<td>14.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
<td>15.0</td>
</tr>
</tbody>
</table>
An adaptive learning algorithm for auto-tuning scaling factors

When a process exhibits time variant dynamic behavior, the controller should be retuned to adapt the variant behavior and improve the closed-loop performance. The main objectives of adaptive control are to reduce the instability and oscillation, and to speed up the system response. These objectives could be achieved by providing maximum drive at larger errors, changing the control action when the predicted output is far away from the setpoint, and providing gentler action when the system is stable and nearer to the setpoint.

There are several methods for adapting fuzzy logic controller such as:

a) modifying fuzzy rules on-line.
b) modifying membership functions of the fuzzy variable on-line.
c) modifying the scaling factors.

The disadvantages of the first method is the convergence time of the control action is tedious since only the fired rule is modified each time and the convergence of the control action is not guaranteed. For the second method, when the number of fuzzy sets is large, there will be so many parameters needed to be adjusted that it is not efficient in real-time control. The third one is discussed in this section. The regulation of the scaling factors on-line is equivalent to the modification of fuzzy rules on-line in some
way. However, it is more efficient and more independent of the process than other methods. Its running time is also shorter. The reasons are that the algorithm for auto-tuning the scaling factors is easy to implement even on a small microcomputer system and the resulting algorithm is simple and easy to understand.

Procyk and Mamdani (1979) mentioned the effect of scaling factors. When the input scaling factors GE and GCE are increased, the output response is more sensitive around the set-point and less sensitive during rise time, and vice versa. For the output scaling factor GU, a low value results in a slow rise time, a larger integral square error, but has fast convergence. A high value may cause instability.

In the proposed adaptive learning algorithm, the values of GE and GCE were adapted to achieve a compromise between error sensitivity and controller instability. When the system response is steady but the error is still large, the values of GE and GCE are increased. If the system response is unstable, or the overshoot is too large, then the values of GE and GCE are decreased quickly.

The control action $u$ is defined as

$$u = u_t + u_c,$$

where $u_t$ is obtained from the look-up table and $u_c$ is
determined by using a credit assignment law (reinforcement learning) in which a scalar error value \( e \) represents the performance of the controller. The goal of the learning law is to force the error asymptotically to zero as soon as possible by assigning positive credit (reward) to desirable action and negative credit (punishment) to undesirable action. The error \( e \) is defined as
\[
e = r - c,
\]
where \( r \) is the desired value and \( c \) is the output of a plant. The \( u_c \) is defined so as to provide maximum drive (short rise time) at larger error and provide small control action offset when the system response is nearer to the set point. The adaptive learning algorithm for adjusting the scaling factors is indicated in Figure 3.5.

**Implementation of the expert FLC**

The implementation of the expert FLC controller can be performed by approximating the sampled values of the controller inputs to the nearest discretization points and then they are scaled by the scaling factors which are adjusted by the adaptive learning algorithm to generate the indices of look-up table \( \Phi(.) \). The value \( u_c \), taken from the look-up table adds the offset \( u_c \) to obtain the control action \( u \) which is expressed as
Input: error (=r-c), CE (fuzzy variable), GE & GCE.
Output: u_c, GE, and GCE.

Procedure.

Begin
if the change in error is zero but the steady-state error is larger than the specification e_{ss}, then

Begin
GE := GE + (1+error)^2;
GCE := GCE + 10*exp(1+abs(error));
end
else if the system response is detected to be unstable or to have too large overshoot, then

begin
GE := GE/2;
GCE := GCE/2;
end;

if GE < initial_value then GE := initial_value;
if GCE < initial_value then GCE := initial_value;

{adjusting the control action u of the expert FLC}
threshold := kT* sqrt(abs(r))*e_{ss}; {r=desired value}
{the unit of sample time kT is msec}
{rewards}
if error > threshold then u_c := 1 + C_1*GE*abs(r)
else if (error > threshold/2) and (error < threshold)
then u_c := GE*(1+error)/C_2
else if (error > 0) then u_c := 1 + error;

{punishment}
if error < -threshold then u_c := -(1 + C_1*GE*abs(r))
else if (error > -threshold) and (error < -threshold/2)
then u_c := -(GE*(1+error)/C_2)
else if (error > -threshold/2) and (error < -0)
then u_c := -(1+error);

if the fuzzy states of error and change in error are zero then u_c := 0;
end;

Figure 3.5 adaptive learning algorithm for auto-tuning scaling factors (C_1=3, C_2=10)
the control action \( u \) can be used to drive the plant (process). The basic implementation algorithm is expressed in Figure 3.6. The whole structure is shown in Figure 3.7.

Procedure:

Begin
\begin{verbatim}
initialize the sampling time \( kT \) and parameters;
load look-up table \( \Phi(.) \);
while (true) do 
begin
for each sampling time \( kT \) do 
begin
sample \( e(kT) \) and \( ce(kT) \);
\{ \( e(kT) := \text{reference input} - \text{output} \) \}
\{ \( ce(kT) := e(kT) - e((k-1)T) \) \}
approximate \( e(kT) \) and \( ce(kT) \) to discretization points;
tune the scaling factor \( GE, GCE \) and offset \( u_c \);
adjust the \( GE*e(kt) \) and \( GCE*ce(kt) \) into table indices \( i \) and \( j \);
\( u := \Phi(i,j) + u_c \);
apply \( u \) to the plant;
end
end;
\end{verbatim}

Figure 3.6 implementation algorithm
Figure 3.7. An expert FLC with adaptive learning (GE, GCE: scaling factor; q1, q2: quantization)
CHAPTER IV. SIMULATIONS AND RESULTS

Introduction

In this chapter, the comparison between the proposed controller and conventional PID controllers is discussed. The expert FLC with adaptive learning was applied to control some plants whose models were viewed as black boxes. This means that their mathematical models are assumed to be unknown. For different control models, the same expert FLC to control the control systems was used. In the conventional approach, one has to tune the parameters of the PID controller for different control models. In the simulation, the nonlinear component of dead zone and saturation is also included in the control system which is expressed in Figure 4.1.

