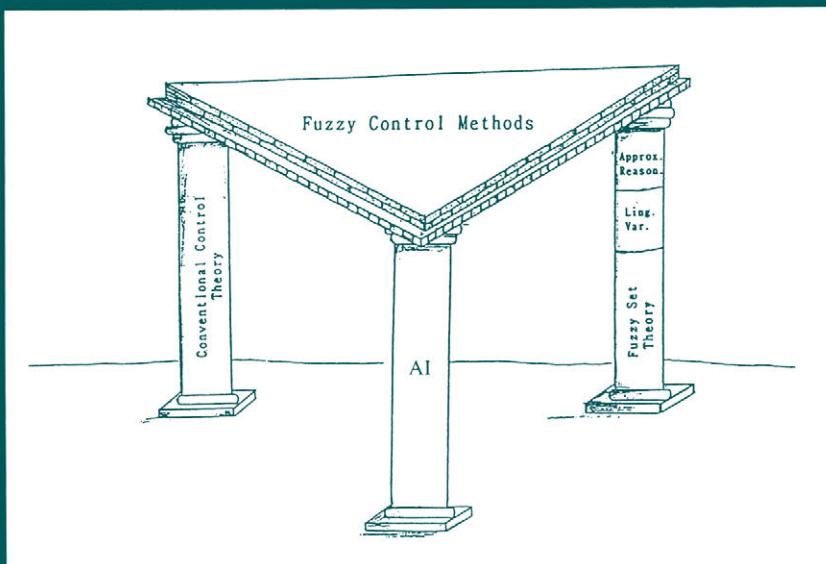


THE FOUNDATIONS OF FUZZY CONTROL

HAROLD W. LEWIS, III



IFSR International Series on
Systems Science and Engineering
Volume 10

THE FOUNDATIONS OF FUZZY CONTROL

International Federation for Systems Research International Series on Systems Science and Engineering

Series Editor: George J. Klir

State University of New York at Binghamton

Editorial Board

Gerrit Broekstra

*Erasmus University, Rotterdam,
The Netherlands*

John L. Casti

Santa Fe Institute, New Mexico

Brian Gaines

University of Calgary, Canada

Ivan M. Havel

*Charles University, Prague,
Czech Republic*

Manfred Peschel

Academy of Sciences, Berlin, Germany

Franz Pichler

University of Linz, Austria

Volume 7 *FACETS OF SYSTEMS SCIENCE*

George J. Klir

Volume 8 *THE ALTERNATIVE MATHEMATICAL MODEL OF
LINGUISTIC SEMANTICS AND PRAGMATICS*

Vilém Novák

Volume 9 *CHAOTIC LOGIC: Language, Thought, and Reality from the
Perspective of Complex Systems Science*

Ben Goertzel

Volume 10 *THE FOUNDATIONS OF FUZZY CONTROL*

Harold W. Lewis, III

Volume 11 *FROM COMPLEXITY TO CREATIVITY: Explorations in
Evolutionary, Autopoietic, and Cognitive Dynamics*

Ben Goertzel

IFSR was established "to stimulate all activities associated with the scientific study of systems and to coordinate such activities at international level." The aim of this series is to stimulate publication of high-quality monographs and textbooks on various topics of systems science and engineering. This series complements the Federation's other publications.

A Continuation Order Plan is available for this series. A continuation order will bring delivery of each new volume immediately upon publication. Volumes are billed only upon actual shipment. For further information please contact the publisher.

Volumes 1-6 were published by Pergamon Press.

THE FOUNDATIONS OF FUZZY CONTROL

HAROLD W. LEWIS, III

*Fukushima University
Fukushima, Japan*

Springer Science+Business Media, LLC

Library of Congress Cataloging-in-Publication Data

On file

ISBN 978-1-4899-1856-7 ISBN 978-1-4899-1854-3 (eBook)
DOI 10.1007/978-1-4899-1854-3

©1997 Springer Science+Business Media New York
Originally published by Plenum Press, New York in 1997
Softcover reprint of the hardcover 1st edition 1997

All rights reserved

10 9 8 7 6 5 4 3 2 1

No part of this book may be reproduced, stored in a retrieval system, or transmitted
in any form or by any means, electronic, mechanical, photocopying, microfilming,
recording, or otherwise, without written permission from the Publisher

To two wonderful young men,
Tosh and Tats

Preface

This book was designed to be useful in several different ways and for various groups of people. It can be a textbook, a conceptual how-to book, and a sourcebook of alternative ideas for the design and analysis of systems. It can be used by students and educators, by practicing engineers and managers, and by scientific researchers in any discipline where there is a need to understand complex systems. The book was designed not just to support these various goals, but to be uniquely easy to understand in all cases.

Partly as an effective way to address these goals, and partly because I believe that fuzzy control could use a breath of fresh air, this book takes a unique approach on many points. Several of these will be introduced more fully in Chapter 1, but we can briefly summarize them here.

1. There are introductory chapters to expose the student to conventional control concepts and to artificial intelligence (AI), as well as the usual background chapter on fuzzy set concepts.
2. An alternative mathematical formalism is presented for describing the approximate reasoning process. This formalism is arguably easier to understand than the standard one, but still is just as mathematically rigorous. The two formalisms are compared, and a proof is given to show that they lead to equivalent results. Also, both a detailed numerical example in Chapter 5 and a simple Pascal language program in Appendix B are provided as examples of just how simple it is to implement the alternative perspective for given circumstances.
3. The question of why fuzzy control has been to date very popular in some countries, such as Japan, and quite unknown in other countries, such as the United States, is reexamined. A unique viewpoint is presented in response to this question. This viewpoint is less anti-Western and less culturally deterministic than the viewpoint often presented. The book never becomes too philosophical on this point, but this view is discussed at various points throughout the text, particularly in Chapters 1, 4, and 5.
4. Much of the progress in fuzzy control methods over the past few decades has come from Japan. I was able to use my knowledge of the Japanese language, my association with Japanese fuzzy research groups, and personal acquaintance with several of the key researchers to provide a some-

what more direct view of the Japanese perspective than normally would have been possible in a book written by a Westerner.

5. Both the section on key historical examples of fuzzy control applications (Section 5.8), and the chapter on examples of variations in the design methods (Chapter 6) are quite unique in providing a detailed focus on the main innovations involved in each key application, but without becoming excessively technical.
6. One unique point about Part II is the nature of the test plant used in the research. On the one hand, it is an actual complex physical device. On the other hand, it was chosen, developed, and used in such a way that the research could easily be duplicated on a very small research budget. This is unique in the sense that most fuzzy control research either is assumed to require major budgets or is based on rather imaginary sorts of examples.
7. Chapters 8, 9, and 10 are all very detailed in their discussions of neural modeling of the plant, genetic auto-tuning, and performance testing, respectively. They present the concepts and the rationale, as well as describing one approach to the actual mechanics of the process.
8. The main question upon which all of Part II is based is itself somewhat unique. Rarely in the literature on fuzzy control is dynamic compensation discussed explicitly as a matter deserving extensive examination.

The only prerequisite knowledge assumed is a basic grasp of mathematical concepts. Specifically, I recommend the following.

1. The type of introductory course, usually called discrete mathematics, which is taught early in most programs for computer science and systems science.
2. At least some exposure to calculus.

Even these need not be considered absolute requirements, but a full comprehension of most chapters would be nearly impossible for those students unfamiliar with such concepts as Cartesian product, function, relation, integral, and so on. On the other hand, students need not be alarmed if they are unfamiliar with differential equations, partial derivatives, or other concepts of which only brief and occasional mention is made in the text.

Although the book was designed to be useful on many levels, it will be particularly clear here that the time required to master the contents will depend on the background of the reader. An engineer already quite familiar with control concepts and the basic AI methodologies may find it possible to breeze through this book in a very few hours, and to go away with a much greater understanding of the nature of fuzzy control. On the other hand, for students with less confidence in their background knowledge or mathematical skills, it may be appropriate to consider this book as the basis for a semester-long course.

The contents will be easiest to understand if the chapters are read in sequence. In varying degrees each chapter builds upon concepts found in preceding chapters. Readers with extensive background knowledge may be able to omit some parts of several chapters, particularly of Chapters 2, 3, or 4. However, it may be better to glance through a chapter with an eye on the style of presentation, rather than to omit the chapter altogether. Also, some students insist on seeing detailed examples of actual applications as soon as possible in their studies of technical fields. Such students may wish to refer ahead to relevant sections in Part II during their first reading of Part I. There is certainly no harm in this, though in such cases it may be beneficial to read Part II again after completing Part I. Each chapter begins with quite detailed introductory comments concerning the content of the chapter, the relevance of that content, and the style of presentation. It is recommended that readers refer to these introductory comments for each chapter when making individual decisions on how to proceed through the book.

The notation used in this book can be considered standard with a few exceptions. Section 4.1 introduces two different forms for representing a membership or a characteristic function. One of these forms is used through the end of Chapter 4, and the other form is used in all succeeding chapters. There are justifications for this use of two different forms, and I explain this in the text, but some may still object to this as inconsistent. Let me just point out here that the use of multiple notational schemes within the same book is actually much more common than one may assume. Consider for example that nearly every calculus textbook uses two different forms of notation for representing derivatives.

One other point relates to the notation for indicating that one must take the maximum (*supremum*) of a list of numerical values. I have found it is easier to avoid confusion if one writes “max” when the list of values is explicitly enumerated, and writes “Max” when the enumeration is implied in a way analogous to the use of the sigma sign for taking sums. A similar distinction is made between “min” and “Min.” This is only a very minor deviation from the norm, and these distinctions are quite helpful, especially when reading through many details such as those found in the proofs in Section 5.5. Appendix C provides a summary of the notational forms used throughout the text as well as the frequently used acronyms.

Finally, under the assumption that this may make it easier to understand the goals of this book, let me explain how it developed. Most of this book is based on what I learned in various informal settings, rather than as formal education. Undoubtedly, this is the real reason that I could not be satisfied with taking a conventional approach in explaining fuzzy control.

Harold W. Lewis, III
Troy, New York

Acknowledgments

It is a pleasant task to express gratitude to many people for the parts they played in producing this book. First I am thankful for the advice and encouragement I received from several of the faculty in the Department of Systems Science at Binghamton University, particularly for the help of my mentor, George J. Klir, Distinguished Professor. I thank Kyoji Hoshino and several others in the Faculty of Economics and in the Information Processing Center at Fukushima University. Special thanks go to Abu Osman and other members of the Faculty of Mathematical and Computer Sciences at Universiti Kebangsaan Malaysia for hospitality, encouragement, and the use of their excellent library. Osman especially encouraged me to express my own perspective freely on various issues. I am also grateful to the Department of Decision Sciences and Engineering Systems at Rensselaer Polytechnic Institute for providing me with still another academic home-away-from-home during the final stages of writing, and I especially thank a wonderful colleague, Mark Embrechts.

Thanks go to everyone at Plenum, particularly to Danielle McPhail and L.S. Marchand.

Figures 3.1 and 3.2 first appeared in one of my papers previously published in *The Shogaku Ronshu: Journal of Commerce, Economics, and Economic History* and are used here with permission of the publisher. Several chapters of this book began during one of my stays in Malaysia, and I wish to thank all of my wife's family for help and encouragement during that time, especially Ah Wan, Michael, Ivan, and Chang Kiat. The back office of C.K.'s video rental shop turned out to be an incredibly productive environment. I am also grateful to my own family, especially to my parents, and to Dave and Mary.

Most of all, I thank my wife, who not only encouraged me, but also provided important advice during the writing of this book.

Harold W. Lewis, III

Contents

Part I. Principles of Fuzzy Control

Chapter 1. Introduction	3
1.1. One Perspective on Fuzzy Control	3
1.2. Overview of Part I	5
1.3. Overview of Part II	7
Chapter 2. Conventional Control	9
2.1. Heuristics and Control	9
2.2. Feedback	10
2.3. Central Role of Models in Conventional Control	14
2.4. Dynamic Compensation	16
2.5. Implementation of Conventional Control	20
2.6. Performance Specification	21
2.7. Practical Disadvantages of Conventional Control	26
Chapter 3. Fuzzy Control as Artificial Intelligence	29
3.1. Concerning Definitions of Artificial Intelligence	29
3.2. Emphasis on Domain-Specific Knowledge	35
3.3. Nature of Human Expertise	36
3.4. Synergistic Model of Applied Artificial Intelligence	41
3.5. Machine Learning	50
3.6. Fuzzy Control and Artificial Intelligence	58
Chapter 4. Some Relevant Aspects of Fuzzy Set Theory	59
4.1. Fuzzy Sets	60
4.2. α -Cuts, the Extension Principle, and Other Concepts	67
4.3. Fuzzy Numbers	70
4.4. Linguistic Variables or Fuzzy Relations of Meaning	73
4.5. Fuzzy Inference (Logic) for Approximate Reasoning	77
4.6. Closing Remarks on Fuzzy Set Methods	92

Chapter 5. Classical Fuzzy Control Design and Implementation	95
5.1. High-Level System Design	95
5.2. Naive Physics and Fuzzy Control Rules	97
5.3. Linguistic Variable Design	101
5.4. Perspective on Inference Algorithms for Fuzzy Control	103
5.5. Comparison of Our Perspective with the Standard One	113
5.6. General Notes on Implementation	122
5.7. Tuning of Fuzzy Control Devices	125
5.8. A Few Examples of Classical Fuzzy Control Applications	126
5.9. A Further Comparison of Fuzzy and Conventional Control	136
Chapter 6. Some Common Variations in Fuzzy Control Design Methods	139
6.1. Fuzzy Singleton Method	140
6.2. Variable Reduction	141
6.3. Variations in Inference Algorithms	143
6.4. Plant Models for Fuzzy Control Design	144
6.5. ‘‘Neuro & Fuzzy’’	148
6.6. ‘‘Neuro-Fuzzy’’	152
6.7. Genetic Algorithms in Fuzzy Control Design	154
6.8. Dynamic Compensation in Fuzzy Control	156

Part II. Case Study on the Utility of Dynamic Compensation in Fuzzy Control

Chapter 7. Miniature Steam Engine as a Test Plant	161
7.1. Brief Introduction to Steam Engine Principles	162
7.2. Steam Engines as Control Problems	164
7.3. Our Miniature Steam Engine and Its Modifications	166
Chapter 8. Simulation Model of the Test Plant	173
8.1. Concerning Approaches to the Modeling of Plants in Fuzzy Control Applications	173
8.2. Details of the Neural-Based Model Design	178
8.3. Evaluation of Model Accuracy	185
Chapter 9. Fuzzy Control Designs with Auto-Tuning and Various Forms of Dynamic Compensation	191
9.1. High-Level System Designs for Fuzzy P, PD, PI, and PID Steam-Engine Controllers	192

9.2. Fuzzy Inference Algorithm Used	195
9.3. Control Rules for Fuzzy P, PD, PI, and PID Steam-Engine Controllers	197
9.4. Linguistic Variable Definition and Auto-Tuning of Fuzzy Steam Engine Controllers by Genetic Methods	207
9.5. Final Designs Resulting from Auto-Tuning	218
Chapter 10. Final Test and Results	227
10.1. Test Input Functions Used for Comparisons	227
10.2. Results of Simulations	236
10.3. Further Examination of the Auto-Tuning Process	256
Chapter 11. Conclusions and Prospects	261
11.1. Summary of the Nature of a Derivative Term in Fuzzy Control Designs	261
11.2. Summary of the Nature of an Integral Term in Fuzzy Control Designs	262
11.3. Some Practical Implications of the Dynamic Compensation Results	264
11.4. Concerning the Rationale of Fuzzy Control	265
11.5. Rational Approach to Research in Fuzzy Control and Other Applications of Fuzzy Set Theory	268
11.6. Prospects for Further Applications and Research	269
Appendix A. Concerning Definitions of Artificial Intelligence	271
Appendix B. Sample Pascal Program Illustrating Fuzzy Inference Algorithms by a Simple Example with Three Inputs and Two Outputs	277
Appendix C. Symbols and Acronyms	283
References	285
About the Author	291
Index	297

THE FOUNDATIONS OF FUZZY CONTROL

PART I

Principles of Fuzzy Control

CHAPTER 1

Introduction

Chapter 1 introduces the overall scope and approach of the book and explains the perspective used in the chapters that follow.

1.1. One Perspective on Fuzzy Control

One of the most challenging classes of problems dealt with on the systems level is the analysis and design of control devices. Fuzzy control is an unconventional approach for dealing with that problem class. We can view fuzzy control methods as a concrete branch of engineering or in a slightly abstract way, as a branch of systems science with important implications for every field where complex systems are studied. In this book we present a solid background in the principles of fuzzy control in a way that is easier to understand than most of the literature to date on the subject.

Although fuzzy control is quite a new approach, its effectiveness is now well-proven. Over the past two decades, engineers have applied fuzzy control methods very successfully, both to large-scale systems, such as cement kilns, a subway train system, and waste treatment stations; and on a smaller scale, including over a hundred different models of home appliances. In many cases fuzzy control mechanisms automated control of the subject plant much more completely than with conventional methods. At the same time fuzzy control has naturally led to a clearer understanding of how humans exercise control and make decisions. Conceivably we can still question the philosophic or mathematical soundness of fuzzy set theory in some general way, but it is certainly no longer possible for anyone knowledgeable of applications developments to remain skeptical of the basic effectiveness of the fuzzy control approach.

Yet interest in fuzzy control continues to be much more widespread in some countries than in others. Japan is clearly the country where interest in this field has been most prevalent, particularly as it relates to practical applications. For several years now at some of the major Japanese universities, even many undergraduate engineering students have developed fuzzy control applications for their termination projects or theses. A few years ago interest in fuzzy control technology spread even beyond the engineering and systems science communities to develop into a popular fascination shared by Japanese society at large. In the United States on the

other hand, despite U.S. dominance in many other new technological fields, there are still only relatively few engineers who know or care very much about fuzzy control methods. While other technologically developed nations have shown a somewhat greater level of interest than the United States—at least on a per capita basis—in applying fuzzy set theory, no other country has yet matched Japan in widespread implementation of fuzzy control technology.

It is an interesting phenomenon when a proven, practical technology continues to enjoy much greater popularity in some parts of the developed world than in others, and it is only natural to be curious about the reasons behind such a phenomenon. Unfortunately this type of question leads to speculation, but it is very difficult to prove anything. Many people have suggested that this phenomenon is primarily a matter of culture: People of different cultures have fundamentally different philosophical assumptions, which lead them to view various technologies in fundamentally different ways. Some seem to hold this view to such an extreme that they believe there is no need to look for a better explanation. Though we cannot disprove this theory of cultural determinism, we prefer an alternative explanation, certainly a more optimistic one, and arguably a more plausible one as well.

It seems to us quite credible that fuzzy control methods are now most popular in Japan primarily because early supporters in Japan were more skillful at education, public relations, and other aspects of promotion. Of particular interest to us are the educational aspects: We suspect that Japanese educators and researchers simply worked harder at presenting fuzzy control concepts in a straightforward, pragmatic, and appealing way than did educators and researchers in other countries.

We believe that if appropriately taught, undergraduate as well as graduate students, even those with only moderate mathematical skills, can develop a very solid understanding of fuzzy control principles. One advantage of fuzzy control is that its principles are more easily mastered and effectively applied than is true of conventional control theory. Therefore fuzzy control should occupy relatively basic slots in various engineering and systems science curricula, which is precisely how many Japanese educators approached the subject.

We do not suggest that there is a completely uniform approach to teaching fuzzy control in Japan, nor do we imply that a uniform approach should be followed precisely by educators around the world. This book offers only a personal perspective on the question of how to best understand fuzzy control principles. In some ways this perspective is inspired by Japanese approaches, and in some ways it is quite independent of those approaches. We have worked hard to develop a form of presentation that is as straightforward as possible but also very general and complete.

Unfortunately, even a new field of study may have an almost uncanny tendency to rapidly develop its own sense of orthodoxy, and there are many potentially detrimental results. Therefore, to offer a fresh perspective may require some deviations from the conventional forms. We all know that it is ill advised to make

needless or reckless breaks with well established conventions, but we also should know that if one has something important to say, then there is no sense in holding back too much in finding a way to express it. It is our goal to write this book in that spirit. We believe that the result will be a book that is useful and quite unique in several ways.

As a starting point and a conceptual basis for later discussions, we emphasize similarities between fuzzy control and various other technologies. Such an emphasis is unique in the existing literature but easy to support conceptually, and it contributes to a better understanding of fuzzy control. We refer to this as our general approach to the field. With this type of presentation, anyone who has studied related fields is able to understand both the nature and purpose of fuzzy control methods in only a few hours. Furthermore anyone who has not previously studied related fields can acquire some basic knowledge of these fields while studying fuzzy control. Our approach more importantly, leads to thinking about fuzzy control in the most comprehensive and insightful manner possible.

A second important aspect in our viewpoint is its emphasis on a conceptual or intuitive understanding of fuzzy control. This is easily justified when we recall that fuzzy control evolved as an intuition-based approach to the problems of automatic feedback control. Therefore in many cases, we favor conceptual explanations over mathematical ones even when a mathematical explanation is more customary or concise.

Part of our conceptual emphasis involves carefully explaining the rationale behind each method and concept presented. Much of the skepticism surrounding fuzzy control and fuzzy set theory may be due to insufficiently articulating the reasons behind the methods. Our goal is to be neither too shy nor too extravagant in writing equations. While we assume previous knowledge of basic discrete mathematics and calculus, we do not overuse mathematical forms to appear respectable, nor do we assume that no verbal explanation should accompany equations.

Also, where it is appropriate to use mathematical formalisms in the presentation of a given concept, we will strive to use the best formalism; that is, a formalism which is sufficiently general and also as simple as possible. This principle may cause us to diverge still further from conventional presentations, as when we discuss approximate reasoning mechanisms in Chapter 5.

1.2. Overview of Part I

The purpose of Part I is to present a detailed description of the principles of fuzzy control. In terms of detail, several different approaches to fuzzy control have proven practical. However, for ease in understanding, in the first several chapters we will start by emphasizing only the classical or Mamdani methodology. This is

the earliest practical approach to fuzzy control as developed by Mamdani and his colleagues at Queen Mary College in the early 1970s. Chapter 6 explains how more recent approaches vary from classical methodology.

There is still some degree of confusion concerning the nature of fuzzy control even in the systems science and engineering research communities. It is possible to be quite knowledgeable of both conventional control theory and fuzzy set theory yet still lack a clear understanding of how these theories are related to the development of the fuzzy control methodologies. Fuzzy control as it is usually presented must seem more confusing still to those who have not yet studied these related fields. This confusion can be largely avoided if we understand from the start that fuzzy control is primarily a brilliant combination of elements from the following traditions:

- Conventional control theory
- Artificial intelligence or specifically knowledge representation methodologies
- Fuzzy set theory or specifically the application of the concepts of linguistic variables and approximate reasoning

When viewed in this way, the methodology of fuzzy control seems much more straightforward than it would otherwise, and one can also understand its similarity to other technologies. This viewpoint is represented by Figure 1.1.

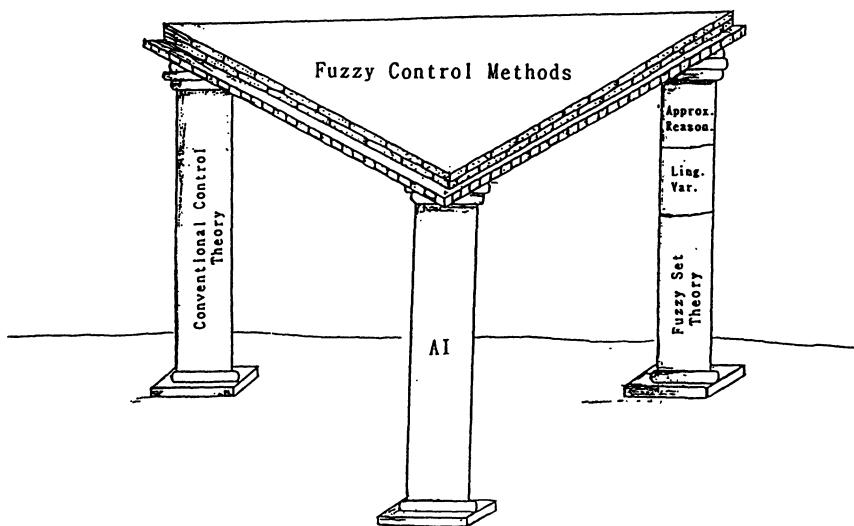


Figure 1.1. The conceptual basis of fuzzy control. AI (Artificial Intelligence), Approx. Reason. (Approximate Reasoning), and Ling. Var. (Linguistic Variables).

Simply stated, fuzzy control uses detailed structures and evaluation criteria from conventional control, but instead of relying on precise models of the physical plant as conventional control does, it uses other model types, such as collections of heuristic knowledge about system behaviors. Fuzzy set theory is necessary because these alternative model types are ineffective in practice unless they are based on linguistic variables or other applications of fuzzy set theory. We consider each of these aspects in greater detail in the chapters that follow.

1.3. Overview of Part II

Part II can be viewed primarily as a case study of fuzzy control methods. It is designed to serve as both a detailed example of how the principles can be put into practice, and an investigation into the general significance of dynamic compensation in the context of fuzzy control designs.

Dynamic compensation, a collection of concepts from conventional control theory, is described in detail in Chapter 2. Here, let us emphasize only that it is a particularly good example of the many connections between fuzzy control and previously existing technologies. Researchers in fuzzy control surely have long been aware of this connection, but they only rarely have discussed it explicitly.

Certainly, it is difficult to discuss the significance of dynamic compensation to fuzzy control in a way that is both completely general and absolutely conclusive, but the case study in Part II is designed to be as general and as conclusive as is possible in a single study. The investigation will center around testing differences in performance between fuzzy control devices with and without various forms of dynamic compensation. However, it is often said that experimentation does not constitute a proof in general, and this is even much more true when the subject of the experiments is essentially just a comparison of different approaches to design; and heuristic-based approaches at that. There is no way we know of to demonstrate absolutely that one design approach was applied as skillfully as another in the test designs. However, even as we acknowledge this unavoidable limitation, we nonetheless believe the study results will be significant.

We have chosen as the test plant for this case study—for several reasons to be explained later—a very small, alcohol-fueled, steam engine over which we will exercise nonlinear, multivariate feedback control, controlling both speed and boiler pressure by using a throttle and a method of regulating the air flow to the burner. Thus there are two controlled and two controlling variables.