![Figure 4.1. Dead-zone and saturation](image_url)
For performance comparison, the performance criteria are readily defined whatever the type of controllers. For example, the control system should respond in minimum time, maximum allowed stress levels should not be exceeded during transients, and overshoots should be restricted to a certain percent of the desired level. Since there is no direct way to utilize performance criteria in the design of rule-based systems, the performance of the proposed controller is compared with that of a PID controller based on step inputs.

Performance Indices

Typical performance indices that are used to characterize the transient response to a unit step input include maximum overshoot, delay time, rise time, settling time, integral absolute error, etc. Figure 4.2 illustrates a typical unit step response of a control system. The above-mentioned indices are defined with respect to a step input as follows:

1) Rise time, $tr$:
   \[ tr = t_2 - t_1, \]
   where $C(t_2) = 90\%$ of the final value of $C(t)$;
   $C(t_1) = 10\%$ of the final value of $C(t)$;

2) Maximum overshoot, $ov$:
   \[ ov = \text{the largest value of } C(t) - \text{the desired value}, \]
71

\[ ov(\%) = \frac{\text{maximum overshoot}}{\text{desired value}} \times 100\% \]

3) Delay time, \( t_d \):
\[ t_d = t_0, \]
where \( C(t_0) \) first reaches the 50\% of the final value of \( C(t) \).

4) Settling time, \( t_s \):
The \( t_s \) is defined as the time required for the transient response to decrease and stay within a specified percentage (3\% or 5\%).

5) Integral of absolute error, IAE:
\[ IAE = \sum_{t=0}^{KT} |e(t)|, \]
where \( e(t) = \text{input} - C(t) \) and \( T \) is sampling time.

6) Average error, AE:
\[ AE = \frac{\sum_{t=0}^{KT} e(t)}{K}. \]

7) Average absolute error, AAE:
\[ AAE = \frac{IAE}{K}; \]
The AE and AAE may be used to detect whether the system response is in limit cycle when the ratio between AE and AAE is large.
Figure 4.2. Transient response specifications
8) Steady-state error, \( e_{ss} \):
\[ e_{ss} = \text{final value of } C(t) - \text{desired value.} \]

**Simulation of First-Order Plants**

A lot of plants in industrial application are simply expressed as first-order transfer functions. In this section, first-order plants \( G(s) \) are used to demonstrate that the performance of the expert FLC with adaptive learning is very good.

\[
G(s) = \frac{C}{AS+B},
\]

where \( C=1 \), \( A=1 \) and \( B=10 \) for the first test case, and \( C=0.8 \), \( A=1 \) and \( B=1 \) for the second test case. One of the advantages of a fuzzy logic controller is that it is used to control a process without any information about its mathematical model. Thus all plants used in the simulation are viewed as black boxes.

The aim of the exercise is to demonstrate the following:

a) The expert FLC tunes itself to reach an optimum state by auto-tuning the scaling factors.

b) The rise time and settling time are easily controlled by adjusting the constant \( C_1 \) in the adaptive learning algorithm. Table 4.1 illustrates the comparison of performance for different \( C_1 \).
Table 4.1. Comparison of the performance for different $C_1$ in the adaptive learning algorithm for Plant=$1/(S+10)$

<table>
<thead>
<tr>
<th>$C_1$</th>
<th>$tr$</th>
<th>$ts$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>41.4 m sec</td>
<td>55.4 m sec</td>
</tr>
<tr>
<td>3</td>
<td>40.5 m sec</td>
<td>32 m sec</td>
</tr>
<tr>
<td>5</td>
<td>15 m sec</td>
<td>18.4 m sec</td>
</tr>
<tr>
<td>7</td>
<td>12 m sec</td>
<td>13.4 m sec</td>
</tr>
<tr>
<td>9</td>
<td>13 m sec</td>
<td>14.2 m sec</td>
</tr>
</tbody>
</table>

Figure 4.3 demonstrates the comparison of the performance of the expert FLC controller when it is used to control the plant $1/(S+10)$ and the plant $0.8/(S+1)$. In the simulation, the plant $1/(S+10)$ was used first, then the plant was changed instantly to $0.8/(S+1)$ at time=0.2 sec. From the results in Figure 4.3, it was concluded that there is no significant difference in the performance between different plants controlled by the same expert FLC controller.

Figure 4.4 demonstrates the comparison of the performance of the PID controller when it is used to control the different plants. The results illustrate that there exist significant differences in the performance. Figure 4.5 illustrates the comparison of the performance between the proposed controller and the PID controller based on the same rise time $tr$ when they are used to control different processes. From Figure 4.5 and Table 4.2, results indicate that the performance of the proposed controller is better than the PID controller since the proposed controller has a shorter settling time and a
Table 4.2. Comparison of the performance of the proposed controller and the PID controller for different plants

<table>
<thead>
<tr>
<th>controller</th>
<th>proposed FLC</th>
<th>PID</th>
<th>proposed FLC</th>
<th>PID</th>
</tr>
</thead>
<tbody>
<tr>
<td>plant</td>
<td>1/(S+10)</td>
<td>0.8/(S+1)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ts, msec.</td>
<td>40.5</td>
<td>137</td>
<td>43.6</td>
<td>&gt;150</td>
</tr>
<tr>
<td>ov</td>
<td>0.1%</td>
<td>12.5%</td>
<td>0.06%</td>
<td>36.7%</td>
</tr>
<tr>
<td>td, msec.</td>
<td>20.8</td>
<td>16.6</td>
<td>24.4</td>
<td>15</td>
</tr>
<tr>
<td>tr, msec.</td>
<td>32</td>
<td>36</td>
<td>34</td>
<td>28</td>
</tr>
</tbody>
</table>

smaller overshoot.

Simulation of Second-Order Plants with Dead-Zone and Saturation

In this section, control systems with second-order plants are controlled by the proposed controller and PID controller respectively. The system configuration is shown in Figure 4.6. The aim of this exercise is to demonstrate that the performance of the proposed controller is better than or as good as the PID controller based on the same rise time (tr). Figure 4.7 illustrates that the integral of absolute error and the settling time of the proposed controller are better than those of the PID Controller, where Kp=38, Ki=500, and Kd=0.3. For simulating the load changes, Figures 4.8, 4.9, 4.10 and Table 4.3 illustrate the comparison of the performance between
the proposed controller and the PID controller when the plant gain is instantly decreased 20% at time=1 sec. The simulation results show that there is no significant difference for the different plant gains. However, the proposed controller has a shorter settling time.

Figure 4.3. Comparison of the performance of the proposed controller on the plant gain=1 and the plant gain=0.8 which is decreased instantly at time=0.2 sec.
Figure 4.4. Comparison of the performance of the PID controller on the plant gain=1 and the plant gain=0.8 which is decreased instantly at time=0.2 sec, where $K_p=4$, $K_i=0.5$ and $K_d=0$. 
Figure 4.5. Comparison of the performance between the expert FLC and a PID controller

Figure 4.6. A closed-loop control system
Time-Varying Case for Second-Order Plant with Dead-Zone and Saturation

The transient responses of the control system with time-varying parameters are discussed in this section. For second-order plants, there are two different experiments set up:

1) plant gain variation and
2) plant time constant and plant gain variation.

Table 4.3. Comparison of the performance between the proposed controller and the PID controller based on the same rise time, tr. (sample time=1 msec)

<table>
<thead>
<tr>
<th>controller</th>
<th>expert FLC</th>
<th>PID</th>
<th>expert FLC</th>
<th>PID</th>
</tr>
</thead>
<tbody>
<tr>
<td>plant</td>
<td>151.5/(s+73)</td>
<td>120/(s+73)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ts, msec.</td>
<td>92</td>
<td>157</td>
<td>101</td>
<td>169</td>
</tr>
<tr>
<td>ov</td>
<td>16.5%</td>
<td>24.9%</td>
<td>14.2%</td>
<td>24.3%</td>
</tr>
<tr>
<td>td, msec.</td>
<td>26</td>
<td>19</td>
<td>28</td>
<td>21</td>
</tr>
<tr>
<td>tr, msec.</td>
<td>25</td>
<td>25</td>
<td>30</td>
<td>28</td>
</tr>
<tr>
<td>IAE</td>
<td>32.91</td>
<td>39.64</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Plant gain variation

Figure 4.11 and Table 4.4 illustrate the comparison of the performance between the proposed controller and the PID controller which are used to control the system with time-varying parameters. The plant gain was increased linearly to 162% at time=0.5 second, then the set point was changed to 2 at time=0.5 second and the plant gain was increased linearly to 225% at time=1 second. The target value was recovered for the gain variation and the set point change. The simulation
Figure 4.7. Comparison of the performance between the proposed controller and a PID controller for a second-order plant with dead-zone and saturation.
Figure 4.8. Simulation of load change by decreasing the plant gain instantly at time=1 sec.
results indicate that no significant effect is seen for the gain variation, but the performance of the proposed controller is better than that of the PID controller. When the set point was changed, the transient response exhibited a larger maximum overshoot because of the higher plant gain.

Plant time constant and plant gain variation

Figure 4.12 illustrates the comparison of performance between the proposed controller and the PID controller when the plant time constant and gain are varying with time. In the simulation, the time constant was changed from increasing 440% at the beginning to increasing 26% at time=1 second and at the same time the plant gain was increased by 225% at time=1 second. Table 4.4 demonstrates the simulation results of the proposed controller and the PID controller from time=0 to time=0.5 second. For the proposed controller, there is no significant effect on the transient response of the closed-loop system despite the parameter changes. However, the PID controller is sensitive to the parameters of plants changes.

Simulation of Third-Order Plants with Dead-Zone and Saturation

In this section, the simulation of a load change by decreasing the plant gain of a third-order plant G(S) is presented. The plant is expressed as
controlled by PID controller

the plant=\frac{151.5}{s(s+73)} \quad \text{the plant}=\frac{120}{s(s+73)}

t_s=157 \text{ msec} \quad t_s=169 \text{ msec}
\quad o_c=24.9 \text{ percent} \quad o_v=24.3 \text{ percent}
\quad t_d=19 \text{ msec} \quad t_d=21 \text{ msec}
\quad t_r=25 \text{ msec} \quad t_r=28 \text{ msec}

gure 4.9. Simulation of load change by decreasing the plant gain instantly at time=1 sec.
Figure 4.10. Comparison of the performance between the proposed controller and a PID controller on the plant gain change.
Figure 4.11. Comparison of the performance between the proposed controller and a PID controller on the time variant case. The plant = 151(1+N/200)/(S(S+73)), where N = time/sampling-interval.
Figure 4.12. Comparison of the performance between the proposed controller and a PID controller on the time-varying parameter case. The plant= $151(1+N/200)/(S(S+73*440/(N+100)))$, where $N=\text{time/sampling interval}$.
The aim of this exercise is to illustrate that the performance of the proposed controller is better than the conventional PID controller when the plant gain is changed instantly. This comparison is based on the same rise time.