We will use a neural model of the actual physical test plant and our own genetic technique for auto-tuning the various designs. Furthermore, the comparisons here will be based on the auto-tuned designs in virtual connection with the neural model of the steam engine. There are two benefits to this detailed approach: First, although some element of subjectivity in the comparisons is still unavoidable, we will at least

have made the design process as objective as possible. Second, as part of our detailed example, we will carefully consider some of the relatively recent advanced techniques in the field of fuzzy control.

The case study in Part II will also be unique in another significant way. To date, most experimentation on fuzzy control methods either required well-equipped laboratories with budgets in the hundreds of thousands of dollars at least, or involved only simulations on simple, abstract models. The sample test plant we use here will be a useful compromise. The experiments are based on an actual physical system in all its realistic complexity, and yet all of this can be duplicated at such a low cost that even a hobbyist might consider it. One would need only a very inexpensive personal computer and a few hundred dollars worth of other hardware and a language compiler. Admittedly, the experiments here are still based on simulations of a sort, but with a few thousand dollars worth of interfacing hardware and some skill in connecting it, one could overcome that limitation as well.

CHAPTER 2

Conventional Control

Since conventional control theory is one field which has much in common with fuzzy control, Chapter 2 will discuss the major points of similarity. The information contained here will seem quite elementary to anyone who has previously studied control theory in a serious way, but our presentation is somewhat unique in that we are emphasizing the conceptual perspective of that information. This is important for a better understanding of several matters relating to a general knowledge of fuzzy control. These matters include an understanding of just why control problems are difficult, which aspects of control theory are most intuitive, and what the drawbacks are of the conventional theory.

2.1. Heuristics and Control

Wiener's term, cybernetics, currently seems to be out of vogue again, but of the many concepts suggested by this word, at least two are useful here. The first of these concepts is that it is important to remember that a science of control or communication must have implications relating to biological systems, several of the social sciences, and any number of other fields outside the realm of engineering. Some proponents of cybernetics may have attributed too grandiose a significance on these implications, but hype is not unique to the cybernetics community, and it remains true nonetheless that some degree of emphasis on the broader implications of control theory is useful. The second major concept from cybernetics that provides a useful emphasis for us here is its consideration of the dialectical nature of control problems; the tendency of these problems to be in some sense intuitively simple and at the same time to be extremely difficult to analyze rigorously.

For both of these reasons, cybernetics held an implicit suggestion for the possibility of a control methodology based on intuitively derived heuristic knowledge. This is precisely what fuzzy control is, at least in the nature of its development to date. We discuss the meaning of heuristic knowledge in much greater detail in Chapter 3, but let us assume for the moment that it is the sort of knowledge that can be applied without completely rigorous analysis. Indeed, practical experience has also been suggestive of the potential for a heuristic approach to control. Ordinary humans, without giving a thought to mathematical models, are capable of learning, with only a little practice, how to control effectively many types of processes in

artificial systems. Furthermore, despite the heavy emphasis on analysis in conventional control theory, in actual practice conventionally designed control devices are often tuned mainly on the basis of experience and intuition. All of this means that if one were to develop a new approach to the design of automatic control devices that emphasized heuristics over rigorous analysis, then that approach, although new, could nonetheless be said to flow smoothly from connections with earlier thinking on the science of control. We will later discuss the development of fuzzy control as precisely such an approach. This already suggests, admittedly in a vague way for now, part of the nature of the relationship between fuzzy and conventional control theories. There are some more specific connections and similarities which we can discuss in detail only after a careful conceptual treatment of conventional control.

2.2. Feedback

A discussion of control theory must begin with the concept of *negative feedback*. Negative feedback simply means that if we wish to control a particular output of a dynamic system (usually referred to as the plant) subject to noise and changes in desired level, then we would normally wish to do it in the following way. We would continuously or frequently measure that output of the plant (the controlled variable) with some appropriate type of gauge or sensing device, and subtract the value we wish the output to be from the measured value, referring to the difference as the error. We would then, also continuously or frequently, make adjustments to one of the inputs of the plant (the controlling variable) by means of an automated actuator or manual adjuster in such a way as to try to change the output in a direction opposite to the direction of the error so as to attempt to make the error equal zero. Whenever a dynamic system is entirely controlled automatically, negative feedback is almost certainly the most basic element of its structure. In manually controlled systems, negative feedback is also usually taken as the intuitively obvious basis for action.

At this point, we should specify the types of control applications we will consider. We are thinking here primarily of control in the regulatory sense, and that is why we state that feedback or *closed loop control* can almost always be assumed to play a central role. Outside of control in the strictly regulatory sense, there is a branch of engineering called *sequential control*. Automatic sequential control devices simply require one operation to be completed before another is initiated. Some aspects of elevator control systems are sequential in nature. For example, a circuit with one or more relays is designed to prevent the elevator from moving before the door has closed completely. The logic of sequential control is usually quite simple to work out without any advanced systems techniques so this field tends to be relatively simple in practice. However, often one automated device will contain both sequential and feedback control elements. For example, elevators also

typically have feedback-based mechanisms for keeping the floor of the elevator level with the floor of the building despite changing loads when the elevator stops at a given floor.

Even within regulatory control we can be slightly more specific as to what we consider to be a typical type of application. In some circumstances, such as most types of thermostatically controlled home-heating systems, it is more practical to operate the furnace or other types of actuating devices only at full power or else to turn it off completely. This is a special type of feedback control system called *on-off control*. It is not considered the most typical form of control problem, and there are many special considerations involved in the design of these systems, so we will mostly overlook them here. We assume the more common situation where the actuating device can be adjusted more or less continuously; a good example is the type of cruise control device installed in many automobiles for automatically controlling speed in highway driving.

A closely related topic, one considered by cybernetics and conventional control theory in general, is *servomechanism*. It is conceptually useful to define a servomechanism as a feedback control device in which the desired value for the controlled variable changes quite frequently due to the nature of the application—to enhance the power of a human operator's actions. A familiar example is the type of power steering found in some automobiles. The place of servomechanisms within the broader context of control theory is an interesting one. On the one hand, it seems that servomechanisms are only a special case and should be treated as such, but on the other hand, an examination of the history of control theory reveals that servomechanism applications have long been thought of as a very central part of the theory. It is also interesting to note that many of the conventions of control theory—the way in which control engineers tend to draw feedback schematics, the way they identify system inputs and outputs, and the way they think of transient response—make sense only when one views them in the context of servomechanism applications, not in the context of regulatory control in general.

Because the concept of negative feedback is so intuitive, many people who have not studied engineering simply assume that all of control theory must be very straightforward and that the design of control systems must be a simple business if one can only develop the necessary sensors. Unfortunately, this is not true. Although negative feedback is an important basis for most regulatory control, it is almost never possible to develop an effective automatic control device only by a naive application of this one concept. In fact, even when the hardware problem of sensor development is left aside, control has been one of the most challenging aspects of engineering. The primary reason that it is usually impossible to control a dynamic system perfectly even if we use negative feedback is that *time lags* interfere with the feedback process. First there is normally some degree of time lag within the plant itself, so that the output value that we wish to control will not react immediately to the changes made in the input. In some cases there may also be significant

delays in the various parts of the control apparatus, including the sensor or gauge, the actuator, and either the automatic control device itself or the reactions of the human controller in the case of a manually controlled system. Because of these time lags, negative feedback typically adjusts the plant not according to its current conditions, but to conditions that existed, say, many milliseconds earlier. In a strictly electronic system, time lags may be more on the order of micro- or nanoseconds, and in large systems entailing thermal processes the time lags may be measured in minutes, but in all cases the relative effect of time lags is similar. If one broadens the context of control theory to discuss macroeconomic regulatory processes for example, then time is measured in weeks or months, but the same patterns can still be seen in the behavior of the feedback process.

Since early in the history of control technology, at least from the time of Watt's flyball steam engine governors, it has been recognized that a feedback control system can and usually does fall short of perfection in three different respects—stability, responsiveness, and accuracy. Each of these imperfections can be blamed either directly or indirectly on time lag.

Let us consider a plant that is quite slow to respond and assume that there is currently a moderately large error in the positive direction. The source of such an error is not of real concern to us at this moment, but let us say, simply to show the relevancy of this scenario, that a large positive error could be caused by someone suddenly adjusting the desired value lower. Alternatively it could occur when the system is first engaged or it reacts to some major source of noise. An example of noise in the case of an automobile cruise controller would be where the vehicle speed began to increase due to coming upon a sudden downgrade in the road. Regardless of its source, if there is a moderately large positive error, our control device or manual operator will make an adjustment to the system in such a way as to attempt to decrease the plant output value so that the error equals zero. However, the plant will not immediately respond to this correction. Therefore, there may continue to be a large positive error for some time. A poorly trained operator or poorly designed control device may respond to this situation by continuing to make the same strongly negative adjustment. When the plant finally does respond, the cumulative effect of the continued, forceful adjustment will be too much, and the plant will overshoot the desired value by a large amount and then produce now a large negative error. If the system is very poorly designed, then the output may continue to oscillate wildly above and below the desired value, possibly increasing in magnitude on each pass until the system goes beyond some practical limits. Such a situation is referred to as a *stability problem*.

When actually designing a control device it is often useful to discuss stability on two levels. It is essential for the system to be at least stable enough so that the oscillating overshoots do not increase in amplitude. This is referred to as *critical*, or *absolute*, *stability*. Beyond this, it is always preferable that the overshoots be as

small as possible. This is referred to as *relative stability*, and it is considered one of the performance criteria of the design.

Another way of looking at potentially critical instability is to consider from the start that system behavior will generally be cyclical in nature; in fact there may be some natural frequency at which the system tends to cycle from a high point to a low point and back again, with a corresponding, more or less constant, period. An extreme example of a critically unstable system is one in which the total time lag is approximately half of this period because this would mean that any attempts at corrections would be 180° out of phase with the actual changes. Therefore our efforts to implement negative feedback would actually result in positive feedback. The outcome would be that even a very small initial error would result in errors of continuously increasing amplitudes.

We now consider the magnitude of the corrective action as well as its sign. If we interpret negative feedback in the most literal and obvious way, then we assume that the magnitude of the corrective action is proportional to the magnitude of error. This is called *proportional control*. Let us refer to this proportion constant as the *gain*. Depending on the application, the gain may actually be adjusted by means of a separate amplifying device or even by something in the plant itself rather than within the control device, but nonetheless it can always be assumed to play a role in adjusting the system control.

If we decrease the gain, then the corrective action is less extreme for an error of the same size. This reduces the problem of instability because it gives the system more time to respond to changes before building up a large cumulative corrective action; thus the degree of overshoot is smaller. Therefore in a system with proportional control, the most straightforward way of improving stability is to decrease the gain. In a manually controlled system this may be expressed as having a lighter hand on the controls.

Although lowered gain in a proportionally controlled system improves stability, it is not necessarily the perfect solution to the overall control problem because lowered gains incur costs in terms of other performance criteria—*responsiveness* and *accuracy*.

A useful and specific way to consider responsiveness is to suppose that an error of a particularly large magnitude has been introduced into the system, and then measure how long it would take the system to return to a state of near-zero error. Once again, in practice this might be done by the reverse technique of suddenly changing the desired value setting. Clearly if we lower the gain, then we decrease responsiveness. It is also clear that in most control problems, we do not want large errors to stand uncorrected for long periods because that would defeat the purpose of control, and this implies that responsiveness is nearly as important as relative stability in control design.

In this context, accuracy is concerned with the system behavior in a nearly steady-state condition. In many applications it is undesirable to allow even a small

error to go uncorrected for long periods of time. A proportional control device with small or moderate gain will use very little force in attempting to correct a small error that persists with more or less constant magnitude and sign, so that the corrective force may be too weak to ever correct this error. Clearly the lower the gain, the more acute this problem becomes, and in extreme cases, the control device may not be able to adjust even for errors of medium magnitude persisting indefinitely. Thus some priority in the design of any control system must also be given to accuracy.

Often the way accuracy is measured in practice is to introduce a disturbance, for example suddenly raising the desired value significantly, then after a specified time, suddenly lowering it to its original level. One then waits for however long it takes for the system to settle back down to a steady state, and compares the new steady-state output value to the one before the disturbance was introduced. The two values may not be the same. Clearly, inaccuracy is not a problem we can work around simply by recalibrating our measurements.

To summarize what we have said so far, control design entails difficult trade-offs. The most straightforward way to interpret the concept of negative feedback is as proportional control, but that still leaves us with the problem of deciding the level at which the gain should be set. A low gain is desirable from the point of view of stability, but a low gain can detract from responsiveness and accuracy, which are also important. All of this makes it plain why a naive application of negative feedback is almost never sufficient to solve the problem of developing an automatic control device for a given application. Control design is clearly more difficult than one would first guess, and if it has evolved into anything at all like a science, then there must be more to it. In fact in most aspects of engineering, the real world challenges faced by control engineers may be much more involved than suggested by any conceptually straightforward discussion of basic control principles. For example one goal may be to design the control in such a way as to minimize the system's behavioral sensitivity to changes that will occur naturally in various parts of the system as it ages. Another specific problem may occur when a large scale, existing system must be modified to improve its performance. Since each piece of the original equipment will probably be expensive to replace, the engineer must determine how to make the best improvement in performance with the fewest changes in actual equipment. Any number of complicating factors can be involved in control design, but even in just its basic elements this is a difficult field to master.

2.3. Central Role of Models in Conventional Control

In conventional control methodology much of the effort to meet both the basic and the practical challenges takes the form of rigorous mathematical analysis. This typically begins with a control engineer attempting to develop a model that ideally

would accurately represent the behavior of the plant and yet be fairly simple. One thing we mean when we say “fairly simple” is that the model takes the form of linear differential equations. Although nonlinear models are not completely out of the question in modern conventional control theory, it is at least true that real world control design with such models is considered a very advanced topic. “Simple” usually means that the model involves only one controlling and one controlled variable. Multivariate models, like nonlinear ones, are acceptable in theory in conventional control, but are much more difficult in practice.

For these reasons one of the first skills emphasized in conventional control is mastering various techniques for simplifying models of physical systems. For example conventional control engineers are usually very practiced in the use of the Laplace transform because it is very useful in the simplification process and much later analysis is based on it as well. These engineers seek to combine the effects of various system elements into one transfer function representing the behavior of the entire system. This emphasis on model simplification, and particularly on model linearity, is important in conventional control for reasons we will discuss momentarily, but it also has costs which we will discuss later when we consider the limitations of conventional control theory.

The next step in conventional control design is to apply one or more analytical tools to the system model to determine various aspects of its behavior. More to the point, an engineer can attempt to determine in this way how to modify the design to attain the desired performance goals. Although these analytical tools evolved during the long history of control theory, and in some sense they are very sophisticated, they have the limitation that most of them were developed for models of a linear and straightforward type, preferably of first or second order. This is why all of the emphasis on model simplification is necessary in the conventional control methodology. Without discussing the details of any of these analytical tools or even claiming any expertise in their use, let us just mention that they have names such as root locus analysis, Nyquist diagrams, Bode plots, and Nichols charts. To say that these analytical tools normally require simple models does not mean that they are simple to learn to use properly. In fact, their mastery requires quite serious study even for those already well-experienced in advanced engineering mathematics. This is another problem we will discuss later as a drawback of conventional control.

Before exploring these problems, we should first discuss some other aspects of the conventional control methodology. We are particularly interested in two aspects that have important implications for fuzzy control in general, and these are particularly central to the discussions in later chapters. The first of these is the meaning of *dynamic compensation*, and the second relates to the methods customarily used to measure performance.

2.4. Dynamic Compensation

The significance of dynamic compensation can best be understood by considering the limitations of proportional control. We noted that P control is the most straightforward way to interpret the concept of negative feedback, but the limitations of this format are also most clear. In some applications it works reasonably well, but often P control leaves us with a particularly difficult compromise between relative stability and accuracy and responsiveness. Typically, the only major factor to manipulate in tuning a P controller is the overall gain. To lower the gain significantly will almost certainly decrease accuracy and responsiveness dramatically, but to raise it significantly will normally decrease relative stability. By using either the analytical techniques discussed or a combination of experience and intuition one can hope to obtain the best compromise possible on performance, but the result may not be altogether satisfactory.

Dynamic compensation consists of a set of concepts or structures that can be used to partially overcome this difficult situation of needing to trade one aspect of performance against another. These concepts have long been an important part of both the theory and practice of conventional control methods. It could even be argued that dynamic compensation is as important to conventional control as all of the techniques of modeling and analysis. To mention only one argument for this point, it is now sometimes recognized, as in (Van de Vugte, 1986), that the ready availability of proportional-plus-integral-plus-derivative (PID) controllers has to a significant degree replaced the need for rigorous analysis in practice.

Of the several forms of dynamic compensation used in practice, three are particularly important conceptually: proportional-plus-derivative (PD) control, proportional-plus-integral (PI) control, and PID control, and we will explain each of these conceptually.

In PD control a term proportional to the derivative of the error with respect to time is added to the term proportional to the present value of the error. As students learn in their first calculus course, the intuitive interpretation of the first derivative of any object with respect to time is the rate of change of that object at that instant in time. Thus it has a short-term anticipatory effect.

Due to this effect, an automatic control device based on PD control can partially overcome the problem of performance trade-offs: If, for example, the error is already moderately large and the rate of change in the error also has a large positive value, then we can anticipate in the short term that the error will become even worse, so a very forceful corrective action is desirable. However, if the error is significantly positive, but the rate of change in the error is significant in the negative direction, then this would be a situation in which the magnitude of the error is already decreasing, and therefore additional corrective action at this point should be small or nonexistent.

Looking at this in another way, one can think of the derivative term as having a damping effect on the oscillatory behavior of the controlled system: The derivative term decreases the severity of overshoots without the need to cut back on the proportional gain. Thus when compared with P control, PD control sometimes allow relative stability to be improved without damaging responsiveness or accuracy. In some conditions it can even improve all factors simultaneously.

The nature of the tuning process for PD control is only slightly more difficult to grasp intuitively than that for P control. Normally one can adjust both the proportional gain and the derivative gain, and intuitively we can think of this as the weights attached to the respective terms. If the derivative gain is decreased to near-zero or the proportional gain is so high as to overshadow the derivative term's effect, then we would largely forego the benefits of PD control because there would be no significant added damping effect.

If, on the other hand, we created the reverse situation of setting the proportional gain very low relative to the derivative gain, then our control system would act too strongly to prevent changes to nearly the exclusion of all else. To speak figuratively, the system would always seek to preserve the *status quo* almost regardless of whether it was good or bad, and that is the same as saying that responsiveness and accuracy would at times be exceptionally poor.

At first thought, it may seem that one could obtain nearly perfect control in all aspects of performance by increasing both gains in PD control to very high levels. However this is not true either because time lags in the system distort the derivative term in much the same way they distort the proportional term; therefore the anticipatory effect is far from perfect. The result of very high gains would be either instability or lack of responsiveness, perhaps even both problems at different times in the same system.

By conceptual arguments we have demonstrated informally what has elsewhere been shown both by mathematical proof and real-world experience. We have discussed why PD control has potential performance improvements over P control, but also why these advantages still require careful consideration of the parameters designed into the control. By this emphasis on intuitive presentation, it is also now clear that a human operator controlling a process manually has the capability to approximate PD control. It is quite easy to grasp, especially after some experience, that one should more forcefully correct against errors of increasing magnitude than against similarly sized errors of decreasing magnitude.

As one would expect from the name, in PI control an integral term is added to the proportional term. This is an integral of the error over time for some appropriate interval up to the present. The calculus student is usually told to interpret integral as area under the curve. This is an accurate explanation, but we hope the student will understand that a definite integral is a sort of continuous version of a summation over the period in question. As such, when comparing intervals of fixed length, the integral is also a measure of average behavior—a way of measuring what the

variable's overall tendency was during one period as compared to another period of equal length. It is in this light that we can best understand the significance of an integral term in dynamic compensation.

The most dramatic achievement of the integral term is to allow major improvements in steady-state accuracy with almost no negative impact on relative stability, and it is easiest to begin its interpretation by considering this aspect relative to the steady-state. We have already noted that lack of accuracy means that after the oscillations die out in the system and the controlled variable stops at a more or less steady value, this value still may vary slightly from the desired value. We said that this happens even in negative feedback systems because if we were to adjust the gain in P control high enough to correct even small errors, then it probably would be so high as to detract very seriously from stability, which would be unacceptable.

If there is an integral term, then the situation is quite different, because the integral of even a fairly small error becomes of significant magnitude if the error continues at the same (though small) value for a long period and if the interval of the integral is suitably long. Thus even with reasonably small gain, the integral term acts to push out small, steady errors, and this improvement in accuracy does not require the proportional gain to be raised at all. So far we have considered only steady-state behavior, but we already have a fairly convincing argument for why an integral term can improve accuracy without damaging stability, and thereby helps in quite a different way from a derivative term to alleviate part of the performance trade-off problem in feedback control.

Unfortunately it muddies the waters a bit conceptually to consider the behavior of PI control during transient conditions. It is tempting to take the easy way out by assuming that if the error is still oscillating, then positive and negative overshoots almost exactly cancel each other in the integral term, so that it has no effect on the dynamics. A moment's reflection however suggests that though there is likely to be some canceling effect, it is not realistic to assume in general that the integral of the error is very close to zero. Indeed we can imagine situations in which the integral term interferes to some degree with the dynamic behavior in respect to both responsiveness and stability. On the other hand, the overall effect need not be a major problem in this regard. First, although the integral term may not always benefit the transient performance, it will tend to help that performance on average more often than it harms it, or so our intuition tells us because the sign of the integral of the error is the same as the sign of the error itself more often than not. Furthermore we have already noted that the integral term can serve its function of steady-state accuracy improvement quite well with even a fairly small gain. Therefore in the dynamic situation it need not have a large effect relative to the proportional term.

The conceptual consideration of PI control is slightly more difficult to grasp than the conceptual discussions of P and PD control, but an experienced person can grasp the nature of PI control on an intuitive level. This is why it is often possible to tune a conventional automatic control device without using the rigorous analysis

normally associated with conventional control theory. Fortunately, it is much easier to demonstrate that an experienced manual operator approximates the integral term in at least a limited but practically significant way: If an operator notices that the system is no longer oscillating and that it remains at a small but continuing error, then the operator will surely consider it worthwhile to initiate an action to attempt to remove this error even though it is small. To speak figuratively once again, the expression “to remove a thorn in one’s side” is quite significant in describing human behavior so it is easy to understand why in a very basic sense the effect of PI control matches well with human intuition.

Naturally, PID control adds both an integral term and a derivative term to the proportional term. Having discussed the derivative term in PD control and the integral term in PI control, it seems unnecessary to now go into a detailed conceptual discussion of the three terms in PID control. Clearly the dynamic behavior of such a control device is more complex than anything we have discussed to this point, but it is still quite clear that in some cases the intuition of an experienced control technician may be nearly as good a guide in tuning a conventionally designed PID control device as the analytical techniques of an accomplished control engineer.

Furthermore since PID control has been shown by experience to be less sensitive in its performance to parameter settings, experienced intuition is often a fine replacement for rigorous analysis. Thus thanks in part to this added robustness offered by dynamic compensation, conventional control in practice is, to a significant degree, also a heuristic-based art.

Once again it is even easier to argue that the behavior of an experienced manual operator shows many of the same effects as PID control. First since the operator clearly bases much of his or her response on the currently sensed value of the error, there is a proportional term. Second the experienced operator naturally senses that he should respond forcefully to a significant error growing in magnitude but much less forcefully to a similarly sized error of shrinking magnitude so there is also the effect of a derivative term. Finally if the system settles at a more or less constant output and the output is even slightly in error, then the operator will attempt to remove the error so there is the most useful aspect of the integral term as well.

Though PID is now considered the norm in most modern conventional control applications, we do not wish to imply that it is always used. In some applications PD control, PI control, or even P control may perform well enough, with the advantage of possibly requiring less effort to design or tune. However, since ready-made PID controllers are commonly available now, and because their usually lower sensitivity to parameter settings often makes them easier to tune, it seems natural to use PID control in most cases. In fact many people now consider PID controller to be practically a synonym for conventional control device. However from the point of view of this book, that appellation is unfortunate in that it is misleading. We assert that dynamic compensation is by no means limited to conventional control.

To summarize, in its various forms, dynamic compensation is a more sophisticated and generally more practical way to interpret the concept of negative feedback than is simple P control. However though more sophisticated, it is still possible to interpret dynamic compensation conceptually. And, though more practical and less sensitive, some consideration must still be given to parameter tuning. With some difficulty we can understand why intuition plays a role in conventionally designed automatic control devices and with much less difficulty we can understand why human intuition can roughly approximate the best aspects of dynamic compensation in general.

2.5. Implementation of Conventional Control

This is an appropriate place to discuss briefly the implementation of conventional control. Naturally, conventional feedback control can be and is implemented in many forms ranging from simple, strictly mechanical devices up to huge systems implemented by use of mainframe computers with appropriate interfaces, and there are many other forms of implementation in between those extremes. For example (Van de Vugte, 1986) discusses in some detail a pneumatic implementation of PID control. However, clearly it seems most natural to think of conventionally designed control devices as being implemented by means of analog electronics. Most introductory courses on circuit design include discussions of methods for connecting resistors, capacitors, etc. in such ways as to make quite reasonable approximations of differentiation, integration, summation, etc., and these circuits are not at all complex.

One particularly effective way to implement feedback control, including dynamic compensation, is by the use of operational amplifiers, which are small, direct-current amplifiers with very high gains. They are also the most significant components in many analog computers. Many control engineering courses include lab sections in which, among other things, the students are required to build circuits of operational amplifiers, resistors, capacitors, and other components in such a way as to implement their own PID control devices. If the circuits include potentiometers, then these can be used to easily adjust the gains.

Though conventional control can be implemented by means of digital electronics, the fit does not seem altogether natural. Some of the conventional literature on the use of computers in control emphasizes process control, particularly in applications that are multivariate but involve more sequential control than feedback control. Often the advantage seen in the computer lies in the peripheral functions it can perform, such as data logging. Other parts of the literature do emphasize feedback control, but the discussion is suggestive of force-fitting analog methods to a digital implementation. Often these methods seem less than inspiring in the way they make use of digital computing power. Every good student of systems

science, applied mathematics, or engineering knows numerical methods and knows how to program a digital computer to approximate integration and differentiation, but it does not require much imagination to suspect that there is some better way to use these tools.