The performance of the expert FLC for different plant gains is indicated in Figure 4.13. The simulation results indicate that there is no significant effect on the transient response of the system in which the plant gain is changed at time=2.5 sec. since its settling time increased only 8%. For the same condition, the performance of the PID controller is illustrated in Figure 4.14 and the result shows that the settling time is greater when the plant gain is decreased instantly by 20% at time=2.5 sec. This means that the
performance of PID controller is sensitive to parameter changes because its settling time increases 50% when the plant gain is decreased by 20% instantly at the time=2.5 sec. The comparison of the performance between the proposed controller and the PID controller is shown in Figure 4.15 and Table 4.5.

When the scaling factors of the expert FLC are tuned to be too high at the beginning, the system response may have too large an overshoot or the system may be unstable. The proposed controller will adjust the gain to keep the system to be stable. Figure 4.16 indicates that the system response tends to be unstable at the early stage and then it is adjusted to be stable later.

Table 4.5. The comparison of the performance between the proposed controller and the PID controller when the plant gain is changed instantly at the time=2.5 sec. The sample time=20 msec and the comparison is based on the similar rise time.

<table>
<thead>
<tr>
<th>controller</th>
<th>expert FLC</th>
<th>PID</th>
<th>expert FLC</th>
<th>PID</th>
</tr>
</thead>
<tbody>
<tr>
<td>plant</td>
<td>10/(s(s+5)(s+10))</td>
<td>8/(s(s+5)(s+10))</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ts, sec.</td>
<td>0.59</td>
<td>1.04</td>
<td>0.66</td>
<td>1.58</td>
</tr>
<tr>
<td>ov</td>
<td>4.4%</td>
<td>9.7%</td>
<td>1%</td>
<td>11%</td>
</tr>
<tr>
<td>td</td>
<td>0.38</td>
<td>0.37</td>
<td>0.41</td>
<td>0.41</td>
</tr>
<tr>
<td>tr</td>
<td>0.38</td>
<td>0.36</td>
<td>0.41</td>
<td>0.41</td>
</tr>
<tr>
<td>IAE</td>
<td>19.5</td>
<td>20.5</td>
<td>20.5</td>
<td>27.4</td>
</tr>
</tbody>
</table>
Figure 4.13. Comparison of the performance of the proposed controller between the plant gain=10 and the gain=8 which is decreased instantly at time=2.5 sec.
the plant gain changed instantly at time=2.5 sec

controlled by PID

\[ \text{plant}=10/s(s+5)(s+10) \quad \text{and} \quad \text{plant}=8/s(s+5)(s+10) \]

\[ \begin{align*}
  \text{ts} &= 1.04 \text{ sec} \\
  \text{ov} &= 9.7 \text{ percent} \\
  \text{td} &= 0.37 \text{ sec} \\
  \text{tr} &= 0.36 \text{ sec} \\
  \end{align*} \]

\[ \begin{align*}
  \text{ts} &= 1.58 \text{ sec} \\
  \text{ov} &= 11 \text{ percent} \\
  \text{td} &= 0.41 \text{ sec} \\
  \text{tr} &= 0.41 \text{ sec} \\
  \end{align*} \]

Figure 4.14. Comparison of the performance of a PID controller between the plant gain=10 and the gain=8 which is decreased instantly at time=2.5 sec, where \(K_p=40\), \(K_i=2.5\), and \(K_d=6\).
the plant gain changed instantly at time=2.5 sec

Figure 4.15. Comparison of the performance between the proposed controller and a PID controller on the plant gain changes.
Figure 4.16. Simulation results of the proposed controller when the gains are set to be too high.
Simulation of a Second-Order Oscillating Model

In this section, the performance of the proposed controller which was used to control a second-order oscillating plant is presented. The plant transfer function $G(S)$ is expressed as follows:

$$G(S) = \frac{1}{S^2+1}.$$

The sample interval in this simulation for the oscillating plant is 50 msec. Figure 4.17 and Figure 4.18 show the step responses of the fuzzy control system. The comparison of the performance between the proposed controller and Wu's controller (Wu, Z. Q. et al., 1992) is shown in Table 4.6. From the comparison, the proposed controller has better performance since the settling time, delay time and rise time are shorter.

Table 4.6. Comparison of the expert FLC with another controller

<table>
<thead>
<tr>
<th>controller</th>
<th>proposed FLC</th>
<th>Wu's controller</th>
</tr>
</thead>
<tbody>
<tr>
<td>plant 1/(S^2+1)</td>
<td>2/(S^2+3S+2)</td>
<td>1/(S^2+1) 2/(S^2+3S+2)</td>
</tr>
<tr>
<td>ts, sec.</td>
<td>2</td>
<td>0.55</td>
</tr>
<tr>
<td>ov</td>
<td>16%</td>
<td>0%</td>
</tr>
<tr>
<td>td, sec.</td>
<td>0.36</td>
<td>0.3</td>
</tr>
<tr>
<td>tr, sec.</td>
<td>0.66</td>
<td>0.34</td>
</tr>
</tbody>
</table>
Figure 4.17. the step response of a second-order oscillating model controlled by the proposed Controller
A second-order damping model

\[ G_1(s) = \frac{1}{(0.5s+1)(s+1)} \]

- \( ts = 0.55 \text{ sec.} \)
- \( ov = 0 \)
- \( td = 0.3 \text{ sec.} \)
- \( tr = 0.34 \text{ sec.} \)

Figure 4.18. The step response of the fuzzy control system for the model \( \frac{1}{(0.5s+1)(s+1)} \)
Summary

In this chapter, the comparison of the performance between the proposed controller and the conventional PID controllers which are used to control different plants with dead-zone and saturation are presented. For first-order, second-order and third-order plants, the performance of the proposed controller is better than that of the PID controllers. The reason is that the performance of PID controllers becomes worse and more sensitive to the process model change or parameters varying with time. In the time-varying case, the simulation results showed that the control systems controlled by the proposed controller had a shorter settling time and a smaller maximum overshoot. Finally, the performance of our controller was compared with Wu's controller. Simulation results demonstrated that the proposed controller had better performance in settling time, delay time and rise time.
CHAPTER V. A NOVEL APPROACH OF MODEL-REFERENCE ADAPTIVE CONTROL

Introduction

In this chapter, a novel approach employing adaptive learning for model-reference control is proposed. The objective of the proposed controller is to force the error between the plant output and the reference model output asymptotically to zero. The idea of model-reference control systems was originally proposed by Whitaker at M.I.T. (1963) and the model-reference control has been referred to as the M.I.T. model. Whitaker considered a problem where the specifications were given in terms of a reference model which tells how the process output ideally should respond to the input command signal.

The design of a control system consists of the synthesis of a controller which most nearly makes the performance of the system coincide with the design specification. The design specification may have a lot of forms. In many instances, it is possible to synthesize an ideal system whose response satisfies the given specification and this ideal system is called the reference model for the system to be designed. The output of this model with respect to the system input represents the desired response for the system and is known as model response.
In general, a suitable reference model would be a system having the following transfer function:

\[ M(S) = \frac{\omega_n^2}{S^2 + 2\zeta \omega_n S + \omega_n^2} \]

where \( \zeta \) is from 0.6 to 0.707.

The reference model is part of the control system which is shown as Figure 5.1. However, the model response may or may not be attainable by

![Diagram of Model Reference Control System](image_url)

Figure 5.1. A model reference control system.
the actual system. For example, a control system which is inherently of third order (or higher) may have a specification of a second-order model in which the system response cannot track the model response. For a real control system, it may be a non-linear system and the model is linear. Thus an adaptive controller for model-reference techniques could be a better approach. Here the researcher proposes a novel controller in which designing the "C1" controller is simple and designing the "C2" controller is the key problem. This problem is nontrivial. It can be solved by adopting the adaptive learning algorithm which was mentioned in chapter three as the C2 controller. In this chapter, three examples showing the effectiveness of the proposed approach are given.

A Novel Approach for Model-Reference Adaptive Control

Considering the system of Figure 5.1, the plant P is actuated by the controllers C1 and C2. Then

\[ P\{ C1*(r - y) + C2*(M*r - y) \} = y \quad (5.1) \]

or

\[ [ I + C1*P + C2*P]*(y/r) = [C1 + M*C2]*P \quad (5.2) \]

In order to obtain model response, let \( y/r \) be \( M \),

\[ (I + C1*P + C2*P)*M = (C1 + M*C2)*P \quad (5.3) \]

or
Since the plant $P$ is taken as an unknown process or a black box, we assume that $P=1$ (ideal plant) and the equation 5.4 is reduced to be

$$C_1 = \frac{M}{1-M} \left( \frac{1}{P} \right)$$

(5.4)

The controller $C_2$ is an adaptive learning mechanism which is used to self-regulate the control action $u$. Let the output of $C_2$ be $u_{c2}$ and the output of $C_1$ be $u_{c1}$, then the control action $u$ is

$$u = u_{c1} + u_{c2}.$$  

The $C_2$ controller is designed by using a credit-assignment law (reinforcement learning) which has a scalar error value to represent the total performance of the controller. Basically, we try to force the error $e_0$ to be asymptotically zero as soon as possible by assigning positive credit (reward) to desirable action and negative credit (punishment) to undesirable action. The error $e_0$ is defined as

$$e_0 = r^*M - y.$$ 

The reaction of rewards and punishments is evaluated by the system in its effort to achieve its goal in which the integral

$$C_1 = \frac{M}{1-M} \frac{\dot{\gamma}}{S^2 + 2\zeta \omega_n S}$$

(5.5)
of absolute error is small. The adaptive learning law is expressed in Figure 5.2.

Procedure

Begin

{Rewards}

Threshold := sample_interval * sqrt(abs(r)) * e_{ss};
{ where the sample time <=1/(2*W_o^2) (msec), }
{ e_{ss} = 0.01 or 0.02 }
if e_0 > threshold then u_c := k_1*(1 + abs(r))
else if ( e_0 <= threshold) and ( e_0 > threshold/2)
    then u_c := (1 + e_0)/k_2
else if e_0 > 0 then u_c := 1 + e_0;

{Punishments}

if e_0 < -threshold then u_c := -k_1*(1 + abs(r))
else if ( e_0 > -threshold) and ( e_0 <= -threshold/2)
    then u_c := -(1 + e_0)/k_2
else if e_0 < 0 then u_c := -(1 + e_0);
end;

Figure 5.2. Adaptive learning algorithm for adapting the control action, where the constant factors can be chosen as k_1=5 and k_2=10

Simulation and Results

In this section, the proposed controller which was applied to three different control systems are described. The plants in the control systems are viewed as black boxes. The given plants are:

1) First-order simulated plants with dead-zone and
saturation,

2) Second-order simulated plants with dead-zone and saturation, and

3) Third-order simulated plants with dead-zone and saturation,

where the dead-zone and saturation is the same as the function in Figure 4.1. The aim of the exercise is to demonstrate the following:

a) The design of the controllers is relatively simple;

b) The objective of the proposed controller can be achieved in model-reference adaptive control.