It is probably no coincidence that conventional control theory and analog electronics are two fields which fit together so naturally. This is because many of the important developments in conventional control theory occurred during the golden age of analog electronics. If low cost, compact, high-capability digital electronic devices like today's microprocessors had been available much earlier, then what we would now call conventional control might have evolved as something very different from what it is.

2.6. Performance Specification

To this point we have spoken of the performance criteria—stability, responsiveness, and accuracy—on only a conceptual level, with no thought to quantification, but in the engineering world quantifiable performance specification and measurement are always preferable. Therefore, it is quite natural that this should be an important aspect of any control theory. To a large degree performance measurement standards can be much the same whether one is using conventional control theory or a new, more heuristic-based methodology, but there are subtle differences in how these standards are used. Since conventional control emphasizes analytical methods, quantified performance specifications can play a central role in all aspects of the design process; this is not as true for control design, where heuristic knowledge is the basis of the design. Nonetheless, let us emphasize the similarities here. Usually, even in conventional control design, the specifications must be viewed flexibly, as subject to compromise.

It is the dynamic or transient aspects of performance that require the most careful thought in their specification: Accuracy requirements can be easily stated as just the maximum steady-state error allowable. One simply states that, no matter what disturbances may have occurred to this point, this is the maximum error the system is allowed to have when it does reach a steady state. But it is not quite so obvious how to measure relative stability and responsiveness.

The first step in defining transient aspects of performance is deciding what it is that we expect the system to respond to. In actual operation we typically wish the system to respond effectively to both various sources of external noise and also to possible resetting of the desired output value. Because it is often difficult to simulate the noise aspects, specifications normally begin with a test input signal—a time domain function that specifies how the desired value will be varied to test the response. Relating this back to the familiar example of an automobile cruise-control device; when we actually use the device we want it to respond appropriately to such

changing external factors as upgrades and downgrades, head winds and tail winds, engine conditions, and so on; and we also want it to perform well when we suddenly reset the desired speed. The common practice in control theory is largely to ignore during the design process the question of transient performance in the face of, say, wind changes, and to focus primarily on how well the device would perform if we reset the desired speed according to a certain function. Often in the broad literature on control, this point is not explicitly discussed, but the implication is given that performance in response to changing input is essentially the same as in response to external noise; therefore it is sufficient to test primarily with test input signals. This is one example of how the conventions of control theory seem more natural in the context of servomechanisms than they do in the context of ordinary regulatory applications.

In any case, one can offer at least a partial justification for this practice of testing with input signals and largely ignoring external sources of noise: The feedback control system is designed to perform on the basis of sensed error. The physical cause of the error is not fundamentally important so long as overall behavior in the face of error is acceptable.

The test input signals used in practice are typically very simple in form. In all cases let $r(t)$ be a function of the desired value of the controlled variable defined at time t . If one wishes to measure how well the system performs in the face of a sudden, dramatic change in conditions, then it is sufficient to introduce suddenly an error to the system by means of a step-input function of the form:

$$r(t) = \begin{cases} \alpha, & t < 0 \\ \beta, & t \geq 0 \end{cases}$$

for some appropriate values of α and β within the range of the controlled variable. Often α is assumed to be zero and β to be unity to form a unit-step function, which is sometimes useful in analysis. We state it here in this general form to make it clear that in practice it may be more useful to test response with other values of α and β .

If one wishes to place some emphasis on performance in the face of momentary disturbances, then it is most appropriate to use an impulse function as a test input. Impulse functions generally take the form:

$$r(t) = \begin{cases} \alpha, & t < 0 \\ \beta, & 0 \leq t \leq k \\ \alpha, & t > k \end{cases}$$

for some values of α and β in the range of the variable and for some appropriate time k . Again, sometimes for analytical purposes, it is useful to consider a unit-impulse function, which usually is taken to mean that $\alpha = 0$ and $\beta = 1/k$, but some other impulse function may be more appropriate in any given application.

In some applications designers may wish to investigate the system's performance in the face of changes that are gradual and continue for some time rather than abrupt changes. The most common function in these cases is a ramp-input function, which takes the form:

$$r(t) = \begin{cases} \alpha, & t < 0 \\ \beta t + \alpha, & t \geq 0 \end{cases}$$

for some α and β ; once again, typically $\alpha = 0$ and $\beta = 0$. But in any case, although this is the usual definition of ramp function, and it is useful in analysis, it is obviously not meant to be taken too literally in practice. Plainly, the change must stop before the system reaches its practical limits, and in most cases it must stop before reaching the limits of linear behavior. In conventional control theory, one sometimes considers quadratic test functions, usually called *parabolic test input functions*.

After deciding on one or more test input functions to use, the next step is to state how we expect the system to respond to each of them. There are a number of more or less standard formats for stating transient performance requirements, of which we will consider the most common three or four that would be normally used with a step function, particularly a unit-step function. Each of these formats provides a practical measure of relative stability, responsiveness, or a combination of the two. The assumption of a unit-step function as opposed to a more general step function simplifies the explanation. This assumption presents no difficulties in practice regardless of the actual application because we can always arbitrarily scale the controlled variable so that its initial value is zero and an appropriate new desired value is 1.0.

The various formats are most easily explained graphically. To illustrate this appropriately, we note first that in conventional control theory, it is often assumed that the linear, second-order differential equation 2.1 is a good approximation of most systems containing feedback control:

$$\frac{d^2y}{dt^2} + 2\zeta\omega\frac{dy}{dt} + \omega^2y = \omega^2r \quad (2.1)$$

where $r(t)$ indicates the desired value (here, the test input function) and $y(t)$ is the actual value of the controlled variable. The parameter ζ is called the damping ratio, and it usually takes a value between 0.1–1.0; ω is called the undamped natural frequency of the system. Clearly, this simple model is only an approximation of some aspects of real-world control system behavior. It does not illustrate the phenomenon of steady-state error, but it does give a reasonable indication of typical transient behavior. If we assume that the system was in a completely stable condition before $t = 0$, then we can use all zero values for the initial conditions. Figure 2.1 plots this equation for a unit-step function, with ζ arbitrarily set to 0.4

and ω to 1.0; various specification formats are indicated on this plot and will be explained below.

Percent overshoot (PO) is a common sense format for stating relative stability requirements. Expressed relative to our assumption of a unit-step function, this is simply

$$PO = \frac{\text{Maximum Value of } y(t) - 1.0}{1.0 - 0} \times 100\%$$

Quite literally this measure states by what percentage the response overshoots the mark on the first oscillation. Occasionally in practice a system is so well-damped that there will be no overshoot at all; in this case the PO naturally equals zero.

Rise time is a simple expression of responsiveness; its most straightforward expression is as t_r , in Fig. 2.1, the time the system takes to go from zero to unity for the first time. But sometimes t_{r2} , the time the system takes to go from 0.1 to 0.9 is the preferred format. One reason for using this second format is that sometimes a well-damped system may have no overshoot at all; thus it may never reach 1.0. Even if there is a slight overshoot, the system may be much slower in reaching 1.0 than in reaching 0.9. This implies that the first format may unreasonably penalize the design for good relative stability. The format used depends on the expectations

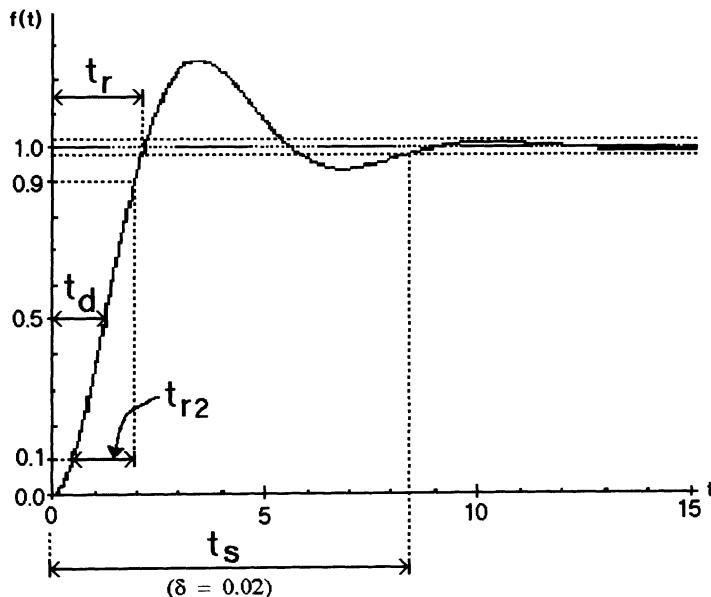


Figure 2.1. Response to a unit-step function, showing various specification times.

assumed for the application and the habits of the designers. Occasionally a delay time t_d , is also specified, indicating the time it takes the system to go from zero to the halfway mark.

Another important measurement is *settling time*, and this indicates a combined effect of relative stability and responsiveness. Settling time, which must be stated in terms of a δ value, usually 0.02 but sometimes 0.05, measures how long the system takes to reach a state where it remains between $1 - \delta$ and $1 + \delta$. This is indicated in Fig. 2.1 as t_s .

If other test input functions are also used, then various formats for performance specification have subtly different meanings. Figure 2.2 plots the response predicted by Eq. 2.1 to a unit-ramp function with the same parameters used in Figure 2.1 ($\zeta = 0.4$, $\omega = 1.0$). Because behavior in response to a ramp function is particularly sensitive to ζ , we also show the results when ζ is set at 0.1 but ω remains at 1.0. Figure 2.3 shows the predicted response to a unit impulse ($\beta = 1.0$, $k = 1.0$). Because the ω parameter is particularly significant in this case, we show results for both $\omega = 1.0$ and $\omega = 3.0$, in both cases with ζ at 0.4. Without going into details here, we must adjust the meanings of PO, rise time, and settling time slightly for them to be useful concepts with respect to these alternative input functions. Thus as an overall statement of requirements, we must specify one or more test input functions a maximum allowable PO, a maximum allowable rise time (of specified format), and a maximum allowable settling time for each; we must also state a maximum allowable steady-state error.

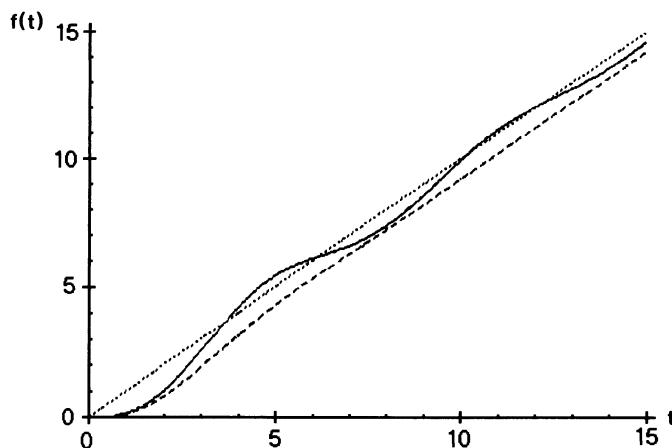


Figure 2.2. Responses to a unit-ramp function; (a): test function; (b) ----: response at $\zeta = 0.4$; (c) ——: response at $\zeta = 0.1$.

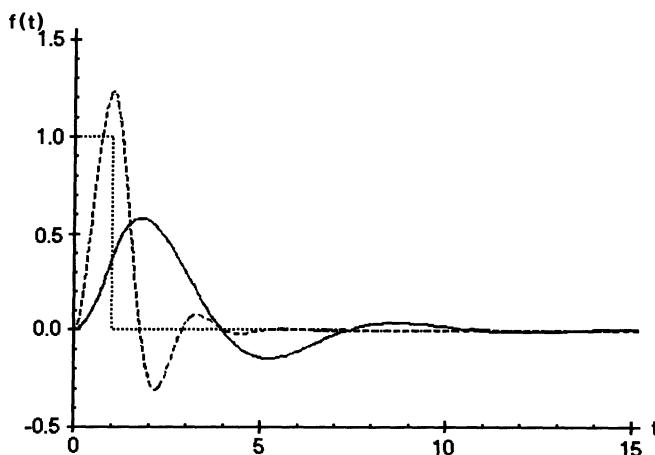


Figure 2.3. Responses to a unit-impulse function; (a): test function; (b) ——: response at $\omega = 1.0$; (c) ----: response at $\omega = 3.0$.

As we suggested early in Chapter 2, there may be other design requirements in addition to those whose specifications we discuss here; for example we may be required to design the control device so that it has low sensitivity to parameter changes. However basic performance criteria and these simple ways of specifying them are sufficient to illustrate the important connection between conventional and fuzzy control methodologies, which is one concern in this book. Unless based on an autotuning method, in some aspects of a heuristic-based design process, performance measurements cannot play so central a role as in an analytically based design process. Nonetheless the nature of the specifications and the final evaluation of the system can be very much the same in both cases. Thus fuzzy control can and does borrow much from conventional control in the area of performance specifications.

2.7. Practical Disadvantages of Conventional Control

This book emphasizes similarities between technologies and specifically considers the debt fuzzy control owes to the conventional theory of control. However there are some inherent disadvantages to be found in each of the technologies that we will discuss, and this includes fuzzy control as well as conventional control. Let us close this chapter with a short discussion of the weaknesses of the conventional control methodology. We explore some drawbacks of fuzzy control in Chapters 4–6.

Only by considering the disadvantages of the conventional theory can we clearly understand what fuzzy control theory is meant to accomplish. This has been a force driving fuzzy control research from its very beginnings.

Though the details of the first actual methodology for implementing fuzzy control did not come until the work of Mamdani *et al.* was published in 1974 or 1975, in some sense fuzzy control began in 1972 when Lotfi Zadeh published a brief article discussing his opinions on the state of control theory and suggesting that perhaps fuzzy set theory could be applied to control problems in some useful way. Zadeh noted that the field of control design had once been a very reasonable science; developing and applying theory where useful, but remaining faithful to the practical problems it was meant to solve. However, he complained that in his opinion, starting in approximately the late 1950s, control theory became increasingly obsessed with abstraction to the point that nearly every paper in the field contained at least one mathematical proof, but little improved in the practical application or performance of the systems developed. In later conversations Zadeh stated that this is one example of what he calls "the curse of respectability."

This opinion continues in the fuzzy set-applications-research community today, especially now that fuzzy control has very much proven itself. It seems that nearly every researcher who writes about fuzzy control has his own particular slant for explaining in precisely what ways he feels that fuzzy control is much more practical than the conventional methods. There must now be literally dozens of different ways in which conventional control is being criticized as impractical. We have no intention of going into all of these. We will stick only to the main points.

The first problem of conventional control is that it is a field which is quite difficult to master. Even if one tries to avoid Zadeh's reported tendency of the field to become overly abstract, and sticks instead to learning the practical application of the techniques, these are still difficult to learn to use outside of the simplified textbook problems. On a personal level this may seem more of an advantage than a disadvantage to some people. Those engineers who are already accomplished in the field may take pleasure in a sense of professional pride. Also some people simply enjoy applied mathematics, and the more challenging, the better. However when one considers the problem from the point of view of society overall, a methodology that is unnecessarily difficult is not to be preferred.

Despite the difficult nature of the conventional methods, there are considerable limitations on their applicability. If a method is based on rigorous analysis of a model of a plant, then obviously a condition for the use of that method is that the physics of the plant and its dynamic behavior must be understood well enough to develop the model. This is by no means always the case. Many of even the earliest fuzzy control applications were for large-scale plants where it would have long been desirable to automate the control, but in which it was impossible to do so previously because the plant dynamics were simply not understood well enough to create the type of model conventional control design requires. These included such plants as

cement kilns, trash-burning plants, and sewage treatment stations. This particular problem is typically more a matter of applied physics than applied mathematics.

A third disadvantage of conventional methods relates to the nature of models of physical systems. Any student of systems science knows that control theory is not unique in its preference for linear models over nonlinear ones. Most systems-related fields have this same preference, but the student also knows that there are costs associated with linear models. When we define a system on some real-world object and attempt to construct a model to represent that system, we should recognize that if we insist on the model being a linear one, then we will probably have to accept that it can be an accurate approximation only if limited to a certain range within the system's overall range of behavior. In some cases we will be interested in only a certain range anyway, and it may be possible to build the model in such a way that, though linear, it reflects the behavior accurately throughout that range of interest. In that case we find ourselves quite happy with the linear model though we should still be somewhat conscious of its limitations. On the other hand, in some applications we may be interested in the behavior over a very wide range of values of the variables, or in some systems the behavior may be so nonlinear that a linear model will not fit well even to a narrow range of values. In these cases a strictly linear model cannot work well. In some fields piecewise linear models (models based on a collection of linear functions to represent various ranges in the system behavior) are useful, but control theory makes almost no use of these.

This implies that there are two problems with conventional control's strong preference for linear models: first, some applications are unsuitable for a linear treatment and force the designer to use the less common and more difficult nonlinear methods. The other problem is that in some applications, a linear model can be made, but it is only accurate for a limited range of conditions. This may result in control devices that function well near the center of their ranges but behave poorly in extreme conditions. Indeed it is often reported on the basis of experience (see for example Matsumoto, 1972) that fuzzy control devices typically perform much better than conventionally designed control devices under extreme conditions.

CHAPTER 3

Fuzzy Control as Artificial Intelligence

Artificial intelligence (AI) encompasses a broad body of knowledge that is difficult to define precisely. Though it still has the aura of a dramatically young field, it is actually more like a war-torn veteran, which has experienced several setbacks and changes in focus over the past four or five decades. Although it is quite difficult to discuss such a subject in one chapter, and although some aspects of this subject are quite controversial, we nonetheless attempt to consider it here because it is important to an overall understanding of the nature of fuzzy control research and applications. In Chapter 3 we will present some points in a rather unconventional way. We attempt to show that fuzzy control indeed fits very well into the broader context of artificial intelligence, and that it is most fruitful to view it in that way, rather than as a subject that somehow stands apart.

3.1. Concerning Definitions of Artificial Intelligence

3.1.1. *High-Level Definitions*

Having said that it is both reasonable and useful to view fuzzy control as a branch of applied artificial intelligence, we should attempt to define what we mean by AI. Those familiar with the field know that unfortunately this is a very difficult thing to do with any precision.

Many definitions for artificial intelligence have been proposed, but none has been universally accepted. There are nearly as many different definitions of AI as there are books on the subject, and in many cases the differences are much more than trivial ones. To consider all of these proposed definitions would be a fascinating discussion in itself, but one we will avoid here. The difficulty is not simply a matter of expression because there is not even universal agreement amongst researchers on which specific areas should fall under the AI umbrella and which should not.

Actually, these difficulties are not altogether surprising when one considers that philosophers would probably be at a loss to define intelligence of the natural sort in a manner that is both very precise and universally agreeable. Despite this

difficulty of precise definition, the word “intelligence” is nonetheless useful, and so too is the term “artificial intelligence.”

A survey of several introductory textbooks on AI suggests that very many different approaches continue to be used for defining the scope of the field. We believe that the particular approach presented in this chapter is a reasonable one, but by no means does this imply that there is only one reasonable approach. Rather than becoming too argumentative in the body of this work, we will present only our main points here, and we refer those who are skeptical of our definition to Appendix A for further discussion.

Let us start from the premise that it is sometimes most appropriate to define a term by what it means by convention amongst the people who normally use it, rather than by what abstract philosophical argumentation might suggest that the term “should” mean. When experts in computing use the term AI to distinguish one application of computing devices from other types of applications, then what are the characteristics upon which this distinction is typically based? Regardless of whether this satisfies everyone’s concept of a valid definition, it is clearly useful to start with a discussion of those characteristics.

3.1.2. Knowledge Processing

Whereas most conventional uses of computers are described as information (or data) processing, most of applied AI is better described as knowledge processing. Clearly, there is a difference between what one means by information and knowledge, although it may be difficult to draw a precise line between the two. Actually, the Japanese term, *chishiki-shori*, seems to be more popular than its English equivalent, knowledge processing, but the concept is useful in any language.

We can consider two extreme examples from the field of accounting to illustrate the difference between information and knowledge processing. Given one predetermined list of assets and another of liabilities, if we train a novice bookkeeper to simply subtract the sum of the liabilities from the sum of the assets, then surely the actions of that bookkeeper are information processing, not knowledge processing. We could also write a rather simple and ordinary computer program to replace the bookkeeper, and such a use of computers would clearly fall within the realm of conventional electronic data processing (EDP). The collection of information about assets and liabilities arranged in the computer is called a simple database.

However if a qualified accountant is asked to advise a newly formed, small firm on several matters, then this would be an accounting problem of an altogether different level than the previous example. The accountant may be expected to advise on matters such as whether a single- or double-entry bookkeeping system would be more appropriate for this particular business, whether it would be worthwhile for this firm to computerize its bookkeeping system, and if so, then which software would be the best to use, how to best avoid audits by the tax authorities, and so on.

Naturally, this advice would be based on both the accountant's general experience and judgment and on specific information about the size and business of the firm in question. The activities of the accountant are knowledge processing. If we could find a way to replace or partially replace the accountant in this advice-giving role with a computer program, then clearly that program would not be considered conventional EDP; rather, it would be applied AI. If some aspects of the accountant's knowledge could be expressed in such a manner that the computer could manipulate them, then this would constitute a knowledge base (or KB), not a database, or at least not a database in the ordinary sense.

To summarize information processing is the manipulation of data according to precisely defined procedures; when automated, it is the conventional role of computers. Knowledge processing on the other hand involves subjective judgments on the basis of information; total or partial automation of such a process results in a somewhat less conventional role for computers, which may be called applied AI. This concept of subjective judgments as opposed to precise procedures leads toward our next approach at a definition.

3.1.3. Heuristics

Another trait that distinguishes artificial intelligence from most other computer applications is that AI tends to be based largely on heuristics, whereas most other uses of computers are based on algorithms. Naturally, we are now faced with defining the distinction between these two terms, and interestingly, there seem to be some subtle, yet important implications of the word "heuristic." Outside of the computer field, the word is used mainly in linguistics and the theory of education, where it indicates an approach whereby students are encouraged to develop their own working models as they go along. The *New Webster's Dictionary* defines heuristic as "a. Aiding or leading on toward discovery; following a teaching method that induces the student to make his own discoveries. n. A heuristic technique or discussion." The Japanese computing community tends to follow this definition more or less directly. If one refers to a Japanese language dictionary of information science terminology the definition of heuristic is likely to contain the phrase *hakkenteki na shuhou*, which we interpret as meaning "an exploratory technique" or "a discovery-oriented technique."

On the other hand, correctly or not, the computer science community in the English-speaking world seems to use the word heuristic in a subtly different way. Even though the phrase is quite informal or colloquial, many researchers use "a rule of thumb" as the most concise definition for what they mean by a heuristic. This manner of definition can be credited to either George Polya (in the context of mathematical problem solving) or Edward Feigenbaum (in the context of computing). In any case its acceptance in the AI community is now widespread. Perhaps this phrase does not translate well into some other languages. For example, in the

Japanese language maybe the word *meyasu* comes closest in meaning to rule of thumb, but does not seem to be so rich in connotations. On the other hand, it is sometimes claimed that the French phrase, *A vue de nez*, or “from the view of the nose” has almost the same meaning as the English rule of thumb.

Rule of thumb implies a mode of thinking that is imprecise but nonetheless useful in practice. It means that those experienced in the subject in question have found, largely on the basis of that experience, something is generally true or that something leads to a suitable solution most of the time.

In science and many other academic pursuits, we are warned not to depend on intuition, and certainly there are good reasons for this caution. However as one studies AI and attempts to make machines capable of approaching human problem solving capabilities, one is often reminded that intuition, especially the trained intuition of an experienced person plays a very important role in human reasoning and problem solving. Intuition is defined as the ability to reach correct conclusions or practical solutions while being wholly or partly unconscious of one's own reasoning processes. Students of AI also discover that heuristics play a central role in intuitive thinking. Thus compared to most sciences, AI research pays greater respect to some aspects of human reasoning, especially intuition and imprecise thinking.

In many areas of artificial intelligence, the heuristic, as rule of thumb, is viewed almost as an end in itself. In this sense, at least, it differs slightly from the meaning of the word as it is used in education, and so on, where it seems to imply a temporary expedient only.

By comparison, algorithms, which form the basis of most uses of computers other than AI, are precisely defined procedures for accomplishing a specific goal. Provided that appropriate input data are available, an algorithm can be expected to guarantee a suitable and precise solution. In the bookkeeping example, if we decide to subtract the sum of liabilities from the sum of assets, then we have described a precisely defined procedure. Provided that the data is correct, we are certain to arrive at a precise statement of owner equity. Although this is a particularly simple algorithm, it is a fair example of the nature of algorithms. Given some degree of programming skill, the process of automating the function of an algorithm in the form of a computer program written in the ordinary “procedural” type of programming language is relatively straightforward.

Since most formal education, particularly formal education in the sciences, teaches us to respect precise, certain, and objective answers based on solidly logical or solidly mathematical rigor, one's first reaction is to assume that the algorithm is altogether a more powerful concept than is the heuristic, but in practice this is not so. Algorithms are wonderful when they are available, but in order to develop an algorithmic approach, we must first completely analyze in a precise manner every relevant aspect of the problem in question. This limits us to only relatively straightforward and precisely measurable problems—a problem class that is nearly

nonexistent outside of mathematics, physics, and some aspects of engineering or accounting.

Traditional applied mathematics attempts to solve problems through algorithmic analyses of precisely stated models, and certainly this is sometimes effective, but sometimes it is not because the model does not accurately represent reality. For this reason, human reasoning seems to depend much more heavily on heuristics than on algorithms. This further implies that if we wish to extend the range of the types of problems that computers can practically assist the human race in solving, then a good way to start is to think of ways to make computers behave on the basis of heuristics as well as algorithms. It seems to us not too great a generalization to say that this is what AI research is all about, at least in the practical sense.

Actually, the distinction between algorithmic and heuristic-based computer uses is not quite so clear as perhaps we have made it seem. The detailed mechanisms of several approaches to AI are often described as algorithms, and one can even see the phrase, *heuristic algorithm*, which would seem to be an oxymoron if the two concepts really were completely distinct. One simple way to explain this is that a heuristic algorithm can be taken to mean a straightforward process whereby an answer of some sort is guaranteed (and thus is algorithmic), but whether that answer will actually solve the problem in question is not entirely certain and involves some informal reasoning (and thus is a heuristic). Much of artificial intelligence seems to be of this nature.

To summarize, it is quite reasonable to define AI in the following way; if a particular use of a computer is based, even in part, on the representation within the computer of some sort of heuristics, then that use is an example of applied AI. If a computer application is based entirely on programming the computer with methods that are strictly algorithmic in nature, then this is not AI. We feel that in a very vague field this dichotomy comes closest to providing a crisp definition of AI.