The limitation of this approach is that the designer must have some knowledge of physical plants, but it is not necessary to know the model of plants. This knowledge is used to determine the suitable specification of a reference model.

**Simulation results of a first-Order plant with dead-zone and saturation**

For a given plant (\(=1/S\)) with dead-zone and saturation, the reference model \(M\) was chosen as \(\zeta = 0.6\) and \(W_n = 20\). The computer simulation result is shown in Figure 5.3 for sample time \(\Delta T=1\) msec. The performance of the proposed controller met the requirement very well and the step response of the controlled system is almost equal to the step response of the model. This means that the proposed controller forces the
Figure 5.3. Simulation of a first-order plant (1/S) with dead-zone and saturation

The model: $\frac{400}{s^2 + 24s + 400}$

The plant: $1/s$ with dead-zone & saturation

Sample time = 1 msec
error between the plant output and the model reference output asymptotically to zero.

**Simulation results of second-Order plants with dead-zone and saturation**

For second-order plants with dead-zone and saturation, the reference-model was given as $\zeta = 0.6$ and $\omega_n=25$. Computer simulations with sampling interval $\Delta T=2$ msec were used to evaluate the performance of the proposed controller for the given plant $G(S)$, where

$$G(S) = \frac{151.5}{S(S+73)}$$

The simulation results which are shown in Figure 5.4 illustrate that the system controlled by the proposed controller tracks the model signal very well. For the time-varying plant gain case, the gain was increased 50% at the beginning, and then gradually decreased to 75% at time=1 sec. The simulation results are shown in Figure 5.5. It is obvious that the performance of the proposed controller is very good. The error is also forced asymptotically to zero.

**Simulation results of third-order plants with dead-zone and saturation**

For a given plant $G(S)$, let the reference model be $\zeta = 0.6$ and $\omega_n=5$. The transfer function of the third-order
Figure 5.4. Simulation of a second-order plant with dead-zone and saturation

The model is $625/(s^2 + 30s + 625)$ and the plant is $151.5/s(s+73)$ with dead-zone and saturation.
Figure 5.5. Simulation results of a second-order plant with a time-varying plant gain
Figure 5.6. Simulation of a third-order plant with a dead-zone and saturation. The plant = \( \frac{1}{(S+1)(0.2S+1)(0.01S+1)} \)
Figure 5.7. Simulation of a third-order plant with a time-varying plant gain. The plant = 
\( \frac{0.75+N/250}{(s+1)(0.2s+1)(0.01s+1)} \)
Figure 5.8. Simulation of a model-reference fuzzy control system
plant is expressed as
\[ G(s) = \frac{1}{(s+1)(0.2s+1)(0.01s+1)} \]

The simulation results are shown in Figure 5.6. The transient response of the controlled system illustrates that it can track the model signal well, but it has a larger maximum overshoot and a little more delay. When the plant gain changes with time from decreasing 25% at the beginning to increasing to 75% at time=5 second, the controlled system can still track the model signal, but the maximum overshoot is too large. The simulation results are shown in Figure 5.7. This drawback can be overcome by using the expert FLC instead of the CI controller. Figure 5.8 illustrates the performance of the expert FLC controller for model-reference control. The simulation results show that the expert FLC can force the error asymptotically to zero.

**Summary and Discussion**

In this chapter, an adaptive learning mechanism which is suitable for model-reference control is presented. Its performance shows that it works very well for first-order, second-order and third-order plants with dead-zone and saturation. For the third-order plant with time-varying
parameter case, the transient response has a larger maximum overshoot. However this problem could be solved by using the expert FLC instead of the CI controller. In the design of a controller for model-reference control, designing the CI controller in Figure 5.1 is relatively simple, but determining the rewards and punishments in the adaptive learning mechanism is based on the heuristic. The sample interval will affect the steady-state error, the longer the sample interval, the larger the steady-state error, so the better sample interval $\Delta T$ could be chosen as

$$\Delta T \leq \frac{1}{2\pi \omega_n^2}.$$
CHAPTER VI. REAL-TIME APPLICATION AND RESULTS

Introduction

The expert FLC employing adaptive learning is implemented on a servo system to control angular position. Hardware requirements include the follows:

1) DC servo motor (80 W),
2) Potentiometer (10 kΩ),
3) Gear box (gear ratio 20:1),
4) DC power supply (±15V/2A),
5) AC power supply (24V/8A),
6) Servo amplifier,
7) IBM PC/AT,
8) Analog-To-Digital/ Digital-To-Analog converter, and
9) Rectifier and filter.

These experiment devices are shown in Figure 6.2. The aim of this experiment is to demonstrate that the performance of the expert FLC adaptive learning in servo system application is better than or at least as good as that of a PID controller.

"Servo" means that the output of a system must follow the input command. In an open-loop system, any change in load, amplifier gain, or any other system variable will cause a
deviation from the set point. In order to let the servo motor follow a desired function independently of changes in these variables, a closed-loop system is needed. The closed-loop system may be expressed as Figure 6.1.