3.1.4. *Symbolic Processing*

Symbolic processing is another characteristic often associated with artificial intelligence, both by AI researchers and by critics of AI. These people speak of AI as emphasizing the manipulation of symbols in contrast to the supposedly primarily numerical calculations of other forms of computing. Certainly in the history of AI research, particularly in the United States, there has been a strong tradition of symbolic processing, and obviously there must be some significance to this tradition for it to be supported by so many esteemed researchers. However symbolic versus numerical processing is not nearly so clear of a dichotomy as it is often made to appear. There are in fact at least two major problems with treating symbolic processing as the distinguishing feature of AI.

First, although many writers on AI discuss symbolic processing as though we can interpret the term as meaning quite literally what it says, this most certainly

cannot be true if we are also to take it as a sufficient condition for a particular computer application to be considered as AI.

What exactly does it mean to process symbols in a computer? We recall a simple introductory course taught in the early 1970's, where the lecturer emphasized to the class from the start that the digital computer should be viewed not as a mere numerical calculator, but rather as a general purpose, automated, symbol-manipulation machine. This was not a class on artificial intelligence, but rather was on FORTRAN programming of all things. Throughout many years of working with computers since that time, on a wide variety of applications, that lecturer's comment has always seemed entirely valid. Indeed, some applications, such as the construction of compilers, strike us as being nothing but the manipulation of non-numerical symbols. Although compiler construction is an interesting topic, we have never heard it described as an AI application. If one did interpret symbolic processing in a literal and general way, it would apply to most applications of computers, not just AI.

In case what we are saying is not already absolutely clear, let us consider, as a particularly obvious example, a comparison of the following two statements:

Statement 1: "READ CARD-FILE AT END MOVE 'E' TO
CARD-SWITCH."

Statement 2: "New shoes sometimes squeak."

To interpret and to act appropriately upon the first statement is something that any COBOL compiler must be able to do, and this generally is not considered to be AI. For a computer to interpret and to act appropriately upon the second statement, on the other hand, would be an obvious example of artificial intelligence. It involves heuristics on several levels. But surely, if symbolic processing really could be taken literally as a sufficient condition, then the processing of either of these two types of statements would equally qualify as AI. The real story is that most people who talk about symbolic processing do not really want us to take them literally. They assume that we will read between the lines and understand what they mean is only certain specific types of symbolic processing such as list processing. LISP (the LISt Processing language) has been closely associated with much of the history of artificial intelligence in the U.S. It indeed appears as though some American computer scientists assume that the two phrases, "AI researcher" and "proficient programmer in the LISP language," should be viewed as almost synonymous. Let us leave aside all discussion as to the general soundness of this assumption, and simply grant, for the sake of argument, that perhaps the use of some specific types of symbolic manipulation can be viewed as a sufficient condition for a certain computer application to be viewed as AI. But what are these specific types precisely, and what are their boundaries?

Furthermore, the second problem we see is that this still leaves us with the question of whether one should view the use of these types of symbolic processing to be a necessary condition for AI. We believe one clearly cannot. Those structures now referred to as *neural nets* have enjoyed an on-and-off sort of popularity since the 1940s, and have generally been viewed to comprise a subfield within AI. Yet neural nets are primarily numerical in nature. Obviously then, it would be invalid, both now and in the historical sense, to say that all AI applications must relate primarily to symbolic processing of a non-numerical type.

To summarize it is difficult to accept that a straightforward interpretation of symbolic processing is meaningful as either a necessary or a sufficient condition for a given computer application to be AI. Furthermore, we suspect that many computer scientists should not be taken literally when they refer to symbolic versus numerical processing, though they seem to assume that they should be. None of this implies that research in the symbolic processing aspects of artificial intelligence is unimportant, only that symbolic processing is not nearly so clear and distinctive a characteristic of AI as is the concept of heuristics. What general statement can one make about the connection between symbolic processing and artificial intelligence? Those types of symbolic processing requiring the use of heuristics should be viewed as AI if they are automated within a computing device.

3.2. Emphasis on Domain-Specific Knowledge

Applications of artificial intelligence take a variety of forms, and they may at first appear to have little in common structurally. We believe it is possible to present a model that demonstrates a basic structural similarity amongst all or nearly all AI applications. In a later section, we will present one attempt at such a model. However, it is useful first to consider the nature of an important development in the history of AI research.

During the first few decades of the era of the electronic computer some researchers seem to have been quite optimistic of an early breakthrough that would allow computers to emulate human intelligence in a very general way, but over time the emphasis on universally intelligent behavior was gradually replaced with a more immediately practical approach that emphasized the emulation of some part of human reasoning in relation to only one specialized area of human knowledge at a time. This change began at least as early as the 1960s, and it reached widespread acceptance by the early 1980s. The most obvious result of this evolution was *expert systems*, which are often viewed as just one area of AI. From the late 1980s until now, AI has again experienced temporary setbacks and subtle changes in direction, but it continues to be clear that nearly all useful applications of AI to date have been specialized. This is particularly true if the concept of specialized knowledge is

interpreted in a reasonably liberal way to include all domain-specific knowledge and skills.

As one additional nuance to the trend, later applications emphasized that domain-specific knowledge and skills need not imply specialized human knowledge in the sense of professional or occupational specialization. For the period of a few years in the mid-1980s, many AI developers felt that expert systems was too narrow in its implications, and therefore they preferred to use *knowledge-based systems* or KBSSs.

3.3. Nature of Human Expertise

A moment's reflection reveals that most examples of productive human activity are applications of specific skills or specific areas of knowledge. Let us list just a small sampling of human capabilities that exhibit some practical sort of expertise.

- The ability of an experienced toy store clerk to advise parents about what types of toys are most suitable for a child of a certain age, with a certain interest.
- The ability of a commercial loan officer in a bank to select business proposals that are good risks from several loan applications.
- The ability of an investment adviser to develop an investment plan for maximizing return, minimizing risk, and legally minimizing tax liability, for a wealthy client, taking into account that client's specific circumstances.
- The ability of an auto mechanic to diagnose an engine malfunction quickly.
- The ability of a skilled manufacturing equipment operator to control a certain process. Similarly the ability to fly an airplane or drive a car.
- The ability to translate technical material from English into French.
- The ability to recognize handwritten text despite many variations in individual penmanship.

We could easily extend this list with dozens or hundreds of similar examples.

Some examples, particularly the last one in the list above, represent specific areas of human knowledge that are not normally considered to be expert knowledge in the strictly occupational sense, but as we indicated above, we wish to include these in our interpretation of "expertise" nonetheless. Furthermore, even this type of skill may need to reach a specialized level under the circumstances of some occupations. For example, all of us can perhaps read handwritten text, but few of us can match the efficiency of an experienced postal clerk in rapidly scanning an address while sorting mail.

As a further preliminary before describing a general model for applied AI, it is useful to present first a rough attempt at a general model of human skills and domain-specific knowledge, as shown in Fig. 3.1, as a basis for comparison. If we

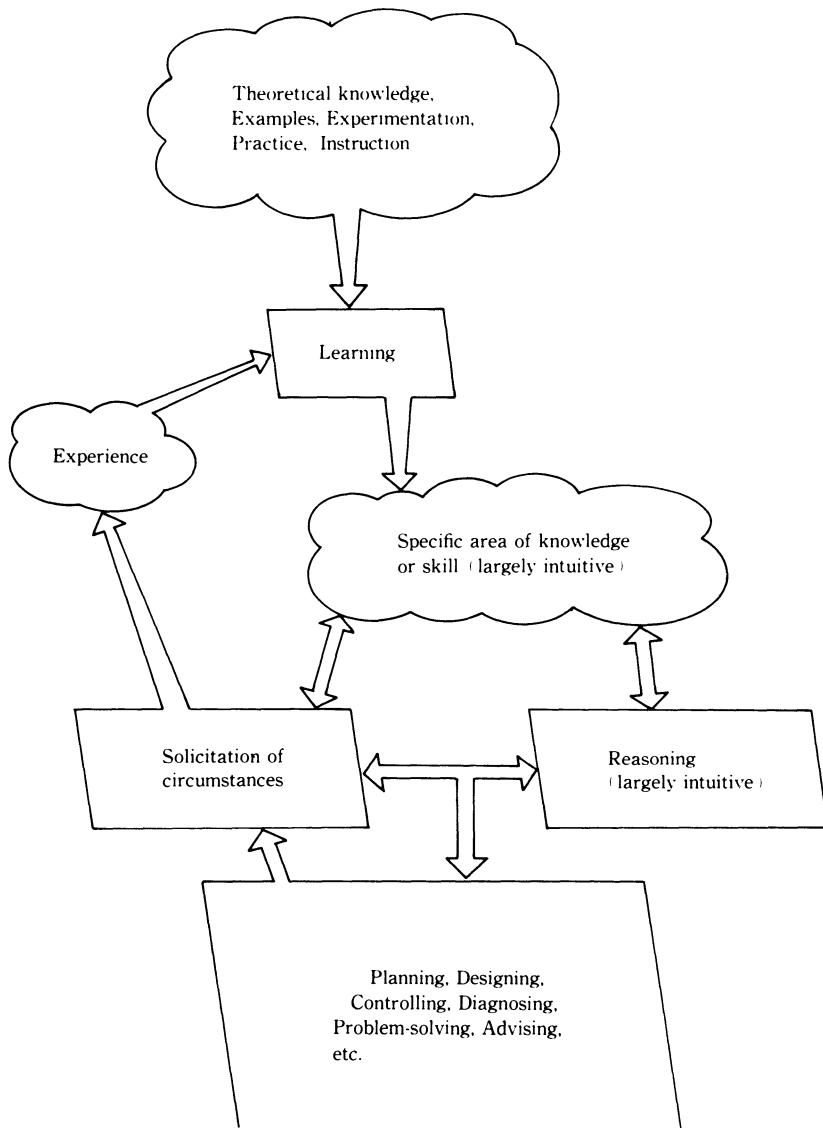


Figure 3.1. Nature of human expertise.

think of human reasoning in a specific area of knowledge or skill as knowledge processing, then the process can be divided into two major, fairly distinct phases. The first phase is learning or attaining some degree of mastery over the knowledge or skill in question, and the second is applying that knowledge to the specific situations which come up on a day-to-day basis.

There may be several sources for learning a skill or acquiring knowledge; the exact mix of sources depends on the nature of the area to be mastered and also on the habits of the individual who is learning. This mix may include several or all of the following; study of relevant theoretical knowledge with consideration of how to apply it, examination of case studies or other sorts of examples, experimentation, actual or simulated practice (possibly with guidance from a more experienced person), and specific instruction from a more experienced person. In addition, the learning process tends to be ongoing in that day-to-day experience becomes a further source of learning, a sort of feedback from phase two. In some cases one may list "common sense" as one source of knowledge, which really means that in the particular knowledge-domain in question there is considerable, relevant carry-over from a general life-experiences sort of knowledge.

The result of the learning phase is of course the knowledge or skill itself, but it is useful to consider the form of this expertise. Anyone who has learned to drive an automobile in busy traffic is well aware that it would be utterly impossible to perform certain tasks if one based all of one's actions on precise, conscious analysis of exact theoretical principles. Anyone who has learned to speak a foreign language fluently is also aware of this. We believe that in the mastery and application of most areas of knowledge and most skills, trained intuition plays a major role. Perhaps many people find it easy to accept this assertion relative to basic skills and everyday human thinking and reactions, but doubt that intuition is a significant factor in the professions or on any scientific matter.

Yet the experiences obtained from applications of AI to date demonstrate that intuition is a major factor regardless of the knowledge domain. Even some of the early expert systems were designed to diagnose certain types of diseases, and these were based on the knowledge of physicians specializing in the particular area in question. Although these expert systems often succeeded dramatically in the end, the greatest challenge in their development was that even though the doctors themselves naturally were very capable of making accurate diagnoses, they had great difficulty in consciously analyzing and explaining the thinking processes they used to arrive at these diagnoses. This topic of knowledge engineering is discussed in Section 3.4.

By intuition, we mean here not a naive state, but a trained intuition that results from a thorough mental digestion of knowledge relating to the area in question. As previously mentioned, it appears to be true that trained intuition generally takes the form of heuristics in the sense of rules of thumb.

The second major phase, the application of the skill or knowledge, involves ongoing interaction between a number of processes. If we consider the example of a toy store clerk and a parent who enters the store seeking advice about a toy purchase for his child, then exactly what processes are involved when the clerk gives advice? Alternatively, we can consider the actions of an industrial equipment operator when reacting to the moment-by-moment conditions of the work he is attempting to control. Despite the apparent differences in these two contexts, the processes of applying knowledge can be viewed in a remarkably similar way. The same is also true of many other contexts. One of the processes can be stated generally as that of ascertaining the state of the problem at hand. Thus the toy store clerk may ask the parent certain questions about the child's age and interests in an attempt to size up the situation. The equipment operator may observe various gauges as well as skillfully observing certain visual, auditory, and tactile clues directly from the work itself.

Clearly, a problem-state input process is essential in nearly every application of knowledge, but often this is not an independent step: It may be closely tied to the knowledge itself. When interacting with other humans, as in the case of the toy store clerk, it is important to ask the right questions in the right way and to interpret accurately patterns in the answers even in the presence of uncertain, incomplete, or extraneous information, all of which requires skill. Similarly, the operator needs to know which gauges to look at, and when; which clues to look, listen, or feel for; and again how to interpret and recognize patterns in the similarly vague, incomplete, or extraneous information. Indeed with some types of skills, such as handwriting recognition, the problem-state input process may be the predominant aspect of the skill itself.

Although the two processes are not entirely distinct, consider the problem-state input process is typically followed by a process of attempting to reason out a solution or an appropriate reaction to that state. Normally, the expert is only partially conscious of the reasoning itself, and thus largely on an intuitive level, the reasoning is based on three entities: information just obtained on the problem state, domain-specific knowledge, and general reasoning ability. As previously discussed, one implication of reasoning on a largely intuitive level is that much domain-specific knowledge is in the form of heuristics, that is, rules of thumb or informal thinking. What has not yet been discussed is the nature of general reasoning under similar circumstances.

Only on some occasions are most people required to consider their reasoning processes in an analytical, fully conscious manner: These occasions might include the study of formal logic, the presentation of a mathematical proof, and to some degree in debating or in justifying a major decision. Apparently, most people most of the time reason in a less formal and less conscious way. Some writers on creative problem solving, such as (Adams, 1986) and (Gause and Weinberg, 1982), emphasize the benefits of becoming more conscious of one's own thinking processes on

occasion in order to avoid the worst potential bad habits associated with reasoning in an undisciplined manner. But even these writers do not suggest that we ordinarily can, or should even try to, be completely conscious of our reasoning process. In fact they indirectly draw attention to the unconscious nature of these processes in ordinary human problem solving and decision making. And clearly, this mode of thinking does work much of the time.

Especially because it is something people are typically not fully conscious of, there is apparently much room for debate on the nature of general human reasoning processes. Some would claim that there are even major cultural influences, stating for example that one culture bases its reasoning on logic while another bases it on vagueness or other uncertainty. However, such arguments emphasize minor differences at the expense of major similarities. It is probably true that the scholarly traditions of one culture may more easily recognize and esteem the role of logic, while those of another culture will do this for the role of vagueness, but quite aside from what scholars wish to analyze, clearly the intuitive reasoning of all intelligent creatures must have some basis in logic and some basis in dealing with vagueness or other types of uncertainty and “approximate reasoning.”

Surely all human reasoning uses concepts of alternatives, of conditions, of implications, etc. that indicate a basic logical structure to the reasoning process. Indeed, it seems clear that often, without being conscious of it, we all use rather complex reasoning with several premises leading to several conclusions, which in turn lead to further conclusions.

On the other hand, it is a daily human experience to be forced to make decisions based on information that is incomplete, concepts that are not clearly defined, measurements that are imprecise, and outcomes that are partly unpredictable. That humans manage somehow to function under these circumstances implies that our reasoning processes must incorporate mechanisms for effectively dealing with vagueness and other forms of uncertainty. These mechanisms are necessary because of possibly vagueness in the information on the problem state and possible uncertainty in some parts of the domain-specific knowledge. When compared to the possible complexities of both the problem-state-input process and the reasoning process, typically the output process is quite straightforward. The reasoning process is expected to come up with a solution, and normally, once this is done it is a simple matter to express the solution, either verbally when interacting with others, or by control actions when operating a machine. However, in some contexts, particularly when interacting with other humans in an advisory role, it may be desirable to present not one solution, but a list of promising alternatives. For example in the end, the toy store clerk will probably only suggest certain possible choices, leaving the final decision to the parent. In this case it is useful to express the degree of certainty one feels that a particular alternative could be the best, along with possible advantages and disadvantages. Furthermore, it is sometimes necessary to be able to explain the reasons that a solution is suggested.

3.4. Synergistic Model of Applied Artificial Intelligence

Having presented a rough model for human knowledge and skills in the last section, it is now possible to demonstrate the similarities this has with nearly all practical types of applications of artificial intelligence described in the literature. The only assumptions made are that any given application of AI should attempt to deal with only one specific domain of knowledge at a time, and that the knowledge should be of a practical nature. These assumptions exclude almost nothing that has been useful to date. This general model of AI is illustrated in Fig. 3.2, and just as with the model of human expertise, it can be divided roughly into a learning phase and a knowledge application phase.

In developing an AI application, there are two approaches to the learning phase, and these are distinguished by how directly the computer is involved in the learning process itself. The two alternative approaches are illustrated by the two major branches in the upper part of Fig. 3.2. In practical applications of AI during most of the 1970s and 1980s, the left branch representing the knowledge-engineering (KE) approach was predominantly used. Although the KE approach continues to be important in many types of applications, there has been a dramatic shift in emphasis during the late 1980s and 1990s toward the right-hand branch, which represents the machine-learning approach. This shift has been noticeable in nearly all contexts of AI applications.

Actually though, to avoid from the start any sense of exaggeration, even when machine learning is used, it is nearly always the source of only part of the knowledge about the domain. At the least, the human developing the system must supply a rough framework, and some performance criteria, either explicitly or implicitly. In fact, they may even program some parts of the system in an algorithmic fashion.

In either case, the end result of the learning phase is more or less the same. It is called a knowledge base or KB, and in some format or another, it represents all of the domain-specific knowledge the computer must manipulate in the given AI application. Just as with the domain-specific knowledge of the human expert, there may be several parts to this knowledge, probably including the following:

- How to ascertain the current problem state
 - What questions to ask or what attributes to look for
 - How to interpret or classify the inputs
- What rules of thumb or patterns to use in attempting to reason out a solution
- How to communicate back the solution or solutions.

Regardless of the format used, all or much of this knowledge is heuristic.

To be useful in an AI application, the knowledge base must be in a format that is readily manipulable within a computer. It is also helpful if the format is easily comprehensible to humans, and this is often the case, but not always. The type of

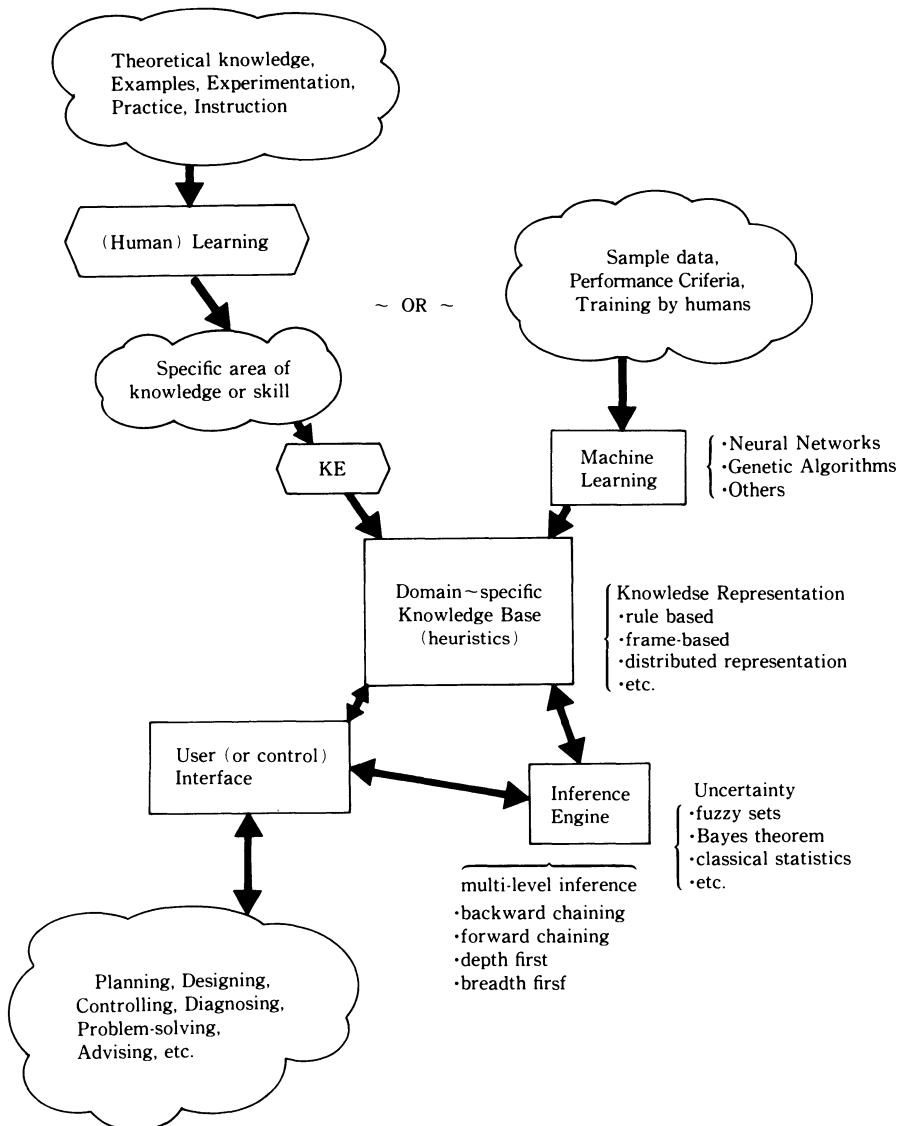


Figure 3.2. Nature of applied artificial intelligence.

the format that best represents a particular body of knowledge depends on the context, and to most AI researchers this is not at all an exact science: knowledge about knowledge representation tends to be heuristic.

In many applications, at least for the rules of thumb relating to the reasoning process itself, the most natural format is a rule-based (production-rule) system. Because an application of this format centers around a collection of simple implications, a very intuitive aspect of logic, this format has the advantage of being readily understandable by humans as well as manipulable by computers. Experience has shown that a typical AI application, even one relating to a type of expertise that is esteemed among humans, can be constructed around a collection of from a few dozen to a few hundred rules, each with the following format:

```
IF premise (or multiple premises joined by  
conjunctions)  
THEN consequence.
```

For convenience we attach a number to each rule. An example of a rule might be

```
RULE #17  
IF the child's age is 10 or above AND  
    the child enjoys constructing things AND  
    the child is interested in airplanes  
THEN the child would probably enjoy a model  
airplane kit.
```

A collection consisting of this rule and one or two hundred similar ones might be a way of expressing much of the knowledge in the area of toy store advice. However, in most cases even when the rule-based format does work well, it would be an oversimplification to say that a domain-specific knowledge base can be constructed strictly from IF-THEN rules of the above type. Typically, a knowledge base would also include patterns for how to ask the questions that must be asked, how to interpret or classify the responses or inputs, and how to express the final conclusions.

Because the knowledge is largely heuristic, there is uncertainty, so the sample rule we stated contains the word probably; many other words such as "maybe," "possibly", etc., also could have been used. Furthermore information about the premises may also be vague or otherwise uncertain. One child might be clearly interested in airplanes, and another child might be clearly completely uninterested in them, but one must also know how to consider the child that is neither clearly interested nor clearly uninterested. A subtly different problem may occur if, for example, an uncle was buying the present for a nephew or niece and simply did not know whether the child took an interest in airplanes. In some contexts a simplified approach that ignores uncertainty works fairly well in developing an AI application, but often some mechanism must be used to represent uncertainty.

Either instead of or in combination with rule-based representations, several other formats for the representation of heuristic knowledge may be used. These include such approaches as frame-based representations, distributed representations, fuzzy sets, heuristic evaluation functions, blackboard methods, and so on. It is interesting to investigate all of these, but for now it is sufficient simply to understand that a useful piece of knowledge can be represented in a fairly simple format that conceivably a computer could manipulate automatically, and that rule-based representation is an example of this.

As already shown in Fig. 3.2, the knowledge base is the result of the learning phase, and having discussed something of the nature of knowledge bases it is now easier to understand the processes of learning as they relate to artificial intelligence applications. Of the two major approaches, knowledge engineering is considered first, and then machine learning.

The knowledge-engineering approach is based on two assumptions: First that one or more human experts already know, or will soon learn, the skill or body of knowledge in question; and second that the expert(s), possibly with the assistance of another person called a *knowledge engineer*, can express that knowledge in a format manipulable by a computer. This is not so simple as it might seem, and particularly the second assumption is often a serious problem. But it is important to note that knowledge engineering is not so much a technical skill in the mechanical sense that the name might imply to some people. When it is difficult, the sources of the difficulty tend to be related to psychology and the limitations of the human capacity for expression of knowledge, especially knowledge that is normally dealt with in an intuitive, largely subconscious way.

During the 1980s knowledge engineering was almost synonymous with practical AI. The success of a large number of expert systems during that time, especially in North America and Europe, and the success of an almost equal number of fuzzy control applications, especially in Japan, indicate the significance of this approach. In the context of expert systems development with the tools available during the 1980s, knowledge engineering focused on knowledge acquisition, or more precisely on the extraction and re-expression of human knowledge. For many people knowledge engineering has this specific meaning.

Because humans do the learning, the original sources of the knowledge obviously can be all of the same ones considered for the model of human expertise presented in the previous section. This represents a richness of sources and a flexibility that will probably continue to be an advantage of the knowledge-engineering approach in some contexts for some time to come.

On the other hand, the main disadvantage of knowledge engineering is that in some contexts it requires a great deal of human effort, more than one would at first think necessary. In some situations, of course, the actual learning requires effort, but humans are quite accustomed to that process so more often the problem is in the expression of the knowledge.