In a servo system, the output is fed back and compared to the desired input. Any difference between the input and output is an error; the error is amplified to be a control action to correct the error. Ideally the closed-loop system is insensitive to variations in parameter, and performs

![Figure 6.1. A closed-loop fuzzy control system](image-url)
Figure 6.2. Hardware devices used in this experiment
correctly despite changes in load condition. However, in real cases, the response of a system depends on the closed-loop configuration and it may be overdamped, underdamped or even unstable. Thus an additional controller is needed to control servo systems.

In Figure 6.1, the angular position of output is converted to voltage by the potentiometer and then converted to a digital signal by the Analog-To-Digital converter. This digital signal is fed back and compared to the set point. The controller is used to convert the error between the input and output into a control action to drive the servo motor.

Servo systems are conventionally controlled by proportional-integral-derivative (PID) controllers. The PID controller is linear. That is, the PID equation must assume a linear relationship. The performance of the PID controller will be good enough by tuning the PID coefficients if the speed and accuracy requirements of the control are not critical or it is in an invariant control environment. However, a real control system has time-varying parameters and nonlinear components, tuning the coefficients cannot cope with these changes. Since the proposed controller is relatively insensitive to variation in control systems and it does not need a mathematical transfer function for formulating control rules, it would be more efficient to adopt this approach to control servo systems. The comparison of the performance
between the proposed controller and a PID controller is discussed in the next section.

Real-Time Results

The performance of the proposed controller in real-time application is shown in Figure 6.3 in which the input command is changed periodically from -4V (-120°) to 4V (120°) and the sample interval is 5 msec. The system response follows the input command with a critically damped response. The same conditions were tested with the PID controller which was tuned before the test by minimizing the integral of absolute error. Figure 6.4 presents the performance of the PID controller. The comparison of the performance between the proposed controller and the PID controller is indicated in Table 6.1.

In this experiment, the scaling factors of the expert FLC are fixed. The GE is set to 1, GCE is set to 5 and GU is set to 0.5. If the scaling factors are set so high that the system response become unstable, the proposed controller has the ability to reduce the scaling factors to force a system into a stable state. The system response changes from unstable to a stable state as illustrated in Figure 6.5.
servo-motor position control
controlled by expert FLC

tₜₛ=490 msec
ₐᵥ=0.6 percent
ₜₐ=150 msec
ₜᵣ=370 msec

Figure 6.3. Transient response of a servo system which is controlled by the expert FLC controller
servo-motor position control
controlled by PID
where $K_p=3$, $K_i=0.1$, $K_d=0$

Figure 6.4. Transient response of a PID control servo system with $K_p=3$, $K_i=0.1$ and $K_d=0$
Figure 6.5. Transient response of a servo system with high overshoot.
Summary and Discussion

The comparison of the performance between the proposed controller and the PID controller in the servo system application is described in this chapter. From Table 6.1, it is obvious that the expert FLC is at least as good as the PID controller. The delay time of the proposed controller is shorter than that of the PID controller. Although the settling time of the proposed controller is only 4% better than that of the PID controller, it is still possible to adjust the scaling factors to get an optimal result.

The PID controller is linear, while the fuzzy controller is nonlinear. Although the fuzzy controller uses the same type of inputs as PID (e.g., error and its derivative), they are processed nonlinearly. Therefore, the fuzzy controller can be viewed as a nonlinear PID controller. When the process dynamics are nonlinear, the fuzzy controller will be suitable to be used in nonlinear systems. Finally, it is important to note that designing this proposed controller does not rely on a mathematical model.
Table 6.1. Comparison of the performance between the proposed controller and the PID controller, where \( K_p=3 \), \( K_i=0.1 \), \( K_d=0 \) and the sample time=5 msec.

<table>
<thead>
<tr>
<th></th>
<th>Proposed FLC</th>
<th>PID</th>
</tr>
</thead>
<tbody>
<tr>
<td>( t_s ), msec.</td>
<td>490</td>
<td>510</td>
</tr>
<tr>
<td>ov</td>
<td>0.6%</td>
<td>0.7%</td>
</tr>
<tr>
<td>( t_d ), msec.</td>
<td>150</td>
<td>280</td>
</tr>
<tr>
<td>( t_r ), msec.</td>
<td>370</td>
<td>356</td>
</tr>
</tbody>
</table>
CHAPTER VII. SUMMARY, CONCLUSIONS, AND RECOMMENDATIONS

In this chapter, the results of the previous chapters are summarized and conclusions from the research findings are drawn. Several recommendations for further studies are also proposed.

Summary

Research background

Fuzzy logic based on fuzzy set theory can serve as a basis for reasoning with common sense knowledge. It is now finding wider and wider applications, from consumer products, industrial process control, pattern recognition and decision analysis to weather prediction, medical diagnosis and other application areas in which the underlying information is imprecise. Fuzzy logic control is one of the very important applications of fuzzy set theory. It is used to model human experience and knowledge. Recent applications of fuzzy logic control in many areas have pointed a way for an effective utilization of fuzzy control in the context of complex ill-defined processes that can be controlled by a skilled human operator without knowledge of their underlying dynamics.

In conventional control system design, system mathematical modeling plays a very important role. However
the cost of obtaining a mathematical model is computationally expensive and it is also difficult to obtain the exact mathematical model of a plant.