Obviously, human effort implies expense, but in addition to that it seems to place some practical limitations on the complexity of systems that can be developed. First, in extreme cases, there are some types of knowledge that humans gain by experience and even take for granted, but which are extremely difficult for us to express accurately. We can pick the face of a friend out of a crowd of similar faces, but most of us would have difficulty in explaining in advance exactly how to do that. Second there are other types of knowledge that humans can express only in a very rough way, especially relative to what we mean by vague and otherwise uncertain concepts in our knowledge. Even for a person who has studied fuzzy set theory, probability theory, and other models for uncertainty, it may at times be difficult to determine the correct parameters to use with the model. Finally, even when uncertainty is not a special challenge, there is a practical limit to how much a knowledge engineer can express in any one application. For example in the case of rule-based systems, it is difficult to push the system beyond several hundred rules. Part of the reason is that even when the individual rules seem to be reasonably expressed, the manner in which these rules interact may become very strange when their number becomes large.

Primarily for these reasons, in recent years there has been consistently growing interest amongst researchers for machine-learning methods. To a large degree, machine learning, adaptive programming, dynamic filtering, and other related topics have been used for a long time in research typically considered as potential ends in themselves. However, we are speaking here of efforts to combine the advantages that machine learning can offer with the proven practicality of domain-specific AI approaches such as expert systems and fuzzy control. In the past, artificial intelligence researchers were often attracted by the apparent potential of machine learning to replace human effort in ordinary programming for algorithmic type information-processing applications. Although that goal has not been discarded, the practical potential of machine learning to replace human labor now seems to be even more promising relative to knowledge engineering for heuristic knowledge processing. One might say that machine learning is proving itself practical in a more dramatic way than was anticipated.

It is difficult to define machine learning precisely, but we can say that machine learning is any method whereby computers directly and automatically play a significant role in the learning process. This leaves us with the task of specifying what we mean by a significant role. For example some software designed for the development of expert systems has a feature for automatically generating simple deterministic rules from a specified type of database, but most researchers would not classify this function as machine learning. This feature involves only a relatively straightforward transformation of format, and somehow that is not considered significant enough to count as real machine learning.

A significant role in the learning process is the ability to generalize: to generate possibly imprecise but generally applicable rules from a collection of specific

instances, examples, or experiences, in other words the ability to produce rules of thumb. Thus, the connection between machine learning and AI is not simply the idea that intelligent creatures learn and therefore intelligent machines should too. Rather, if we say that the use of heuristic methods is a distinguishing characteristic of AI, then the connection between this characteristic and what we have just said about machine learning becomes very clear.

Other approaches to machine learning may prove practical, but at present most interest centers on the two approaches known as neural nets and *genetic algorithms*. Both of these are based on analogies to biological systems: neural nets on an organic level and genetic algorithms on an ecological one. A third approach currently receiving attention is related to genetic algorithms—*genetic programming*. Examples of these and other approaches can be found in the literature, but here it is useful to consider a few general points.

The sources and the results of knowledge in machine learning is an area subject to debate. One point of view is that present techniques are more limited than the scope of human learning. From this perspective, machine learning in its present methods only produces general working rules on the basis of sample data, performance criteria, or training by humans. In this view, computers are not yet capable of either directly applying or directly developing theoretical knowledge. However even viewed from this perspective, machine learning is very useful in terms of much of what we wish to accomplish in applied AI. Furthermore another point of view, perhaps best represented by Herbert Simon [see (Simon 1981) and elsewhere], believes that currently used methods do produce or use theoretical knowledge. Regardless of which view is accurate, machine learning has proven useful in many applications.

As with human expertise, the application of the knowledge incorporated in a knowledge base involves complex interaction between various processes. In a broad sense these processes are similar regardless of the nature of the knowledge domain, but there are many possible variations in the details. Much research has centered around issues relating to these variations.

Analogous to the human expert, if the contents of the knowledge base represent domain-specific knowledge, then there is still the need for some mechanism to ascertain information on the problem state, then output information about the conclusions. The nature of this mechanism depends on the broad category of the application. In the case of a system that is to communicate with people this would naturally be called the user interface, and it would typically refer to the knowledge base for direction on how to ask questions of the user, how to interpret the answers, and how to present the results. If, on the other hand, the purpose of the system is to control a device or process, then this mechanism is a control interface. Nonetheless it still typically refers to the knowledge base for direction on how to interpret inputs and produce outputs. In either situation there is an interactive connection between knowledge base and interface. Though the difference between a user interface and

a control interface is not entirely trivial, the two are very similar in terms of their fundamental functions.

In addition to the domain knowledge and the problem-state information, the third requirement for the human expert was general reasoning ability; the mechanism corresponding to this in an AI application is called the *inference engine*. This is a computer program that manipulates the knowledge format in a way that emulates human reasoning, including even rather complex reasoning. Obviously the inference engine must interact intensively with the knowledge base but also it generally interacts with the interface in many ways that are more or less direct. For example the inference engine often controls what questions are asked and when, since only it knows where information from answers fits into the reasoning process.

To this point the discussion has been quite straightforward and generally applicable, but several nontrivial details remain to be considered. First, as discussed above, is the matter of knowledge representation formats. We already considered that in many contexts, rule-based formats are a very natural way to represent heuristic knowledge, but that in other cases another format may be more natural. And in some cases where the knowledge is acquired by machine learning, the format may not even be readily comprehensible to humans. A further point to consider here is that clearly, whatever the format is, the inference engine must have been designed to manipulate it. Even on this point alone there is already an element of variety in the implementation of artificial intelligence applications.

The situation becomes obviously even more complex when one remembers that, especially as it relates to heuristic knowledge, the human reasoning process is more than simple application of *modus ponens*, and thus it is by no means obvious how best to emulate this reasoning process in a computer program or other automated device. There are two reasons for this. The first is that most human thinking in daily life is based not just on logic in the usual straightforward sense, but also on the ability to function in the presence of incomplete information, vaguely defined concepts, and uncertainty of outcomes. Although it may not always be necessary to emulate the ability to deal with uncertainty in every application, in general the theory used in practical AI applications should contain some provision for these capabilities.

The second reason that human reasoning is more complex than it may at first seem is that, quite aside from the uncertainty factors, it is often necessary for the human mind to follow many leveled chains of thought in the process of coming to a final conclusion. Again, it may not be necessary to emulate this in every AI application, but in many cases it will be, and this too requires careful thought in design.

Because the representation of uncertainty in reasoning processes is the subject of the next entire chapter we need not go into detail on it here. However, it is appropriate now to consider a few key points about how it fits into the overall scheme of applied AI.

It is appropriate now to consider a few key points about how representation of uncertainty relates to the overall scheme of AI. First, uncertainty representation is an issue in virtually all branches of applied AI. Certainly, there are specific applications for which it is possible largely to overlook uncertainty representation just as it is always possible that a given specific case will allow one to use a simplified version of any standard model, but this does not imply that the model can be universally simplified. Uncertainty is implicit in the concept of heuristics, and heuristics are the distinguishing feature of our view of artificial intelligence.

Second, there is, and apparently always has been, consensus in the AI research community on the need for uncertainty representation. In itself, this has not been a controversial subject. Claims to the contrary are simply not true as is obvious by a review of the history of expert systems, for example.

Third, on the other hand, there has been some controversy over what mechanisms one should use to represent properly the uncertainty. Unfortunately, this is a very difficult matter conceptually. In the past, researchers anxious to make progress with applications development have often adopted approaches based roughly on their own intuition, usually without claiming these to be necessarily the best mechanisms to use. This has been the case of confidence factors for example, as used in expert systems. However even those who have made a rigorous study of the subject often come to different conclusions, and it is no simple task to sort out who is right, as illustrated by the very high level of the debate between those supporting methods based on fuzzy set and fuzzy measure theories and those adhering to Bayesian or other traditional probabilistic approaches. Finally, whatever mechanism used to represent uncertainty will influence the design of all parts of the AI application. This would include the processes of the inference engine, the manner of ascertaining and interpreting input information, the manner of expressing or resolving uncertainty in results, and possibly the manner of expressing uncertainty about the domain knowledge itself.

The matter of inferences through multiple levels of implication, while less controversial than reasoning in the presence of uncertainty, nonetheless allows some variety and complexity. The ability to form a chain of reasoning through many levels of premise and consequence is sometimes regarded as a sign of high intelligence even in humans so it is not a simple matter to determine where to begin when creating a computer program to emulate this behavior.

When faced with a rule-based knowledge base with perhaps hundreds of rules, many having compound premises joined by conjunctions in the antecedent and many instances of one rule's consequent forming the antecedent or part of the premise of one or more other rules, it is not obvious how to proceed with automated inference. In practice two approaches have proven workable—*forward chaining* and *backward chaining*. Forward chaining involves taking all facts known at the present, then on the basis of repeated application of the rules reaching as many conclusions as possible. Backward chaining is more analogous to a mathematical

proof. It requires us at the start to define the conclusion we hope to establish as true or false, and then reviews the rules to determine whether or not (or in uncertain situations, to what degree) available facts (evidence) support the conclusion.

Whether forward or backward chaining is more appropriate again seems to depend on the nature of the application. Although at first thought forward chaining may seem the more intuitive approach, closer evaluation of many types of applications shows that backward chaining is actually more natural in most cases. As a result backward chaining has played a significant role in both the tools and applications of practical AI. For example the important logic programming language PROLOG functions most of the time in a backward chaining fashion. However there are still applications where forward chaining is more natural, and it is sometimes necessary to force a normally backward chaining tool to function in a forward chaining manner.

Other subtly different issues also pertain to multiple-level reasoning. Many problems in AI can be represented as hierarchical or inverted tree-like structures, and solving the problem becomes a matter of searching the tree for the best possible branch. In fact such texts on artificial intelligence, as (Winston, 1992) present tree search as a central issue in AI. In some contexts the tree is a natural representation, as in game trees as structures for representing alternatives in some types of board games. Actually it seems to be possible to represent almost any type of complex decision process as a tree or hierarchy, and this includes reasoning through chains of rules.

An important issue is how to search through the branches, sub-branches, twigs, and leaves of the tree most efficiently to obtain the best solution. Broadly speaking there are two approaches; depth-first search and breadth-first search. In the inverted tree structure, to search in a depth-first manner means to go down to one branch, to one of its sub-branches, and so on; exhausting all possibilities on one branch before moving to the next. A breadth-first search involves scanning first across all the main branches before considering any of the sub-branches below any of them. As with many other points, the best strategy in practice depends on the particular application.

Particularly in the case of expert systems development, there has been emphasis on using generic inference engines and interfaces. The inference engine and interface can be incorporated into commercially available AI-development-software packages, or shells, so that it is necessary to develop only the knowledge-base component when developing a new AI application. This idea has proven very practical, but it makes the major assumption that both the application in question and the user's point of view are compatible with the package developer's views on issues relating to the many details just discussed. Once again these are matters of knowledge representation formats, mechanisms for representing uncertainty, and mechanisms for multiple-level reasoning. Also if the knowledge base must interact

with an algorithmic part of the overall system, then the interface for this must also be considered.

In summary, in some ways, it seems relatively simple and quite intuitive to learn to use computers practically in this way, but in other ways, it seems that very much the opposite is true. Skeptics may quibble over whether or not it is appropriate to call it artificial intelligence, but the widespread use of expert systems, fuzzy control, and other similar applications proves beyond question that the manipulation of heuristic knowledge can be very useful if (for now at least) an application is developed by considering only one specific area of knowledge or skill at a time. On the other hand, those who feel nervous that artificial intelligence may allow computers to take over the world or make human thought redundant would seem to have very little need to worry, at least at present.

Machine learning has been viewed for decades to have various possible uses such as replacing the need for ordinary programming, directly emulating the workings of the human mind in a general way, etc. At present the greatest interest in machine learning seems to be in its use as a part of the process of developing and manipulating domain-specific knowledge formats in the computer. This certainly includes fuzzy control applications. Thus all parts of the structure illustrated in Fig. 3.2 are important to the future development of practical applications of AI. Having reviewed this role of machine learning, it is now possible to consider its specific mechanisms.

3.5. Machine Learning

As we discussed an important trend in AI research in applications development is an increasing emphasis on machine-learning methods as part of the overall structure or development process. These methods are practical as a partial replacement for the labor-intensive and complexity-constrained process of knowledge engineering when combined with expert systems, fuzzy control, heuristic search, and other categories of AI application approaches. Therefore we examine the most common machine-learning methods.

Neural net methods are probably the most common approaches in machine learning, so they receive most of our attention. We also mention genetic algorithm methods, which are receiving some attention especially with regard to fuzzy control. Many of the details are left to later chapters where we consider specifically how these approaches can be applied as part of a fuzzy control design process.

3.5.1. Neural Net Fundamentals

Although artificial neural nets, or neural networks, come in many varieties; they all function very roughly like some part of the nervous system (especially the

brain) in humans and other animals. We say very roughly because there is still much about the brain that neurologists do not know with certainty, so it is impossible to know how closely various artificial neural net structures approximate actual neurological structures. Also in creating efficient machine-learning mechanisms for practical applications, researchers may intentionally allow their structures and methods to vary somewhat from how they suspect the brain works.

To add another element of confusion to the subject, there are at least two quite different perspectives on the resurgence of interest in neural nets. Some researchers view neural nets as an alternative approach to computer hardware architectures. This view emphasizes the compatibility between neural nets and parallel (or “non-von Neumann”) computation to such a degree that they almost become synonyms. Clearly there is compatibility, and some potential applications require specialized parallel hardware because application run time would be too slow if implemented on conventional hardware. However the emphasis on parallel hardware makes it seem that all progress in neural net applications is tied to the development and availability of this special hardware, which fortunately is not true.

The other perspective treats neural nets simply as a mathematical model, a conceptual mechanism that displays interesting properties of learning, distributed representation, and inherent fault tolerance; all regardless of the medium of implementation, be it nerve cells, specialized electronic hardware, software for use on ordinary computers, or even pencil and paper. One rationale for this perspective is that modeling may provide further insight into purely scientific questions about how the nervous system works. There is however a practical advantage of this perspective as well: Some applications may be too slow for software implementation to be practical, but others may not. This is especially true in many cases for machine learning that is part of an applied AI systems, our focus here.

Many variations exist in the models and methods used in neural nets, but the general points are the same. As the name implies, these nets are based on a network structure, that is a directed graph with additional edges added to connect from and to the outside world. In some neural nets, especially those used as associative memories, the directed graph is a complete graph. But for typical applications with distinct inputs and outputs, the nodes are partitioned into two or more distinct layers with all directed edges pointing in one direction only. That is some edges indicate flow from a node in one layer to a node in the next. Others indicate flows from the outside world to various nodes in the first layer; and still others indicate flows from the various nodes in the last layer to the outside world. Figure 3.3 illustrates a typical small network of this multilayered type. Note: All signals flow from left to right in this diagram, though no arrows are drawn here on the diagonal lines.

In analogy to an actual nervous system, each node represents a nerve cell, and the flows indicated by the directed edges are signals. Typically at any time each signal can be indicated by a numerical value, often called an activation. In some

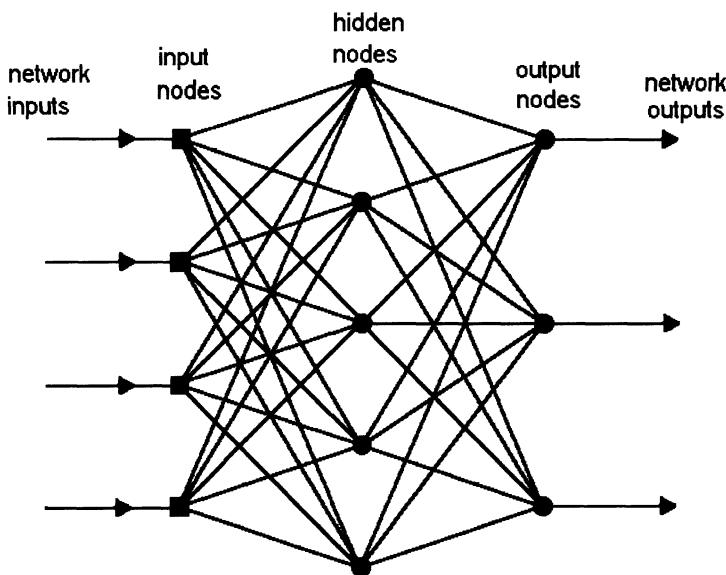


Figure 3.3. Structure of a typical layered neural net.

models the activation values are taken to be discrete, while in others they are from a continuous range in the real numbers.

Most nodes play active roles in propagating the activations through the network. Each node produces an output activation from all input activations it receives, using a given function determined by both the neural net model used and certain parametric values associated with the node. Even when several edges flow out of a given node, each has the same activation. The function used is usually thought of as the composition of two functions that are the basis function summarizing input states and an activation function that determines actual output activation from the value of the basis function.

In the model known as backpropagation networks (BPN), where activations assume values within a range of real numbers, a typical basis function for a given node i that receives activations x_j for $j = 1, 2, \dots, n$ as inputs is simply:

$$S_i = \sum_{j=1}^n w_{ij} x_j + \beta_i$$

where w_{ij} and β_i are parameters associated with the node, called the weights and the bias, respectively. (Often when actually programming this, it is most convenient

to use a special method for treating the bias as just another weight.) A typical activation function in a BPN model is the sigmoid or sigmoidal function:

$$y_i = \frac{1}{1 + e^{-s_i}}$$

where y_i is the resulting activation from node i . Several alternative function choices are possible. An exception to the preceding description of how a node functions is usually made for nodes in the first or input layer of the network. Each node retransmits the same signals it receives from the outside world; thus they simply serve as distribution points.

Turning again to the overall structure of the network, the number of nodes in the first layer naturally corresponds to the number of network inputs, and the number of nodes in the last layer equals the number of outputs desired from the network. Whether to add one or more additional layers placed between the input and output layers and how many nodes these intermediate or hidden layers should contain are more difficult questions. In fact the literature suggests that the best way to decide such questions is often through experimentation.

The most interesting point about most uses of neural nets is that the knowledge contained in the system takes the form of the values of the many weight and bias parameters. This is what we mean by *distributed representation*. In a typical approach, the weight and bias values are first set randomly, which represents a state of complete ignorance, then they are gradually adjusted by some appropriate algorithm, which represents a gradual process of machine learning.

To train a network through examples, a layered network with basis and activation functions as just described and some learning algorithm, such as the backpropagation method, may be very effective. We compile a collection of sample cases, where each sample case is a particular combination of typically expected input states and output states that we desire the system to produce when faced with such inputs. The network is then repetitively trained on the sample cases in this collection until there is acceptable agreement between the desired outputs and the actual outputs for each case. If the network is appropriately designed, then it should be able to generalize, that is, to fill in the blanks and to determine good outputs when faced with input states other than the specific ones in the training cases.

The detailed process of each step in this repetitive training slightly adjusts the values of the various weight and bias parameters according to the chosen learning algorithm. In the case of real-valued activations and no hidden layers for example, the algorithm can simply be a gradient search for the minimum error as a function of the weights and bias associated with the output nodes taken one at a time. Many variations are possible according to the actual neural net model used, and when hidden layers are used, the question of *blame assignment* becomes more complex.

We refer the reader to the literature and to Chapter 8 for further details on these matters.

There is one major disadvantage to distributed representation: It is often virtually impossible for the people who develop and use the application to interpret the results intuitively. This is particularly true when hidden layers are present in the network structure. Although this disadvantage is a significant one, it is often outweighed by the benefits of effective machine learning.

Concerning the effectiveness of artificial neural nets implemented (some would say “simulated”) in software on conventional digital computing hardware. It is clear that in the case of layered networks, if discrete time intervals are assumed, then sequential calculation does not influence the results obtained provided activations are calculated for one layer before moving to the next. Therefore in the case of neural methods applications using layered networks, the only possible drawback to software implementations is in efficiency of calculation. The calculation times required naturally depend very much on the complexity of network structure that is to be used, but in any case, the calculation time needed during the training period tends to be much greater than that needed for producing responses from the network once trained.

Therefore, the question of whether, for a given application, special purpose, massively parallel hardware is necessary, or, on the other hand, implementation in software on conventional hardware is sufficient, depends on a number of factors: How complex must the network structure be to discriminate sufficiently in its responses? Must the network learn in real time, or is it necessary for it only to respond in real time once trained? Must it be trained in each user’s special circumstances and each set of day-to-day conditions, or can it be trained for all users and all conditions? It is often assumed that artificial neural net applications necessarily imply special hardware, but depending on the answers to these questions, this is not necessarily true. Many applications, including some types related to general AI, work well without special hardware.

3.5.2. Neural Nets in Applied Artificial Intelligence

Neural nets developed for the types of practical applications discussed here rarely exist in isolation. Though the appeal of neural methods is that they can in many applications ease the effort of the knowledge-acquisition phase, it is rare for a complete, practical knowledge base to be composed solely of a trained neural net. Even when a neural net is the predominant element of a finished application, system inputs may be preprocessed, or at the very least the structure as designed by the developer is an implicit aspect of knowledge about the application. Furthermore to be of practical benefit, the neural net must normally be connected with appropriate human or control interfaces.

There are at least two major approaches to using neural methods in applied AI. Let us first define a typical AI application as one whose knowledge base consists primarily of two major components. The first component is a collection of production rules or frames representing the basic rules of thumb used in decision making; these comprise the system's judgment of how properly to react to a given situation. The second component is information about how to represent uncertainty in the functioning of the system. This may include a collection of fuzzy sets to represent system inputs and outputs as fuzzy variables, a collection of confidence factors, a set of prior and conditional probabilities associated with various frames for use with a Bayes' theorem approach in a frame-based system, or any number of other alternatives. Most expert system applications as well as nearly all fuzzy control applications fit this definition of typical.

Given this definition of a typical application, the two potential roles for neural net learning are

- To replace either totally or in part the first major component as just described, basic system heuristics with a trained neural net. Such a replacement by default also replaces the second major component as well because neural methods implicitly deal with vagueness and other uncertainty without the need for other mechanisms.
- To continue using rules, frames, etc., as before and also to continue to use fuzzy sets, etc., to represent uncertainty in the finished system; but to use neural methods to learn the best parameters to use in the chosen uncertainty mechanism, for example, to learn the optimal fuzzy set definitions to use with a predetermined set of basic heuristics.

There has been a great deal of interest in both of these approaches over the past few years, in both control and expert system applications.

Based on previous discussions, a strategy emerges quite naturally for applying machine learning from neural nets to real-world problems. When faced with a situation for which it seems advantageous to develop a system whereby a computer can emulate human decision-making, to some degree relative to a specific field of heuristic knowledge or a specific skill then applied artificial intelligence techniques are likely to be useful. To determine whether to include neural net methods in these techniques, it is useful to categorize the problem in one of the following classes.

3.5.2.1. Class 1. This class includes problems for which it is natural and relatively easy to express knowledge as if-then rules in frames or in some similar format and for which any vagueness associated with the knowledge can either be easily evaluated or ignored. In such a circumstance the disadvantages of using neural nets probably outweigh the advantages. It would probably be preferable to use ordinary knowledge-engineering techniques to express knowledge learned by humans in a convenient format for manipulation in the computer.

3.5.2.2. *Class 2.* A problem of this class is one for which it is fairly simple to express useful heuristic rules about how to make decisions in the problem area, but where there are significant factors of vagueness associated with the application of those rules, and it is difficult to express some aspects of that vagueness. For this problem class, the best strategy is probably to develop a rule-based AI application that uses fuzzy sets to express the vagueness and neural net methods to define the membership functions of those various fuzzy sets. (In Chapter 4 this is interpreted as defining a fuzzy relation for a linguistic variable.) As previously stated there has been much interest in this approach during the past few years.

3.5.2.3. *Class 3.* These are problems difficult for humans to express useful generalizations for decision-making on the area in question. In these circumstances it is difficult or impossible to use knowledge- engineering techniques to express suitable if-then rules, and some alternative is required. The alternative is usually some methodology for machine learning, and the best methodology available at present in many contexts is neural nets.

Thus problems in Classes 2 and 3 typically benefit from approaches using neural methods or some other mechanism of machine learning within the context of applied AI. This is generally true regardless of the specific application.

3.5.3. *Genetic Algorithms*

These algorithms offer quite a different alternative to neural nets as a method for incorporating machine learning into the development of practical AI systems, and they also serve as a widely applicable optimization technique. Knowledge of the fundamentals of genetic algorithms (GAs) is now quite widespread though perhaps not so much as with neural nets. The GA approach roughly emulates the process of natural selection within a population of biological organisms of the same species. There are at least two reasons for interest in genetic algorithms. One is to consider GA as just one aspect, though an important one, of the field known as artificial life, in which entire simplified biological systems are emulated for various purposes. Another reason is that GA methods offer a useful approach when investigating certain types of problems within any context, and this is the use of GA that most interests us here.

In this general problem-solving context, we develop a schema for representing the problem space as a simplified chromosome structure. These artificial chromosomes are strings of values representing the states of the various attributes of the problem space. Many GA researchers tend to use bit strings, but this is not actually required, although it certainly is desirable for each attribute to have only a finite number of possible states. Once the problem-representation schema is determined, an initial population of artificial organisms (or actually just the chromosomes representing those organisms) is generated. Typically this is done randomly.

The population is then subjected to a simulated process of reproduction and natural selection for dozens, hundreds, or thousands of generations. Just as in real-world biology, a natural selection process involves three factors—inheritance, variation, and selection. Although typically no distinctions are made between male and female in the artificial populations, reproduction is sexual in the sense that each offspring has two parents and inherits part of each one's chromosome. Although mutation sometimes plays a role as well, the major source of variation in sexually reproduced organisms tends to be sexual recombination. In the case of GA this can be simulated by some scheme for randomly choosing which parts the offspring inherits from each parent. There may be another random mechanism for simulating random mutation, but in many GA applications, this is found to be unnecessary.

In the general problem-solving context, the selection mechanism should obviously be based primarily on the objective function of the problem. However in practice it is often necessary to modify fitness criterion to be also based on the relative uniqueness of a given organism within the population. This is necessary to keep the population from becoming too uniform, and to avoid the worst cases of stalling at local maxima or minima within the problem solution space. Also as with real-world biological systems, it is often desirable to add an element of random chance to the selection process. That is, the more fit organism is not guaranteed to survive and produce offspring, and the less fit organism is not guaranteed to die; rather the more fit organism is simply given a higher probability of surviving than the less fit one.

One virtue of genetic algorithm methods is that they are so easy to grasp conceptually that this simple narrative description is all that is needed for a good programmer to begin experimenting with them. Those experienced in systems science or applied mathematics will also recognize that in the general problem context, GA methods are most easily applied to optimization problems with large but finite solution spaces and well-defined objective functions. The objective function need not take values in a continuous range, but its range must at least be quite finely graduated. There are many problem-solving contexts where the problem already fits this format, or it can fairly easily be transformed to fit.

The connection between GAs and general applied AI can be discussed on two levels. First in a very general way the overall nature of GA methods allows them to be described as AI regardless of the context, because they are clearly a form of heuristic search. Second in a much more specific way, GA methods can be applied with for example performance-based objective functions to handle Class 2 applications. This is one reason that researchers in fuzzy control methods have also become more interested in genetic algorithms over the past few years. They can be used to define fuzzy sets for system inputs and outputs.

3.6. Fuzzy Control and Artificial Intelligence

Fuzzy control should be classified as an applications area of artificial intelligence for the following reasons. We shall see later that fuzzy control is based even more on a general concept of heuristic reasoning than on the specific concepts of fuzzy sets. (We showed earlier that the existence or absence of heuristics seems the most precise way of determining which forms of automation are or are not AI.) Furthermore nearly all aspects of AI research deals with uncertainty in some manner; in fact reasoning under uncertainty is implicit in our definition of heuristic. Finally fuzzy control tends to follow the same trends as other application areas of AI, such as expert systems. The utility of viewing fuzzy control as an application area of AI will be discussed more fully after considering related details in later chapters.

CHAPTER 4

Some Relevant Aspects of Fuzzy Set Theory

Chapter 4 presents an applications-oriented view of the theory of fuzzy sets. It emphasizes but is not strictly limited to, those aspects most closely related to fuzzy control to give a concise explanation of both the mechanisms and their rationale. We cover some topics in greater depth than others, but we fairly thoroughly discuss basic fuzzy set theory and fuzzy logic, and we make some mention of fuzzy arithmetic and its applications. However we do not explicitly discuss fuzzy measures or the information-theoretical approach to these topics.

These latter topics are quite involved, and they have only indirect connections to the usual approaches to fuzzy control applications. Although most of what is discussed as fuzzy inference could also be called possibility theory, which is a specific case of fuzzy measure theory, this does not necessitate a general presentation of that topic. For in-depth coverage of specifically fuzzy measure theory, see (Wang and Klir, 1992). Klir (1990) and other recent works by the same author are good sources for exploring the relationships between information theory and fuzzy measures; on this same topic (Klir and Yuan, 1995) are most comprehensive. Several other sources, such as (Kandel, 1986), examine certain aspects quite deeply, and a few other works, such as (Jumarie, 1990) and (Losee, 1990), provide other insights into information theory that are in part related to fuzzy measure theory.

We present fairly unique views and approaches for certain of the topics—ways of arriving at the same results from quite different perspectives than standard ones in the literature. These are primarily related to the following points:

- Unlike the usual presentation of linguistic variables, discussed only as they relate to fuzzy numbers we present the same concept by means of fuzzy relations between the universe of the base variable space and the universe of a term set. In this way an element, a crisp set, or a fuzzy set in one universe can readily be associated with a fuzzy set in the other universe, in either direction.
- Fuzzy inference proper, as opposed to the overall workings of a fuzzy inference engine, are dealt with as a process functioning strictly in terms of various (input and output) linguistic variables themselves, that is in terms of their fuzzy linguistic values only. Thus, it is a process independent of all

base variables, and completely separable from fuzzification and defuzzification processes. This is much easier to understand than the usual approach where all processes are to a large degree lumped together. However assuming the min-max operators, the end results are the same, as Chapter 5 shows.

- Although a general approach is superior in most respects, it is easier to motivate fuzzy propositional logic in a specialized way—that is, what happens when we attempt to apply ordinary propositional logic operations when elementary propositions are related to questions about membership in fuzzy sets? This approach makes the connection between fuzzy set and fuzzy logic theories more concrete than just isomorphic structures, but unfortunately this approach is rarely seen in the literature, at least not in this format. The presentation here is only semirigorous at best.

In some cases examples are drawn from fuzzy control, but a complete description of fuzzy control methods is left to Chapter 5, so some practical aspects are intentionally incomplete at this stage. In that context, Chapter 5 also further clarifies differences between our perspective on fuzzy inference or approximate reasoning and the one commonly presented in the literature. A proof in Chapter 5 confirms that the end results of the two perspectives are the same under conventional operators.

4.1. Fuzzy Sets

To state concisely the most basic concepts of fuzzy set theory, it is sufficient to consider only the fairly simple way in which they form a generalization upon the corresponding concepts in conventional set theory. Thus, especially if it is assumed that most scholars are already familiar with the basic definitions and notation of the conventional theory, this discussion can proceed quickly. Although we do make this assumption, it is nonetheless useful to review and to clarify a few potentially confusing points about conventional set theory as a preliminary to the definitions relating to fuzzy sets.

Alternative approaches to the basic definitions of conventional set theory are possible, but the most common approach is to begin with an explicit statement of the context considered in a given discussion of sets. This is called a *universe of discourse* or *universal set*; it contains all elements that can conceivably be included in the discussion or problem at hand. We can then discuss various sets made up of some portion of the elements in the universal set. When this approach is used, all sets considered can be discussed as *subsets* of the universal set. This approach is essential to a consideration of fuzzy set definitions. In the notation used here, the universal set is labeled X , where x represents one element of X .

As reminders of a few basic considerations, in some discussions the universal set is infinite, such as the set of all real numbers, while in others the universal set is finite, such as the set of all known species of fish. Obviously, subsets of infinite sets can be either infinite or finite, whereas subsets of finite sets can only be finite. In any case, the null set can be considered as one of the subsets of the universal set. The null set, plainly, is a set that contains no elements at all, and thus has none of the elements of the universal set. Our notation designates the null set by \emptyset .

As a further preliminary before jumping into fuzzy set definitions, it is useful to consider that it is possible to define any conventional subset by means of a characteristic function defined on the universal set. Conventional set theory considers only what fuzzy set theorists know as *crisp*, or *binary*, sets. If set A is a crisp subset of universal set X , then each element x of X is either an element of A or not an element of A . Thus set A can be represented by *characteristic function* χ_A , defined as:

$$\chi_A(x) = \begin{cases} 1 & \text{where } x \in A \\ 0 & \text{where } x \notin A \end{cases} \quad (4.1)$$

By definition the range of a characteristic function consists of only the two values 0 and 1, that is χ_A is a mapping from X to the set $\{0,1\}$.

The common operations of conventional set theory can be stated quite simply in terms of these functions. The complement of set A , which is a set consisting of exactly those elements of X not contained in A , and denoted by \bar{A} , could conceivably be defined as follows:

$$\chi_{\bar{A}}(x) = \begin{cases} 1 & \text{where } \chi_A(x) = 0 \\ 0 & \text{where } \chi_A(x) = 1 \end{cases} \quad (4.2)$$

Similarly, one possible way of defining the concept of intersection for two crisp sets, A and B , could be

$$\chi_{A \cap B}(x) = \begin{cases} 1 & \text{where } \chi_A(x) = 1 \text{ AND } \chi_B(x) = 1 \\ 0 & \text{elsewhere} \end{cases} \quad (4.3)$$

The union of sets A and B could be defined as

$$\chi_{A \cup B}(x) = \begin{cases} 1 & \text{where } \chi_A(x) = 1 \text{ OR } \chi_B(x) = 1 \\ 0 & \text{elsewhere} \end{cases} \quad (4.4)$$

Although characteristic functions are not the most customary way of defining sets and operations thereon in conventional set theory, it is clear that this formulation is valid, and in fact quite convenient.

Conventional set theory is regarded as one of the philosophical foundations of mathematics, and it is often implicitly or explicitly assumed in various types of logical discussions; therefore it is obviously a philosophical tool that is often of

great value. However there is one rather troubling deficiency in conventional set theory as it relates to the ways humans tend actually reason about everyday phenomena: Conventional set theory assumes that every set worth discussing can be defined so that a given item is either absolutely in the set in question or absolutely not in that set, yet this assumption is often in conflict with how humans consider classifications of both concepts and tangible objects.

Examples of vague groupings of objects abound in nearly every field imaginable, even when discussed in a professional context. An ichthyologist may find it convenient at times to set aside the usual taxonomy of order, family, etc., and to discuss characteristics of fish according to fresh-water as opposed to salt-water species. Yet the scientist knows that some species of fish cannot be clearly classified in this way for various reasons. For example several species of salmon spend part of their life cycle in fresh water and part at sea, and many species of fish live, or can live, in estuaries, where it may be difficult to classify the brackish environment as salt water or fresh. Despite these and other difficulties in making this classification exact, we humans in our reasoning processes have nonetheless found it useful at times to classify fish in this way. There are multitudes of similar examples in nearly every field of human inquiry for which a method of classification is less than absolute and yet still conceptually useful.

Even when it is possible to quantify an attribute with a high degree of precision, we still often find it useful to reason in terms of imprecise classifications. It is conceivable that a standard pediatric procedure might be to treat an infant with a high fever and certain other symptoms in a certain way. It would be possible to state the conditions more precisely. Instead of "infant," we could say "a child up to the age of precisely one year," and instead of "high fever" we could say "a body temperature above 103 °F." Yet it often seems reasonable to us intentionally to avoid such precision. If a child's age is a few days or even a few months beyond one year, then we would probably trust the pediatrician to use his best judgment in deciding whether the symptoms still warranted the same treatment. Similarly, if the body temperature is 102.9°, or even 102.0°, the pediatrician may reasonably still favor the same treatment depending on the other symptoms. In most contexts, humans tend to view as arbitrary, rules with conditions that are overly precise. This attitude is actually quite appropriate because it is based on our knowledge of the heuristic nature of such rules.

Human reasoning patterns actually very often make use of vague or imprecise classifications, even of attributes that potentially could be quantified. Even to begin to emulate this reasoning in an automated device requires us to develop some rigorous methods for defining sets in a vague manner and for manipulating these usefully once defined. This is precisely the rationale for fuzzy sets, fuzzy numbers, linguistic variables, fuzzy inference, fuzzy arithmetic, and various related concepts. To begin with the matter of rigorously defining such sets, it is now possible to

present the straightforward generalization of conventional set theory to fuzzy set theory.

All that is necessary to allow for a more general definition of a set is to replace the characteristic functions used to represent sets above with a more general category of functions known as *membership functions*. A membership function is simply less restricted in its range than is a characteristic function. It is permitted to take values other than just 0 and 1. Although alternative interpretations are possible, in the standard format a membership function is permitted to take values in the continuous interval $[0,1]$, that is the real numbers from 0–1, inclusive.

If this standard interpretation is used and if μ_A is the membership function representing set A , then $\mu_A(x_1) = 0$ indicates that x_1 is absolutely not a member of A , and $\mu_A(x_2) = 1$ indicates that x_2 absolutely is a member of A , as was the case for χ_A . However unlike χ_A , μ_A is also permitted to take intermediate values, so that for example, $\mu_A(x_3)$ could conceivably equal, say, 0.63 or any other value between 0–1. This indicates that x_3 is neither absolutely in A nor absolutely out of A but is in a vague state in between these two extremes. Naturally the closer $\mu_A(x)$ is to 1, the more we say that x is something close to being absolutely a part of A , and the opposite for the closer it is to 0. Typically, $\mu_A(x) = 0.5$ could be taken to mean that the concept of including x in A is so vague that there are exactly as many arguments in favor of inclusion as against inclusion.

A set defined by a membership function in this way is a *fuzzy set*. As a special case of a fuzzy set, we consider a set whose membership function takes on no values other than 0 and 1. This is called a crisp set, which represents precisely the same concept as a set in the conventional theory of sets. Thus fuzzy set theory is a generalization upon the conventional theory, and in its basic structures, it is actually a very straightforward generalization.

Because the concept of a fuzzy set is so easy to misunderstand at first, we briefly emphasize a few points. First the universal set upon which a given discussion of sets is based must still be defined the same way in fuzzy set theory as it was in the conventional theory. There is never anything fuzzy about the universal set itself.

A more difficult point of confusion is that as a first impression on being exposed to the concept of a membership function, one is likely to assume that this is something very similar to a probability distribution, and thereby miss much of the true significance of fuzzy set theory. It is quite involved to consider this point from all angles, but a simple explanation is as follows. Probability is related to aspects of randomness in various phenomena, including natural and even totally mechanical phenomena. It has nothing whatsoever to do with vagueness of concept. Before a die is rolled, one does not know which number will face up, but provided that the die is rolled in the normal way on a flat surface, once it is rolled the result will be perfectly clear. This is an example of an experiment for which there is a significant element of randomness, but no aspect at all of vagueness of concept.

On the other hand, there are circumstances in which the opposite is true. If a child's temperature is 102.3° F, is this a high fever or merely a moderate one? The answer to such a question has no element of randomness in it at all, and is difficult to answer only because of the vagueness of concept as to exactly what one means by "high fever." This is not a problem of a random phenomenon, but it is a problem of finding a reasonable way to represent the vague manner in which humans tend to reason about various types of problems. Admittedly, situations do arise in everyday life in which there are both elements of randomness and elements of vagueness, and as an advanced topic it may be worthwhile to learn how to model problems of that type. But it is nonetheless clear that there is a fundamental difference between randomness and vagueness. Probability theory as it has developed over the centuries is primarily the tool used to model randomness. Fuzzy set theory, with its membership functions, as it has been applied over the past few decades, is regarded by many to be the best tool designed to represent vagueness of concept.

Another question that often arises is how to determine the membership function values to use in defining a given fuzzy set, when for example one is attempting to use fuzzy sets in an actual application? The simple answer for the time being is that in practice it often actually is a quite subjective process, but there is not really anything wrong with that. If a tool is designed for modeling vagueness in human reasoning processes, then it would seem quite unreasonable to expect that specific instances could always be defined in perfectly objective ways. One should keep in mind that the word "subjective" is quite different in meaning from the word "arbitrary." Also fortunately, experience with applications of fuzzy set principles has shown that there tends to be a natural robustness to the ways in which they can be used, and among other things, this implies that precise definitions are not a critical problem.

The next matter to consider is how to define such basic set operations as complement, intersection, and union for fuzzy sets. To be meaningful we wish these definitions to be generalizations on the corresponding concepts in the conventional theory of sets. That is, the way in which we defined complement, intersection, and union in (4.2)–(4.4) above in terms of characteristic functions should be the special cases of whatever way we do end up defining the operations on fuzzy sets in terms of their membership functions.

This too turns out to be quite an involved topic, and it will be discussed in greater detail later in the chapter in the context of fuzzy logic, but for now we can consider just one collection of definitions, the so called "classical set operators of fuzzy set theory." These are as follows.

$$\mu_{\bar{A}}(x) = 1 - \mu_A(x) \quad (4.5)$$

$$\mu_{A \cap B}(x) = \min[\mu_A(x), \mu_B(x)] \quad (4.6)$$

$$\mu_{A \cup B}(x) = \max[\mu_A(x), \mu_B(x)] \quad (4.7)$$

where $\max[a, b]$ indicates the maximum of a and b and $\min[a, b]$ indicates their minimum. A quick check confirms that Eqs. 4.2–4.4 are indeed special cases of Eqs. 4.5–4.7, respectively. However we reemphasize that Eqs. 4.5–4.7 are by no means the only generalized definitions conceivable.

Certain notational conventions have developed to represent various concepts in fuzzy set theory that cannot be easily represented in conventional forms of mathematical notation. Some of these conventions are slightly confusing in themselves, and furthermore are not used in a completely consistent manner throughout the literature. Therefore, it is worthwhile to explain the most basic of these conventions in the hopes of avoiding possible confusion.

One motivation for this new notation is the following. In conventional set theory, one can normally represent a given set without any mention of its characteristic function. We can write, for example,

$$A = \{\text{Tom, Dick, Harry}\}$$

to represent a particular subset of friends. Or in another context, one could write,

$$A = \{\dots, -5, -3, -1, 1, 3, 5, \dots\}$$

to represent the odd integers. Therefore, is it not natural that we should be able to represent a given fuzzy set without an explicit reference to its membership function?

The general notational format used for this purpose can be summarized as follows.

$$A = \sum_{x \in X} \mu_A(x)/x$$

Obviously here the solidus (/) does not mean “divided by” but rather “_ is the membership grade of element _.” This notation, though confusing at first, is perfectly functional in the case of small, finite sets, and it can be used with some level of abstraction for all types of fuzzy sets.

As a simple example, let us consider as our universal set a small group of friends:

$$X = \{\text{Tom, Dick, Harry, Mary, Sally}\}$$

We wish to describe a fuzzy set A_{th} which is the subset of these friends who are athletic. Suppose Tom and Sally are clearly athletic individuals because they are in excellent physical condition and participate regularly in a variety of sports. Let us further suppose that Dick is clearly not an athletic person because he is a “couch potato” and that Harry and Mary are each best described as somewhat athletic, Mary slightly more so than Harry.

One way to define set A_{th} would be by the membership function $\mu_{A_{th}}$, which can be shown in tabular form:

x	Tom	Dick	Harry	Mary	Sally
$\mu_{A_{th}}(x)$	1.0	0	0.4	0.6	1.0

However we can use the notational convention to represent A_{th} directly as:

$$A_{th} = 1.0/\text{Tom} + 0.4/\text{Harry} + 0.6/\text{Mary} + 1.0/\text{Sally}$$

It is unnecessary to include the term for Dick because we know that $\mu_{A_{th}}(\text{Dick}) = 0$.

As a further notational abstraction, a variation on this convention sometimes used for uncountable sets is to write the integral sign along with the solidus as:

$$A = \int_{x \in X} \mu_A(x)/x$$

This is a good place to introduce another notational issue, and a rather difficult issue it is. To this point we have used χ and μ to represent characteristic and membership functions, respectively. In each case the set name is indicated as a subscript of the Greek letter, as in μ_A to indicate the membership function that identifies fuzzy set A . This is an older form of notation, used in most early works on fuzzy sets. The older form, which we prefer, has many advantages: It helps keep clear the distinction between a set and the function representing that set and whether we are describing a characteristic function representing a crisp set or a membership function representing a fuzzy set. We suggest that students continue to use the older notation until they are thoroughly comfortable with their own understanding of those basic concepts.

Most researchers feel that the older form of notation is too cumbersome, so it is customary to use an abbreviated form of notation where the same symbol indicates both the fuzzy set itself and the membership function expressing that fuzzy set. Thus A may mean either the fuzzy set A or the membership function that identifies fuzzy set A . The reader is expected to distinguish on the basis of context whether the symbol means the one or the other. Presumably the same system is to be used to represent both a crisp set and its characteristic function.

In a work widely regarded as a classic on the subject of problem solving, (Polya 1945) includes notes on the nature of mathematical notation. To quote one of his many significant points: “Although it is forbidden to use the same symbol for different objects it is not forbidden to use different symbols for the same object” (p. 136). In our opinion the newer abbreviated format for representing membership functions violates the first part of Polya’s statement. However while we use the older notation with μ and χ throughout Chapter 4, we refer to the second part of the quote when we use the newer, abbreviated notation from Chapter 5 onward.

4.2. α -Cuts, the Extension Principle, and Other Concepts

There are many further extensions one can make to a discussion of fuzzy set theory both in the basic terminology and in the conceptual structures. Although nearly all of these extensions have some potential utility in practical applications, it is the purpose of this section to concentrate on only a few basic aspects that are clearly relevant to many types of applications including fuzzy control.

A simple but useful term is *normal* (or *normalized*) fuzzy set, which indicates a fuzzy set with membership function μ , for which:

$$\max_{x \in X} \mu(x) = 1$$

That is a normal fuzzy set is one whose membership function attains the value of 1 for at least one element of the universal set.

A meaningful way to generalize the definition of *cardinality* for a fuzzy subset of a finite universal set is simply as the sum of the membership grades; that is

$$|A| = \sum_{x \in X} \mu_A(x)$$

In the preceding example the cardinality of A_{th} is $3.0 = 1.0 + 0.4 + 0.6 + 1.0$.

An α -cut (or α -level set) is a crisp set defined for a given real value $\alpha \in [0,1]$ and for a given fuzzy set A ; it is written as A_α . This is typically defined as

$$A_\alpha = \{x \in X \mid \mu_A(x) \geq \alpha\}$$

However occasionally we refer instead to the strong α -cut, defined as:

$$A'_\alpha = \{x \in X \mid \mu_A(x) > \alpha\}$$

A simple example based on the fuzzy set A_{th} in the preceding section is $A_{th,0.5} = \{\text{Tom, Mary, Sally}\}$.

The concept of the α -cut is important precisely because it allows a given fuzzy set to be defined by means of an (infinite) collection of crisp sets and corresponding values. This is sometimes called the theorem of representation in the literature [see (Negoita and Ralescu, 1987)]. This assures us that there is at least one way to define the extension to fuzzy sets of any sort of structure that can be defined on crisp sets.

This leads us to the *extension principle*, which is very useful for making sure that we know precisely what we mean when we consider various extensions based on the concept of fuzzy sets. If f is a function defined on X with range Y and if A is a fuzzy subset of X , then we can consider the concept of $f(A)$, a new fuzzy subset of universal set Y , defined as follows:

$$f(A) = \sum_{x \in X} \mu_A(x)/f(x)$$

This definition can result in multiple terms for the same element of the range where the function is not one-to-one. In the standard interpretation, for each value of $f(x)$ we take the term with the maximum membership grade. In some contexts this form of the extension principle is useful in itself; however it is often more useful when combined directly with the concept of α -cuts.

To define the extension principle relative to α -cuts, we begin by extending the definition of a function to a crisp set. If C is a crisp subset of universal set X and if f is a function defined on X with range Y , then we can define a crisp set $D = f(C)$, a subset of Y , as follows:

$$D = f(C) = \{y \in Y \mid \exists x \in C \text{ such that } f(x) = y\}$$

We can further extend this definition to fuzzy sets by stating the extension principle as follows:

$$[f(A)]_\alpha = f(A_\alpha) \quad \text{for all } \alpha \in [0, 1] \quad (4.8)$$

Depending on how we wish to apply fuzzy set principles, Eq. 4.8 is useful for understanding the extension principle. Eq. 4.8 represents a particularly useful view of the extension principle in the context of fuzzy numbers as we will see in Section 4.3.

This is really no more difficult if we think of X as the Cartesian product of two or more sets and f as a multivariate function defined on that Cartesian product. For example, in the case that $X = X_1 \times X_2$, and $f(x_1, x_2)$ is defined for all $x_1 \in X_1$ and $x_2 \in X_2$ and has range Y , then for any crisp set $C \subseteq X$, we can define a crisp set $f(C) \subseteq Y$ as follows:

$$f(C) = \{y \in Y \mid \exists (x_1, x_2) \in C \text{ such that } f(x_1, x_2) = y\} \quad (4.9)$$

The application of the function can then be further extended to fuzzy sets by applying Eq. 4.8. Among other things this approach is useful in establishing fuzzy arithmetic.

If we discuss fuzzy subsets of the real numbers, then another useful concept is *convex fuzzy set*. This is simply a fuzzy set for which all of the α -cuts are convex sets. Continuing for the moment to think only one dimensionally, this implies that fuzzy set A , a fuzzy subset of the real numbers, is a convex fuzzy set if and only if A_α is either a single interval, a single point, or the empty set for each $\alpha \in [0, 1]$. Often in the literature the definition is presented in a more general context, but in a large number of applications, this interpretation of convex fuzzy sets is sufficient and intuitively appealing.

From the start we have viewed fuzzy sets as generalized subsets of the universal set, but it is useful to generalize the concept of subsethood further to discuss fuzzy subsets of sets that are themselves fuzzy; such a generalization follows, and seems to be a satisfactory definition of what one would wish subset to mean in this context. For two fuzzy sets A and B , subsets of universal set X , with A and B defined by membership functions μ_A and μ_B , respectively, $A \subseteq B$ is equivalent to:

$$\mu_A(x) \leq \mu_B(x) \quad \text{for all } x \in X$$

Note: For now we do not distinguish between proper and improper subsets, so all subset relations are denoted by the symbol \subseteq .

In conventional set theory a *power set* is a set that consists of all possible subsets of a given set. That is, it is a particular type of a set whose elements are themselves sets. Given the set $A = \{\text{Tom, Dick, Harry}\}$, the power set of A , denoted $\mathcal{P}(A)$ is the set $\{\emptyset, \{\text{Tom}\}, \{\text{Dick}\}, \{\text{Harry}\}, \{\text{Tom, Dick}\}, \{\text{Tom, Harry}\}, \{\text{Dick, Harry}\}, \{\text{Tom, Dick, Harry}\}\}$. Thus $\mathcal{P}(A)$ is a set consisting of eight sets. In conventional set theory, it is apparent that the power set of any finite set is also finite, in fact the cardinality of the power set is simply a function of the cardinality of the base set as follows:

$$|\mathcal{P}(A)| = 2^{|x|}$$

The concept of power set can be extended to the set of all fuzzy subsets of a given set. The fuzzy power set of a given set A is denoted as $\mathcal{F}(A)$. For any base set other than the empty set, the cardinality of the fuzzy power set is infinite. For example given $A = \{\text{Tom, Dick, Harry}\}$, for a fuzzy subset of A , the membership grade $\mu(\text{Tom})$ could conceivably take any value in the continuous interval $[0,1]$, so clearly the fuzzy power set is infinite even before we consider that $\mu(\text{Dick})$ and $\mu(\text{Harry})$ can each take values in the same range.

Finally we consider the concept of fuzzy relations. Many sources consider the properties of various special types of fuzzy relations and even fuzzy relational equations in great detail, but it suffices here to express the basic concept as simply as possible. A *crisp relation*, that is a relation of the ordinary type, can be viewed as a crisp subset of the Cartesian product of two or more sets. To describe a relation between two sets X and Y , we define a set $Z = X \times Y$, then label the relation R , which is a subset of Z . If the given relation associates a particular $x \in X$ with a particular $y \in Y$, then $(x, y) \in R$; if no such association is made, then $(x, y) \notin R$. Thus $R \subseteq Z$ can be thought of as a crisp set. If the concept is extended in such a way as to allow R to be defined as a fuzzy subset of Z , then this allows for the definition of a fuzzy relation. Conceptually this allows us to consider situations where for example it is not entirely clear whether $x \in X$ is associated with $y \in Y$. Fuzzy relations are generalizations on crisp relations in precisely the same way that fuzzy

sets are of crisp sets. In a fuzzy relation we are allowed to consider a degree of association.