Fuzzy logic control can provide an algorithm that can convert human experience and knowledge into an automatic control strategy for a control system whose mathematical model is unknown. One key problem is that although many fuzzy logic controllers with approximate reasoning have been developed to emulate human decision-making behavior, few focused on an important aspect of human learning which is the ability to modify fuzzy decision rules or scaling factors based on experience. Thus developing a fuzzy logic controller by employing adaptive learning and adopting a more systematic way to define fuzzy rules is a challenging goal.

Research design

A computer-based fuzzy control system was designed for this study as a prototype to offer simulations and real-time experiments. Hardware requirements included the following: 1) DC servo motor (80 W), 2) Potentiometer (10 kΩ), 3) Gear box (gear ratio 20:1), 4) DC power supply (±15V/2A), 5) AC power supply (24V/8A), 6) Servo amplifier, 7) IBM PC/AT, 8) Analog-To-Digital/ Digital-To-Analog converter, and 9) Rectifier and filter. The implementation of the expert FLC controller that is used for simulations and real-time control
can be performed by approximating the sampled values of the controller inputs to the nearest discretization points and then they are scaled by the scaling factors which are adjusted by the adaptive learning algorithm to generate the indices of the look-up table $\Phi(.)$. The value $u_c$ taken from the look-up table adds the $u_c$ which is generated by the adaptive learning algorithm to obtain the control action $u$. The control action is expressed as

$$u := \Phi(.) + u_c,$$

and the control action $u$ can be used to drive the plant (process).

Considering the computational efficiency and real-time control, the look-up table (inference matrix) based on discrete universes, which define the output of a FLC for all possible combinations of the input signals, can be obtained by off-line processing. For designing the learning algorithm, we have adopted a reinforcement learning law in which a scalar error value $e$ represents the performance of the fuzzy controller. The goal of the learning law is to reduce the error to be zero as soon as possible by assigning positive credit (reward) to desirable action and negative credit (punishment) to undesirable action.
Research results and finding

In computer simulations, the comparison of the performance between the proposed controller and the conventional PID controllers were shown in Figure 4.3 to 4.15. The controlled systems included first-order, second-order and third-order plants with dead-zone and saturation components. The simulations show that, as expected, the performance of the proposed controller is better than that of the PID controllers. The expert FLC with adaptive learning can make the fuzzy control system achieve a shorter settling time, a faster rising time and a smaller overshoot. It was also found that the performance of PID controllers became more sensitive to the process model change or parameters varying with time. In the time-varying case, the simulation results showed that the control systems controlled by the proposed controller had shorter settling time and smaller maximum overshoot. The performance of the proposed controller was compared with that of Wu's controller. The simulation results demonstrated that the proposed controller had better performance in settling time, delay time and rise time.

For model reference adaptive control, an adaptive learning mechanism which is suitable for model-reference control is presented. The simulation results shows that this novel approach can force the error between the plant output and the reference model output asymptotically to zero. The
controlled systems included first-order, second-order and third-order plants with dead-zone and saturation. For the third-order plant with time-varying parameter case, the transient response has a larger maximum overshoot. However this problem could be solved by using the expert FLC controller instead of the CI controller.

In real-time applications, most of the fuzzy control experiments were concerned with slow control systems. In this study, the proposed controller was used to control a fast response servo system. The comparison of the performance between the proposed controller and the PID controller is shown in Table 6.1. It is obvious that the expert FLC is at least as good as the PID controller.

Conclusions

The problems stated in Chapter I were solved as a consequence of this study. First, a more systematic way to define the fuzzy logic rules was developed. Second, an attempt to develop a fuzzy logic control scheme by integrating human decision-making (by using approximate reasoning) and learning behavior (by using reinforcement learning) was made. The computer simulation results show that the learning capability of the proposed controller can shorten the rise time and reduce the steady-state error. The results also show
that the proposed fuzzy control scheme exhibits robustness properties with respect to parametric disturbance, instant load change and unstable systems. Thus the use of the fuzzy logic controller with adaptive learning can make up for the areas in which the conventional system doesn’t do as well. A novel approach for model reference control was developed in this study. The computer simulation results show that it can force the error between the nonlinear plant output and the reference model output asymptotically to zero. In real-time applications, the experiment results show that the fuzzy control scheme can be applied to fast servo systems.

Recommendations

According to the research results and findings, the following recommendations for further studies are suggested:

1. One of the important concepts concerning the properties of control systems is stability. The analysis of system behavior in fuzzy control systems with adaptive learning is worthy of further study.

2. Many fuzzy logic controllers by using approximate reasoning have been built to emulate human decision-making behavior, but emulating human learning behavior remains a challenging goal that needs further study.
3. A self-learning controller that combines the fuzzy logic control scheme and neural network theory is worth studying.
REFERENCES


fuzzy controller in a warm water plant. *Automat.*, 12(4), 301-308.


design of fuzzy control systems. *Fuzzy Sets and Systems,* 45, 135-156.


in Proc. 2nd IFSA Congress, Tokyo, Japan, 349-352.


I would like to express my warmest appreciation to my major professor, Dr. William D. Wolansky, for his guidance and kindly assistance that have made my time at Iowa State University both pleasant and productive. I also would like to express my sincere appreciation to my committee members, Dr. Hsien-sen Hung, Dr. Robert E. Strahan, Dr. John Newton Riley, Dr. Johnny S. Wong for their insight and valuable guidance.

I would like to thank to Dr. Dar-Chin Rou, the chairman of my department in Taiwan, for his encouragement and support. Special thanks to my friend Dr. Chin-Ming Hung for suggestions about this study. Finally, I would like to thank my wife, Wan-Chen Chang, and my sons, Po-Gin and Chung-Shiuan, for their love and prayer for me during the time of my graduate career.