4.3. Fuzzy Numbers

These are the building blocks of a large number of practical applications of fuzzy set theory to date. Fuzzy numbers are the obvious basis of fuzzy arithmetic, which in turn forms a basis for much of what can be called fuzzy operations research (OR) or fuzzy management science. In addition nearly all useful applications of linguistic variables and fuzzy inference are built around fuzzy sets that are fuzzy numbers, at least as linguistic variables are actually defined. This most certainly includes virtually every application of fuzzy control methods. For an applications-oriented person, it is only a small exaggeration to say that fuzzy numbers are the central theme of the bread-and-butter aspects of fuzzy set theory.

A fuzzy number is simply a fuzzy set defined on the real numbers ($X = \mathcal{R}$); the only additional conditions is that the fuzzy set must be both normal and convex. In Fig. 4.1 the fuzzy set represented in (a) is a fuzzy number, but the one in (b) is not because it is not normal; the one in (c) is not because it is not convex.

Membership functions of fuzzy numbers can take many forms, but in the interest of simplicity, there has been a tendency in applications to use *piecewise linear functions*. This emphasis on simplicity is quite reasonable because the fuzzy numbers are typically defined in a manner that is at least partially subjective anyway, and it is difficult to imagine the precise definition of quadratic or higher order functions on the basis of subjective representation of vague concepts. Furthermore, experience with the robust tendencies of the methods used in fuzzy set applications has suggested over the years that piecewise linear membership functions work well in practice. A piecewise linear membership function that is convex suggests what is known as a *trapezoidal fuzzy number* (TrFN), though it is quite

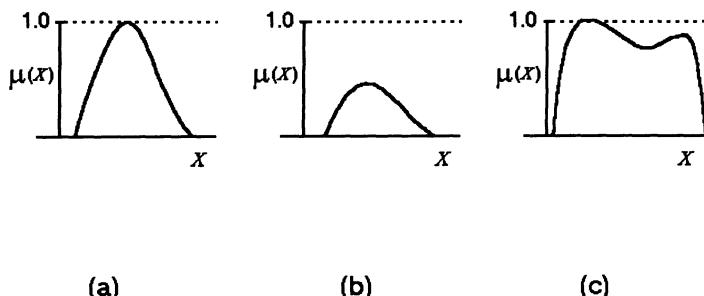


Figure 4.1. (a) A fuzzy number, (b) a fuzzy set that is not normal, (c) a fuzzy set that is not convex.

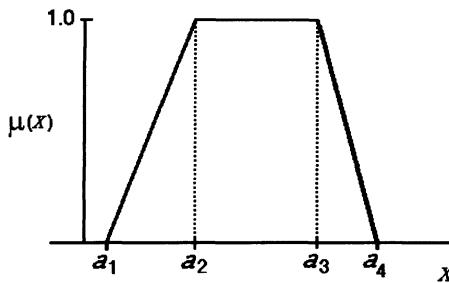


Figure 4.2. Trapezoidal fuzzy number.

common to further restrict the function to a format known as a *triangular fuzzy number* (TFN). Figure 4.2 illustrates the type of membership function used to define a trapezoidal fuzzy number, it is obvious that a TrFN can be identified by a quadruple (a_1, a_2, a_3, a_4) because the membership function can then be stated as follows:

$$\mu_{\text{TrFN}}(x) = \begin{cases} 0, & x \leq a_1 \\ (x - a_1)/(a_2 - a_1), & a_1 < x \leq a_2 \\ 1, & a_2 < x \leq a_3 \\ (a_4 - x)/(a_4 - a_3), & a_3 < x \leq a_4 \\ 0, & x > a_4 \end{cases}$$

The type of membership function used to define a triangular fuzzy number is illustrated in Fig. 4.3; a given TFN can be identified by a triple (a_1, a_2, a_3) . To consider the detailed form of the function, this can be written as follows:

$$\mu_{\text{TFN}}(x) = \begin{cases} 0, & x \leq a_1 \\ (x - a_1)/(a_2 - a_1), & a_1 < x \leq a_2 \\ (a_3 - x)/(a_3 - a_2), & a_2 < x \leq a_3 \\ 0, & x > a_3 \end{cases}$$

An alternative more compact form follows:

$$\mu_{\text{TFN}}(x) = \max\{0, \min[(x - a_1)/(a_2 - a_1), (a_3 - x)/(a_3 - a_2)]\}$$

The α -cuts of a TFN are the intervals $[\alpha a_2 + (1 - \alpha) a_1, \alpha a_2 + (1 - \alpha) a_3]$ for all $\alpha \in (0, 1]$, which in the specific case that $\alpha = 1$ simply means $\{a_2\}$. Thus most

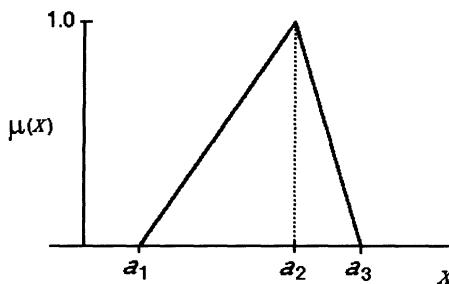


Figure 4.3. Triangular fuzzy number.

useful forms for expressing a TFN can be easily derived from the (a_1, a_2, a_3) form; this is a convenient method for identifying this type of fuzzy set.

Although fuzzy arithmetic is not directly part of typical applications of fuzzy control, it is useful to consider a few aspects of fuzzy arithmetic because it is central to many other applications and a useful demonstration of how to manipulate fuzzy numbers. In particular it offers a dramatic demonstration of the significance of the extension principle, which we use again later in our particular approach to fuzzy set-based approximate reasoning. Many sources, such as (Kaufmann and Gupta, 1985), (Zimmermann, 1987), and (Kaufmann and Gupta, 1988), provide detailed descriptions of fuzzy arithmetic that are both theoretically complete and aimed at applications.

A simple example considers a natural way of defining addition on TFNs. Equation 4.9 provides a way of precisely stating what we would reasonably mean by arithmetic addition on crisp sets defined on the reals, such as the α -cuts of fuzzy numbers. Making matters even more convenient, the α -cuts of fuzzy numbers are convex sets. For two TFNs $A = (a_1, a_2, a_3)$ and $B = (b_1, b_2, b_3)$, we find by considering all combinations of elements of the α -cuts of the sets (read intervals) that:

$$A_\alpha \boxplus B_\alpha = [\alpha(a_2 + b_2) + (1 - \alpha)(a_1 + b_1), (a_2 + b_2) + (1 - \alpha)(a_3 + b_3)]$$

Interpreting this in terms of Eq. 4.8, we find that adding TFNs A and B results in a third TFN with the intuitively appealing form:

$$A \oplus B = (a_1 + b_1, a_2 + b_2, a_3 + b_3)$$

Similarly we determine that a natural way to define subtraction of TFN $B = (b_1, b_2, b_3)$ from TFN $A = (a_1, a_2, a_3)$ also results in a TFN, which takes the following form:

$$A \ominus B = (a_1 - b_3, a_2 - b_2, a_3 - b_1)$$

Unfortunately a similar approach for defining the product or quotient of two TFNs does not, in general, result in a TFN. However it is not difficult to analyze these situations, especially if we restrict the problem further by working with TFNs defined on the positive reals. The results can be well approximated by TFNs that are also easy to calculate. These matters are well discussed in the literature. Furthermore similar analyses can be made for trapezoidal fuzzy numbers.

One difficulty of working in practical applications with fuzzy numbers is that it is not obvious how best to define a linear ordering on them. That is given a *sheaf* or collection of several fuzzy numbers, many of which overlap to some degree, it is not unambiguous to define a meaningful total ordering on them. This problem is significant in many applications in fuzzy OR, and also in fuzzy control and other applications of fuzzy inference, where it has indirect implications relative to defuzzification methods. Kaufmann and Gupta (1988) studied many approaches presented elsewhere in the literature and concluded that the best results were obtained from a complex and rather arbitrary system based on three different criteria. If it is so difficult to define in a not arbitrary way, a meaningful total ordering in the case of fuzzy numbers, there is at least as difficult a problem as ordering fuzzy sets that are not restricted to fuzzy numbers. This is just one suggestion of why researchers have been willing to settle for rather arbitrary, or at least somewhat *ad hoc*, defuzzification mechanisms in applications of fuzzy inference.

4.4. Linguistic Variables or Fuzzy Relations of Meaning

In the standard view the concept of *linguistic variables* has been central to fuzzy control and nearly all other applications of fuzzy inference, or more precisely of fuzzy set-based approximate reasoning. In its early years, fuzzy control was often called linguistic control. Yet in most of the introductory literature, linguistic variables are discussed only informally if at all. Furthermore, in our view even where fuzzy variables are considered formally, the formalism used is not the best one.

This section explains linguistic variables from the perspective of fuzzy relations used to relate their meanings between two different semantic contexts.

We began by explaining why linguistic variables are important. In many everyday situations one encounters an attribute which can be represented by the ordinary type of scalar (or possibly vector) variable, and can thereby be quantitatively measured with a high or moderately high degree of precision. Yet, it is likely to be the case that this precision makes the variable somewhat incompatible with intuitive human reasoning processes. In order to make direct use of this sort of variable, it is necessary to reason in terms of precise mathematical relations, and we humans rarely reason this way, except perhaps when we are given the leisure to

cautiously analyze certain types of problems, and even then only if we have been trained to think mathematically. In order to consider the attribute in a form more compatible with ordinary reasoning it is necessary to define another variable to represent it in a different way that intentionally discards the nonintuitive aspects of the precision, and yet accurately represents the conceptual structure whereby we naturally think of the attribute.

Thus, we are describing two types of variables to represent the same attribute. The first variable is of the ordinary type that takes numerical values, hereafter referred to as the base variable. The second variable is a closer approximation to the way one or more people intuitively reason about matters related to the attribute. This second type of variable we will call “the linguistic variable.” It takes values from the fuzzy power set of a small, specific set of linguistic terms.

The secret of the concept is that by defining a fuzzy relation between the linguistic context and the base variable space, we have a rigorous way of both interpreting and analyzing the linguistic variable.

A base variable can normally be thought to take values on the reals. Alternatively we can say that it takes values on a subset of the reals, first because there are typically some practical upper and lower limits beyond which it is inconceivable that the variable would range, and second because ordinarily the variable cannot be measured beyond a certain degree of precision. In practice we can switch between these two concepts, thinking of a base variable as taking values from either the reals or a finite subset of the reals, whichever concept seems more convenient at the moment, just so long as the fuzzy relation is defined appropriately.

The linguistic variable takes values from the fuzzy power set of a set of terms that one could be expected to use when informally describing a state of the attribute. This term set is usually a small one, meaning that it will normally have a cardinality somewhere from two to twelve. The particular set of terms depends on the attribute, so for a possible *linguistic term set* describing body temperature is {subnormal, normal, slight fever, moderate fever, high fever}.

However, in some application areas of fuzzy inference, particularly in fuzzy control, it is custom to use a fairly standard set of terms for the linguistic variable. This is usually based on the idea that every variable can be thought of as an error term or something similar, so that meaningful values center around zero. In this case a fairly standard way of writing the set of terms is {negative big, negative medium, negative small, (near) zero, positive small, positive medium, positive big} or the abbreviations of these, {NB, NM, NS, ZO, PS, PM, PB}. This set of terms is perhaps not particularly aesthetic, but it has been customary in some circles.

It is also necessary to express an association in meanings between the base variable space and the term set. This is not a matter of “giving meaning,” but rather of relating between two contexts of meanings. This is done by defining a fuzzy relation between the term and base variable sets. If we think of the base variable as taking values from the reals and if T is used to represent the term set, then one can

say that the point is to define a fuzzy relation on (T, \mathcal{R}) . Also, ordinarily this definition is constrained in the sense that the projection of the relation at any term must be a fuzzy number.

For example if the attribute is body temperature, $T = \{\text{subnormal, normal, slight fever, moderate fever, high fever}\}$ is the term set of the linguistic variable, and the Fahrenheit temperature on an oral thermometer is the base variable, then we can give a numerical elaboration upon the linguistic term set by relating it to the base variable with fuzzy relation $R : T \times \mathcal{R} \rightarrow [0,1]$, where $t \in T$ and $x \in \mathcal{R}$; it is specified as follows (see Fig. 4.4).

$$\mu_R(t, x) = \begin{cases} \max\{0, \min[1, (98.6 - x)/1.5]\}, & t = \text{subnormal} \\ \max\{0, \min[(x - 97.1)/1.5, (100 - x)/1.4]\}, & t = \text{normal} \\ \max\{0, \min[(x - 98.6)/1.4, (101.5 - x)/1.5]\}, & t = \text{slight_fever} \\ \max\{0, \min[(x - 100)/1.5, (103 - x)/1.5]\}, & t = \text{moderate_fever} \\ \max\{0, \min[(x - 101.5)/1.5, 1]\}, & t = \text{high_fever} \end{cases}$$

A few observations can be made about this particular fuzzy relation. First fuzzy sets associated with each linguistic term in the preceding fuzzy relation are not only fuzzy numbers but TrFNs. In fact considering the definition of TFN in a slightly creative way, these are each TFNs, including even the cases of subnormal and high fever. We use the (a_1, a_2, a_3) notation from the previous section to summarize the relation concisely as follows:

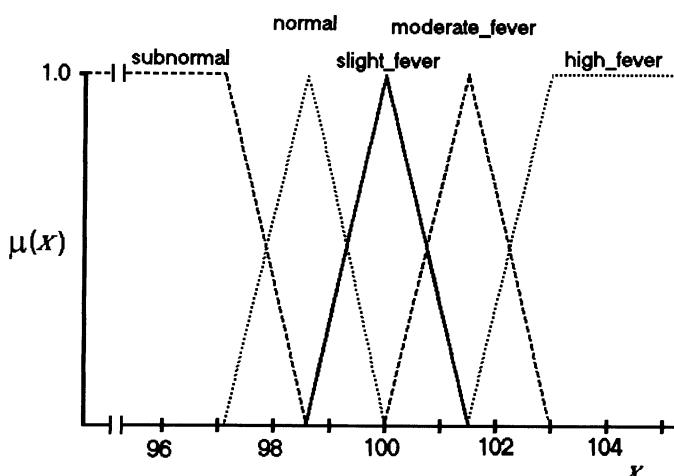


Figure 4.4. Example of a fuzzy relation for a linguistic variable representing body temperature.

- Subnormal: $(-\infty, 97.1, 98.6)$
- Normal: $(97.1, 98.6, 100)$
- Slight fever: $(98.6, 100, 101.5)$
- Moderate fever: $(100, 101.5, 103)$
- High fever: $(101.5, 103, \infty)$

In the early development of fuzzy control and fuzzy expert systems, it was common to use fuzzy numbers other than TFNs, but as explained in the previous section, it became customary to use TFNs in most applications.

One can also notice in the preceding fuzzy relation that

$$\sum_{t \in T} \mu_R(t, x) = 1 \quad \text{for all } x \in \mathcal{R}$$

That is the fuzzy subset of T that is the projection of the relation at any real number has a cardinality of 1. This is not an explicitly stated requirement of linguistic variable definitions, and some applications work well without this, but it has been customary historically.

The advantage of specifying a linguistic variable by a fuzzy relation between the term set and the real numbers is that it is the best way to suggest the two-way nature of the association between the base and linguistic variables. Just as it is true that each $t \in T$ is associated with a unique fuzzy subset of \mathcal{R} , it is also true that each $x \in \mathcal{R}$ is associated with a unique fuzzy subset of T . For example as already noted, the term `moderate_fever` $\in T$ is associated with the TFN $(100, 101.5, 103)$ or alternatively:

$$\int_{x \in \mathcal{R}} \max(0, \min[(x - 101.5)/1.5, (103 - x)/1.5])/x$$

This is a fuzzy subset of \mathcal{R} ; more precisely this is the projection of R at `moderate_fever`. While this is obvious in all thinking about linguistic variables, when viewed in terms of a fuzzy relation, it is just as obvious that if for example we consider $x = 100.6 \in \mathcal{R}$, this is associated with:

$$0.6/\text{slight_fever} + 0.4/\text{moderate_fever}$$

This is a fuzzy subset of T . Furthermore this two-way association can be applied in either direction starting not only from single elements but also from crisp or fuzzy sets once we define what is meant by *composition*.

There has been some interest in an application of fuzzy set principles to the study of linguistics in a more general way. One simple point often mentioned in the literature is the concept of *linguistic hedges*. For example if one finds a reasonable way to represent the phrase `middle-aged` by defining a fuzzy subset of the universal set of ages 0–100, then is there some stock way in which to modify the fuzzy set

to represent very middle aged or somewhat middle aged? If so, would it apply equally well to another attribute, for example for modifying a fuzzy set representing tall to represent very tall or somewhat tall? Although we occasionally read of fuzzy hedges and various related concepts applied in fuzzy control, it is not common to do so at present. The point is that although linguistic variables are a central concept in fuzzy control and other applications of fuzzy inference, we cannot assume that other fuzzy linguistic concepts are equally well used.

4.5. Fuzzy Inference (Logic) for Approximate Reasoning

The terms fuzzy inference and fuzzy logic have taken on rather distinct meanings in the literature, although it seems that some aspects of this distinction are rather arbitrary. This is perhaps particularly true in the English-language literature where under the heading of fuzzy inference one normally finds rather vague discussions of the generalized *modus ponens* or the compositional rule, whereas most specific discussion of generalizations of logical operators is reserved for the heading of fuzzy logic. In the earlier Japanese literature a similar distinction was made, but there recently seems to have been a movement away from that in Japan. Many Japanese researchers now use the Japanese language term for fuzzy inference or fuzzy reasoning to refer to all aspects relating to the design and application of what could be called fuzzy inference engines for fuzzy control, fuzzy expert systems, etc. Thus, the Japanese use of fuzzy reasoning has almost the same connotations as does approximate reasoning when used by English-speaking researchers in the fuzzy community. Although part of the motivation for this change actually may be the tendency in the Japanese national self-image to think of logic as a nasty word, it is nonetheless reasonable, at least from the applications point of view, to use fuzzy inference in this comprehensive way.

Many sources explain fuzzy inference concisely by discussing a manipulation of the rules and linguistic variable definitions so that the entire process can be presented as a direct outcome of one all-encompassing application of what is called the compositional rule of inference. This may be a useful approach for a presentation where brevity is all important, but it does very little for gaining a thorough understanding of the actual nature and rationale of a practical application of fuzzy inference. We avoid this approach, though we refer to the compositional rule in a less encompassing way, and we also consider composition as a part of defuzzification.

We begin with a global discussion of a fuzzy inference engine. Decision-making processes or system behaviors can be viewed in various ways. In all cases we assume multiple output values that must be determined on the basis of multiple inputs, as shown in Fig. 4.5a, where by way of example, we assume three inputs and two outputs. If the process is completely quantitative, algorithmic, and deter-

ministic, then it may be most convenient and natural to view the decision-making process or system behavior as a collection of multivariate functions, one function for each output variable, as shown in Fig. 4.5b.

On the other hand, if the sets of possible values for each input variable are finite, and not necessarily quantitative, but without any significant vagueness in the concepts of the values, then it may be convenient to express the process or behavior as a rule base with an accompanying inference engine of a nonfuzzy sort (see Fig. 4.5c). This is only a matter of convenience. Even under these circumstances it is possible to express the process in terms of functions, as in Fig. 4.5b, but the rule-based concept may be more natural and may be much easier to develop or analyze.

Finally, though this does not exhaust all possible circumstances, we consider the case for which all variables are quantifiable, but where it is very difficult to express the process or behavior in the form of precisely stated functions, yet it is quite easy to express it in vague terms. In this case the best way to develop the process or to analyze the behavior may be by means of a fuzzy inference engine, which uses not only a rule base, but also linguistic variables (one linguistic variable is associated with each input and output); see Fig. 4.5d. Again, it is possible to reinterpret the rule base as a collection of functions, as in Fig. 4.5b, but these functions would be defined for the term sets of linguistic variables, not directly on the reals. To investigate this approach in greater detail, one can follow the actual flow, considering the “fuzzification” step first, the fuzzy inference engine proper second, and finally the “defuzzification” step.

All linguistic variables are assumed to be defined in advance as part of the design or analysis of the system. Therefore because we presented the concept of linguistic variables in terms of fuzzy relations, the process of fuzzification is obvious. The value of each input base variable x_i is a real value (normally a scalar, though a vector is also possible if the base variable is defined in that way), and just as described in the previous section, the fuzzy relation associates with this $x_i \in \mathcal{R}$ a fuzzy subset of the term set T_{x_i} of the associated linguistic variable. Thus, the value of x'_i in each case will be a fuzzy set of the nature of the 0.6/slight_fever + 0.4/moderate_fever example we saw previously. If the more or less standard terms for fuzzy control are used, the x'_i value may be of the nature of 0.23/NM + 0.77/NS.

In the fuzzy inference engine itself, there are many details to consider. The nature of its inputs was just explained, and next we consider the nature of the rule base.

Although in actual practice in fuzzy control applications, a more compact format is sometimes practical, one can begin by considering the rule base to consist of between several dozen and several hundred rules, each of the same simple format that is used in rule-based expert systems or production rule systems as discussed in Chapter 4.

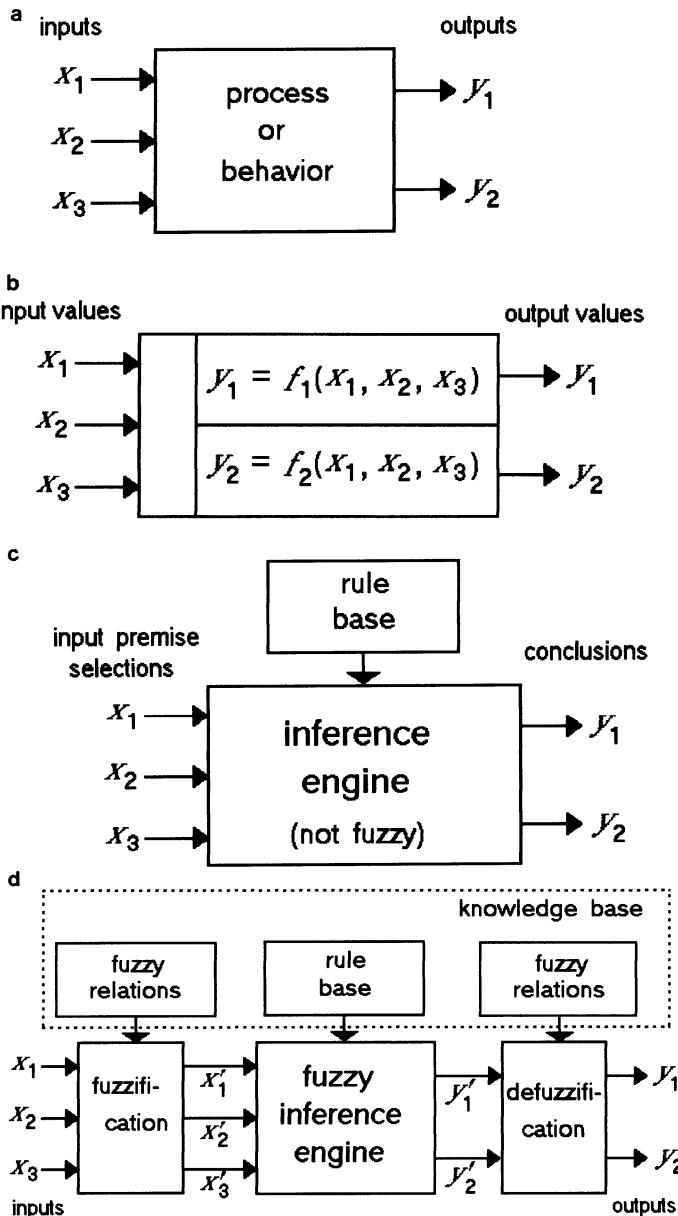


Figure 4.5. (a) Basic system structure, (b) system represented as multivariate functions, (c) simple production rule system, (d) fuzzy system.

RULE #k
 IF premise THEN consequence

As before the antecedent can be very simple, consisting of just one basic proposition, or it can be built of multiple basic propositions joined by the conjunction AND. Furthermore, if convenient, the premise can also make use of logical negation NOT. It is considered by some to be bad form to use disjunction OR within a rule, because it is always possible to avoid doing so by dividing a rule with OR into two or more rules without OR. However other researchers, particular in the fuzzy control area, are accustomed to using disjunction within rules. In any case disjunction is implicit in the overall rule base because the rules are all considered to be joined in this way:

RULE_#1 OR RULE_#2 OR . . . OR RULE_#n

The rules are based on implication, which means that, in all, we must consider how to deal with implication, conjunction, logical negation, and disjunction.

The basic propositions are usually stated strictly in terms of the actual linguistic terms. Where the variable is obvious, we can think of a basic proposition as simply the term itself, such as high_fever. However if a standardized or otherwise ambiguous term set is used, then it is necessary to indicate to which variable the proposition refers, as in SE = PS, which is an abbreviation for speed_error is positive_small. Typically the rules are written so that each consequence is a single proposition concerning a single output variable. These are written in the same format as for the input variables in the basic propositions within the antecedents of the rules.

Putting this all together, one particular rule in a fuzzy expert system for medical diagnosis is conceivably the following, though of course this is given as an example only and is not based on any professional medical knowledge.

RULE #27
 IF (infant OR toddler) AND (moderate_fever OR
 high_fever) AND (vomiting OR stomach_pain) AND
 NOT (congestion)
 THEN stomach_virus.

In a fuzzy control application, one rule may be of the nature of the following:

RULE #14
 IF SE=(PM OR PB) AND SC=(PS OR PM OR PB) AND
 NOT (PC=(NB OR NM))
 THEN TC = NB.

where SE = speed error, SC = speed change, PC = pressure change, TC = throttle change, PM = positive medium, PB = positive big, PS = positive small, NB = negative big, and NM = negative medium.

One last consideration before investigating the internal workings of the fuzzy inference engine relates to the form of its immediate outputs, represented in Fig. 4.5d as the y'_j values. Very much like the x'_i input values, these are best thought of as fuzzy subsets of the term set of the respective linguistic variable. However the y'_j values are likely to be somewhat less regular than the x'_i values, because there may be several terms with nonzero membership grades, not just one or two, and the cardinality is unlikely to be exactly 1. For an output representing throttle change in a fuzzy control system, the y'_{TC} value may take such a form as $0.21/NM + 0.15/NS + 0.24/ZO + 0.03/PS$.

Often in the literature the concept of the fuzzy inference engine is presented as though such a device makes inferences directly related to fuzzy numbers, but this way of viewing the process is difficult to follow. We have just discussed a view of the functioning of the inference engine proper in such a way that both the immediate inputs and the immediate outputs of a fuzzy inference engine are not fuzzy numbers, nor are they even defined directly on the reals. Rather these are fuzzy subsets of term sets of linguistic variables. That is they are fuzzy subsets of such sets as {NB, NM, NS, ZO, PS, PM, PB} or {subnormal, normal, slight_fever, moderate_fever, high_fever}. Furthermore the rule base used by the typical sort of fuzzy inference engine is stated in terms of elements of these term sets of linguistic variables. Thus the way we view the central parts of the fuzzy inference engines, they have nothing directly to do with fuzzy numbers, though they do relate to fuzzy subsets of sets of linguistic terms.

It is therefore in the context of elements and fuzzy subsets of sets of linguistic terms that the central part of the fuzzy inference engine must interpret the fuzzy extensions of the logical operators implication, conjunction, disjunction, and negation. In order to design accurately a fuzzy inference engine one must consider carefully how best to extend the concepts of propositional logic to deal with fuzzy sets.

Klir and Folger (1988) specifically discuss an isomorphism between the operations of set theory and propositional logic, first in terms of the conventional theories, then in terms of their fuzzy extensions. Similar approaches are presented elsewhere, but rarely are they explained so clearly. As a general rule such a generalized approach is desirable, but the purposes here are slightly different, and therefore the presentation will be motivated differently. We make the observation that nearly all examples of what one hopes to study by means of fuzzy propositional logic could be reinterpreted as the application of propositional logical operators under circumstances in which the elementary propositions relate to questions of membership in fuzzy sets. This observation or reinterpretation leads to an approach for presenting fuzzy logic that makes the connection between fuzzy sets and fuzzy logic more concrete than mere structural similarity. This way of thinking does not necessarily lead to fuzzy predicate logic, because we are not necessarily interested in quantifiers in the same sense as in predicate logic. To understand the nature of

fuzzy inference in fuzzy control, fuzzy expert systems, and other applications, it is helpful to understand the true nature of the mechanisms.

We can begin by considering propositional logic, where elementary propositions refer to membership in crisp sets, and we consider the meanings relative to the characteristic functions, as explained in Section 4.1. Consider A , a crisp subset of universal set X defined by the characteristic function $\chi_A : X \rightarrow \{0,1\}$; and B , a crisp subset of universal set Y defined by characteristic function $\chi_B : Y \rightarrow \{0,1\}$. Let $x \in X$ and $y \in Y$ and let \wedge , \vee , \neg , \Rightarrow , and \Leftrightarrow represent the logical operators conjunction, disjunction, logical negation, implication, and equivalence, respectively. Then based on Eq. 4.1, the following equivalences are true, regardless of whether $X = Y$.

Conjunction:

$$\begin{aligned} & [(x \in A) \wedge (y \in B)] \\ \Leftrightarrow & [(\chi_A(x) = 1) \wedge (\chi_B(y) = 1)] \end{aligned}$$

Disjunction:

$$\begin{aligned} & [(x \in A) \vee (y \in B)] \\ \Leftrightarrow & [(\chi_A(x) = 1) \vee (\chi_B(y) = 1)] \end{aligned}$$

Logical Negation:

$$\neg (x \in A) \Leftrightarrow \neg (\chi_A(x) = 1) \Leftrightarrow (\chi_A(x) = 0)$$

Implication:

$$\begin{aligned} & [(x \in A) \Rightarrow (y \in B)] \\ \Leftrightarrow & [(\chi_A(x) = 1) \Rightarrow (\chi_B(y) = 1)] \end{aligned}$$

Nothing has to be proven here; these are simply restatements according to the definition of characteristic function.

Next we note that there are various ways of restating these equivalences without changing their meaning for crisp sets. One possible way of restating the preceding operations is

$$[(x \in A) \wedge (y \in B)] \Leftrightarrow \min[\chi_A(x), \chi_B(y)] = 1 \quad (4.10)$$

$$[(x \in A) \vee (y \in B)] \Leftrightarrow \max[\chi_A(x), \chi_B(y)] = 1 \quad (4.11)$$

$$\neg (x \in A) \Leftrightarrow [1 - \chi_A(x)] = 1 \quad (4.12)$$

$$[(x \in A) \Rightarrow (y \in B)] \Leftrightarrow \chi_B(y) \geq \chi_A(x) \quad (4.13)$$

Leaving implication aside for the moment, this is an interesting result for conjunction, disjunction, and negation. In each of these cases, a Boolean function appears very naturally for representing the truth of the logical operation. However it is very important to note that the forms used in Eqs. 4.10–4.12 are by no means the only forms for stating these functions for the case of crisp sets. We will emphasize this point particularly for conjunction and disjunction. Consider for example the following:

$$[(x \in A) \wedge (y \in B)] \Leftrightarrow \chi_A(x) \cdot \chi_B(y) = 1 \quad (4.14)$$

$$[(x \in A) \vee (y \in B)] \Leftrightarrow \chi_A(x) + \chi_B(y) - \chi_A(x) \cdot \chi_B(y) = 1 \quad (4.15)$$

When limited to crisp sets, these are the same functions as Eqs. 4.10 and 4.11, respectively. This is true because characteristic functions take only values 0 and 1. Though it is important to note that a variety of forms could be used, we continue to use the forms of Eqs. 4.10–4.12 for the moment.

One can use the same approach in restating problems that are much more involved; consider the following. Let A be a crisp subset of X , where $x_1, x_2 \in X$. Let B be a crisp subset of Y , where $y_1, y_2, y_3 \in Y$. Let C be a crisp subset of Z , where $z_1, z_2 \in Z$, and finally let D be a crisp subset of W , where $w \in W$. Characteristic functions $\chi_A : X \rightarrow \{0,1\}$, $\chi_B : Y \rightarrow \{0,1\}$, $\chi_C : Z \rightarrow \{0,1\}$, and $\chi_D : W \rightarrow \{0,1\}$, are used to define A , B , C , and D , respectively. We are given the following rule:

$$\begin{aligned} & [(x_1 \in A \vee x_2 \in A) \wedge (y_1 \in B \vee y_2 \in B \vee y_3 \in B) \\ & \quad \wedge \neg(z_1 \in C \vee z_2 \in C)] \Rightarrow w \in D \end{aligned}$$

On the basis of the definitions, this can be restated as follows:

$$\begin{aligned} & [\{\chi_A(x_1) = 1\} \vee \{\chi_A(x_2) = 1\}] \wedge \\ & \quad [\{\chi_B(y_1) = 1\} \vee \{\chi_B(y_2) = 1\} \vee \{\chi_B(y_3) = 1\}] \wedge \\ & \quad \neg[\{\chi_C(z_1) = 1\} \vee \{\chi_C(z_2) = 1\}] \Rightarrow \{\chi_D(w) = 1\} \end{aligned}$$

This in turn can be rewritten as follows:

$$\begin{aligned} & \chi_D(w) \geq \min\{\max[\chi_A(x_1), \chi_A(x_2)], \max[\chi_B(y_1), \chi_B(y_2), \chi_B(y_3)], \\ & \quad 1 - \max[\chi_C(z_1), \chi_C(z_2)]\} \end{aligned} \quad (4.16)$$

Every rule and in fact every collection of rules, written in terms of conjunction, disjunction, negation, and implication, where the elementary propositions refer to membership in crisp sets can be rewritten in this manner.

This approach as explained so far is precisely what is needed to develop a fuzzy inference engine except for the one point that we must still find an appropriate way to generalize the approach so that the crisp sets can be replaced with fuzzy sets. In one sense this seems quite simple. It seems appropriate to use a format such as Eq. 4.16 only replacing the several characteristic functions with membership functions.

However, the problem actually requires more careful consideration. In the case of characteristic functions and crisp sets, Eqs. 4.10 and 4.14 differed only in form, but if extended to membership functions and fuzzy sets, their differences would be more significant. In the case of crisp sets, $f(x, y) = \min[\chi_A(x), \chi_B(y)]$ and $g(x, y) = \chi_A(x) \cdot \chi_B(y)$ are the same function, but in the case of fuzzy sets $f_f(x, y) = \min[\mu_A(x), \mu_B(y)]$ and $g_f(x, y) = \mu_A(x) \cdot \mu_B(y)$ are altogether different functions. Furthermore, it turns out that these two are not even the only choices of function for representing conjunction. In fact, there are infinite choices. A similar problem exists for disjunction of course, which is suggested by comparing (4.11) and (4.15).

These are indeed difficulties of practical significance. Even now the choices of functions relative to conjunction and disjunction are debated amongst the top researchers in fuzzy control, fuzzy expert systems, and other applications of fuzzy inference. Such recognized figures as Hung T. Nguyen, the keynote speaker at the 1993 Symposium of the Japan Society for Fuzzy Theory and Systems, raised the question of whether the functions commonly used are the best [see (Nguyen, 1993) and (Mizumoto, 1993)]. In some sense a similar problem exists for logical negation, because $1 - \mu_A(x)$ could also be replaced by infinite alternatives, such as $\{1 - [\mu_A(x)]^2\}^{1/2}$ for example. However that is a more esoteric problem because it is rarely suggested that an application should use anything other than $1 - \mu_A(x)$ and there are few reasons to believe that there may be advantages in using an alternative.

To present even the essence of a rigorous examination of the functions that could be used for fuzzy extensions to conjunction and disjunction as presented above, one must begin with an axiomatic treatment of the desirable properties. Begin by allowing f to represent a general function representing fuzzy conjunction. Clearly f is a mapping of the form $f: [0, 1] \times [0, 1] \rightarrow [0, 1]$, with the following properties, where $x, y, z, x', y' \in [0, 1]$.

1. $f(0, 0) = 0, f(1, 1) = 1$ (Boundary Conditions 1, as for crisp case)
2. $f(x, y) = f(y, x)$ (Commutativity)
3. if $x \leq x'$ and $y' \leq y$, then $f(x, y) \leq f(x', y')$ (Monotonicity)
4. $f[f(x, y), z] = f[x, f(y, z)]$ (Associativity)
5. $f(x, 1) = x$ (Boundary Conditions 2)

If g symbolizes a general function representing fuzzy disjunction, then its form and axiomatic presentation can be exactly the same as for f except that Axiom 5 is replaced by the following:

$$5.' \ g(x, 0) = x \text{ (Boundary Conditions 2')}$$

Interestingly, this leads to a pair of mathematical structures that are not at all new; in fact these have been studied by mathematicians, particularly statisticians, for a long time now, and these are known as *T-norms* (or *triangular norms*) for the case of Axioms 1–5, as presented for f , and *T-conorms* (*triangular conorms*, or occasionally *S-norms*) for the case of Axioms 1–5', as presented for g . Fuzzy set researchers have recognized that the class of functions that can reasonably be used to represent conjunction (or intersection) is equivalent to the T-norms, and the class of functions for reasonably representing disjunction (or set union) are the T-conorms.

Many basic texts, such as (Terano *et al.*, 1989) and (Kaufmann and Gupta, 1988), present these topics in great detail based on the T-norm and T-conorm definitions. Klir and Folger (1988) also provide a very complete discussion, but it is related to set intersection and union rather than conjunction and disjunction. The latter do not specifically discuss a connection with T-norm and T-conorm definitions, but in other ways their treatment is complete on the various ways of defining infinite classes of operators. Zimmermann (1987) presents one of the most detailed discussions of the practical problem of choosing operators in actual applications.

The purpose here is not to present these matters in such detail as the sources just mentioned but to relate that some matters are not yet settled in applications research relating to fuzzy inference. Some functions, such as drastic product,

$$f(x,y) = \begin{cases} y, & x = 1 \\ x, & y = 1 \\ 0, & \text{elsewhere} \end{cases}$$

meet the axiomatic requirements for T-norms, but it is hard to imagine an extensive use of them in practical applications. But there do continue to be practical issues, such as whether minimum or algebraic product is the better choice for conjunction. Perhaps these can never be resolved, for as Nguyen in (1993) explains, “Most [fuzzy control] designs are ad-hoc, and one could argue inherently so, [and] experiments do not constitute a proof.” The min-max operators will be used here, because they are the most commonly used in all applications, and these are efficient to calculate.

It remains to discuss the matter of inference itself in slightly greater detail, and this can be approached in a variety of ways. We choose to present two of these approaches here, one as typically done in the literature, and the other being our own way. We have intentionally avoided using the manipulations necessary to present a total fuzzy inference system as one large application of the compositional rule of

inference in its usual form because it provides little direct insight into the rationale for fuzzy inference. However we present the use of the compositional rule of inference in a general way, based in part on Mizumoto's presentation (1992a, 1992b).

We begin by noting that the fundamental basis for fuzzy inference can be described as the generalized *modus ponens*. Whereas classical *modus ponens* can be presented as:

$$\begin{array}{ll} \text{Inference rule:} & \text{If } x \text{ is } A, \text{ then } y \text{ is } B \\ \text{Fact:} & x \text{ is } A \\ \hline \text{Conclusion:} & y \text{ is } B \end{array}$$

Generalized *modus ponens* is presented as:

$$\begin{array}{ll} \text{Inference rule:} & \text{If } x \text{ is } A, \text{ then } y \text{ is } B \\ \text{Fact:} & x \text{ is } A' \\ \hline \text{Conclusion:} & y \text{ is } B' \end{array}$$

where A' is given an interpretation such as more or less A and similarly for B . An example could be that if a country is located in the tropics, then it is hot year round and Taiwan is more or less located in the tropics, lead us to the conclusion that Taiwan is more or less hot year round. Though we can state what we mean by it in various, more rigorous ways, generalized *modus ponens* remains the philosophical basis of fuzzy inference applications, and it also remains essentially axiomatic.

The compositional rule of inference views the problem as follows. Given fuzzy subsets A and A' of X , where $x \in X$, and fuzzy subsets B and B' of Y , where $y \in Y$:

$$\begin{array}{ll} \text{Inference rule:} & A \Rightarrow B \\ \text{Fact:} & A' \\ \hline \text{Conclusion:} & B' \end{array}$$

The compositional rule states

$$B' = A' \circ (A \Rightarrow B)$$

where \circ indicates composition. Thus we wish to determine fuzzy set B' as a composition of fuzzy set A' with fuzzy relation $(A \Rightarrow B)$ (considered a fuzzy relation

because in general A and B can also be fuzzy sets). The most commonly used interpretation of the composition of fuzzy set with fuzzy relation is called max-min composition, which in this case implies

$$\mu_B(y) = \underset{x \in X}{\text{Max}} \{\min[\mu_A(x), \mu_{A \Rightarrow B}(x, y)]\}$$

However we have not yet said what we mean by fuzzy relation ($A \Rightarrow B$), with membership function $\chi_{A \Rightarrow B}(x, y)$. Actually there continues to be some controversy over the basic concept of implication as it applies to fuzzy sets, but there must be something at least approximating the idea of fuzzy implication for the concept of generalized *modus ponens* to have meaning, and there must be some way to interpret it. Surprisingly enough the interpretation used in nearly all applications is

$$\mu_{A \Rightarrow B}(x, y) = \min[\mu_A(x), \mu_B(y)]$$

This is curious because it is equivalent to conjunction and quite different from the usual logical interpretation of implication; in any case it leads to:

$$\begin{aligned} \mu_B(y) &= \underset{x \in X}{\text{Max}} (\min[\mu_A(x), \mu_A(x), \mu_B(y)]) \\ &= \min[\underset{x \in X}{\text{Max}} (\min[\mu_A(x), \mu_A(x)]), \mu_B(y)] \end{aligned} \quad (4.17)$$

See (Magrez and Smets, 1975) and (Tsukamoto, 1979) for other interesting approaches to fuzzy inference.

We now view fuzzy inference in our own way. Equation 4.13 for implication in the case of crisp sets is a suggestive starting point for understanding, within the framework as presented here, the way in which generalized *modus ponens* is applied in practice. Replacing the characteristic functions in Eq. 4.13 with membership functions, we have the following interpretation of generalized *modus ponens*:

Inference rule:	If x is A , then y is B
Fact:	$\mu_A(x) = r$
Conclusion:	$\mu_B(y) \geq r$

This rule can be taken to mean that if $\mu_B(y)$, based on our previous information, were less than $\mu_A(x)$, then on encountering this rule, we must adjust $\mu_B(y)$ to equal $\mu_A(x)$; otherwise we leave $\mu_B(y)$ as it stood. All that remains is to determine the value to which we should initialize $\mu_B(y)$ before encountering any rules. Intuition suggests that without any *a priori* knowledge, the initial value should be zero. This last argument as stated may seem to lack rigorous justification, but surely no more so than some aspects of the compositional rule of inference which must be accepted as axiomatic.

A more precise description of a fuzzy inference engine algorithm is reserved for Chapter 5, where the specific case of fuzzy control is considered with a rule base of a different format. However it may be helpful here to present an example of how the concepts of fuzzy inference come together in the workings of a fuzzy inference engine relative to a specific rule. Let us consider once again an arbitrary sample rule in the rule base of a fuzzy control device:

RULE #14
 IF SE = (PM OR PB) AND SC = (PS OR PM OR PB) AND
 NOT (PC = (NB OR NM))
 THEN TC = NB.

Let us further assume that the states of the various input linguistic variables are as follows.

$$SE = 0.75/PM + 0.25/PB$$

$$SC = 0.61/ZO + 0.39/PS$$

$$PC = 0.13/NM + 0.87/NS$$

If these were crisp sets rather than fuzzy sets, then Eq. 4.16 would directly apply, but because the inputs and outputs are fuzzy sets, we generalize this as follows:

$$\begin{aligned} \mu_{TC}(NB) &\geq \min[\max(\mu_{SE}(PM), \mu_{SE}(PB)), \\ &\quad \max(\mu_{SC}(PS), \mu_{SC}(PM), \mu_{SC}(PB)), \\ &\quad 1 - \max(\mu_{PC}(NB), \mu_{PC}(NM))]\} \\ &= \min[\max(0.75, 0.25), \max(0.39, 0, 0), 1 - \max(0, 0.13)] \\ &= 0.39 \end{aligned}$$

Thus $\mu_{TC}(NB)$ must be greater than or equal to 0.39. If it were already greater than 0.39 (due to rules preceding this rule that also had $TC = NB$ as their conclusion), then it remains as it were; otherwise it is now set to 0.39. If we took the alternative approach of joining all rules with the same $TC = NB$ conclusion with disjunctions, the results would be the same. The final results (after defuzzification of the outputs) are also the same if the more confusing compositional rule of inference is used in an all-encompassing way with the max-min form of composition.

Defuzzification is an essential step in most applications of fuzzy inference. Regardless of how we view a fuzzy inference process, the output of the inference process itself is a fuzzy set. Fuzzy sets can conceivably be acceptable outputs from an expert system used in a decision-support capacity, but in many fuzzy expert systems and in all fuzzy control devices, this form of output is unacceptable. A fuzzy set is a formalism for representing vagueness in human thinking, so it has no

meaning whatsoever to a steam engine, a cement kiln, a train-braking system, or a washing machine. Therefore it is essential that the output fuzzy sets be translated into precise (crisp) values which can result in actuator actions in a control device or precise conclusions in an expert system.

Despite this essential role, defuzzification arguably continues to be the most *ad hoc* aspect of the generally *ad hoc* process of fuzzy control design and the design of other applications of fuzzy inference, although there have been recent attempts at rigorous approaches [see (Filev and Yager, 1991) and (Zhao and Goving, 1991)]. In the historical development of fuzzy control, methods for translating fuzzy sets back into real (crisp) numerical values have been guided more by intuition than anything else.

Our presentation adds an additional step to the defuzzification process, but it makes the process no more difficult. We consider the output of the application of the rules to be fuzzy subsets, not of the real numbers but of the term sets of the linguistic variables. That is we consider the outputs y'_j of the fuzzy inference engine proper to be of the nature of $0.21/\text{NM} + 0.15/\text{NS} + 0.24/\text{ZO} + 0.03/\text{PS}$. These must each first be translated into fuzzy subsets of the value set of the corresponding base variable (typically the reals), a form we refer to as the y''_j form, before finally being translated into a crisp value y_j in the base variable. However this step presents no problem if we accept the commonly used definition for composition of a fuzzy set with a fuzzy relation. Note: We do not use composition to define the overall process of fuzzy inference as sometimes done, but rather for our own special purposes because of the ways we defined the outputs of the rule-based inference (as fuzzy subsets of term sets) and the output linguistic variables (in terms of fuzzy relations).

We use fuzzy relations to define the relation between the linguistic and the base variables in all cases, which of course includes the outputs. Thus for the j th output, a fuzzy relation $R_j(T_j, \mathcal{R})$ is defined in advance to relate what we mean by the terms used for the linguistic variable, and the value y'_j is a fuzzy subset of T_j . Thus we can determine y''_j , a fuzzy subset of \mathcal{R} , by composition of a fuzzy set and a fuzzy relation:

$$y''_j = y'_j \circ R_j$$

The usual interpretation of composition under these circumstances [see for example (Zimmermann, 1987) for a good, general explanation without regard to inference] is *max-min composition*, which in this case results in the following:

$$y''_j(y) = \underset{t \in T}{\text{Max}} \{ \min[\mu_{y'_j}(t), \mu_{R_j}(t, y)] \} \text{ for all } y \in \mathcal{R} \quad (4.18)$$

Though for the typical output variable the result y''_j is a fuzzy subset of the reals, generally, it will not be a fuzzy number because it will usually not be normal and also may not be convex.

Whether using the standard interpretation of fuzzy inference or our own, we still face the problem of translating such (generally not normal and often not convex) subsets of the reals into crisp values in the reals. This is the part for which approaches used to date, though reasonable, are somewhat *ad hoc* because until recently there was little rigorous analysis. Two methods commonly used to date are the *mean-of-maxima (MOM)* and the *center-of-gravity (COG, also centroid or center-of-area) methods*.

The MOM method was used in the earliest fuzzy control applications. If the membership function μ_y'' attains a distinct maximum at some value y , we simply take that value. However, particularly where min-max logic and max-min composition are used, it is often true that the maximum is not attained at a distinct point but over a possibly wide plateau. It is also possible, though not so common, that the (global) maximum could be reached at two or more distinct points. In these cases, as the name suggests, the mean-of-maxima method indicates that the midpoint of the plateau or the mean of the distinct maximum points should be used.

The COG method has been more common in recent applications. At first thought it may seem less efficient to calculate, but in practice it is usually quite efficient because the value set of the base variable is typically a much smaller set than the real numbers. In fact it can nearly always be considered a finite set because there are practical upper and lower limits on the values as well as limits in the precision of measurement for inputs and in the precision of adjustment for outputs. For output variables in fuzzy control applications, the number of levels to which the actuator can be adjusted is usually not only finite but small—less than a few dozen. Thus, in practice crisp output under the COG method is usually calculated as follows:

$$y_j = \frac{\sum_{y \in Y_j} y \cdot \mu_{y''_j}(y)}{\sum_{y \in Y_j} \mu_{y''_j}(y)}$$

where Y_j is the actual value set of the base variable of the j th output. If the control device is implemented by means of a general purpose (nonfuzzy) microprocessor for example, this calculation is very rapid.

Both the MOM and the COG methods seem reasonable enough, and without more complete analysis, it was appropriate to apply them. For many years the research community remained in this state—recognizing the incomplete and *ad hoc* nature of knowledge on this matter but accepting it for lack of anything better. It has been only in the past few years that significant progress has been made toward analyzing defuzzification.

Probably the most significant step was made by Filev and Yager [see (1991), where they present their parametrized approach to defuzzification methods]. Their work is related to an application of uncertainty invariant transformations, presented by Klir (1990), to the conversion of a possibility distribution (here, the membership function representing a given y_j'' value) to any distribution in a parametrized class of probability distributions. The authors refer to these as basic defuzzification distributions (or BADDs). Filev and Yager claim that to take the expected value of any distribution in this class results in a valid defuzzification method.

Presenting the BADD idea in our notation, for a given y_j'' value (a fuzzy subset of Y_j for some system output j) and for a choice of the parameter $\alpha \in [0, \infty)$, the following probability distribution is a BADD:

$$p_\alpha(y_b) = \frac{[\mu_{y_j''}(y_b)]^\alpha}{\sum_{y \in Y_j} [\mu_{y_j''}(y)]^\alpha} \quad \text{for all } y_b \in Y_j$$

We then obtain a crisp value by taking the expected value of this BADD:

$$y_\alpha = \sum_{y \in Y_j} y \cdot p_\alpha(y)$$

According to the BADD concept, y_α is a valid result for the defuzzification of y_j'' .

Where $\alpha = 1$, the BADD approach results in the center-of-gravity method; and where $\alpha \rightarrow \infty$, it results in the mean-of-maxima. Thus BADD is a generalization of both previously accepted defuzzification methods. Furthermore, the authors discuss an information-theoretical perspective on this parametrized approach, and state the view that setting a higher value of α is equivalent to stating a higher level of belief in the fuzzy set to be defuzzified, with $\alpha = 0$ equivalent to a state of a complete lack of belief in the fuzzy set and $\alpha = 1$ equivalent to average belief.

Filev and Yager make clear that in their view, the most obvious advantage of this approach is that it adds an element of adaptation to the defuzzification process. They consider that the parameter could be adjusted either according to the subjective degree of confidence in the other parts of the approximate reasoning model or by machine learning. This view is significant and worthy of greater attention, but for our work here, we will limit ourselves to the now quite standard center-of-gravity method, that is $\alpha = 1$. On the other hand, it will be quite obvious how the particular approach to auto-tuning that we will discuss in Chapter 9 could be easily extended to include automatic adjustment of BADD parameters as well as fuzzy relations.

In Chapter 5 we examine the fuzzy inference process as applied in a typical fuzzy control application. We demonstrate more clearly why our perspective is

easier to understand than the standard treatment and also prove that the end results are the same in either case under standard operators.

4.6. Closing Remarks on Fuzzy Set Methods

4.6.1. *Vagueness and Randomness*

Even as one becomes gradually more familiar with fuzzy set theory it is still necessary to remind oneself frequently that fuzzy sets are fundamentally different from probabilities: Anyone familiar with probability and statistics recognizes that the ways in which one manipulates membership functions of fuzzy sets are typically quite different from the ways one manipulate probability distributions. More importantly the two formalisms represent fundamentally different concepts—vagueness of concept in human thought on the one hand and randomness of natural events on the other; and therefore it is not even reasonable to attempt to compare them.

In studying some phenomena, one may be faced with situations in which there is both a significant factor of vagueness of concept and a significant factor of randomness. Fortunately there was increased interest during the 1980s in developing methods of fuzzy statistics and similar concepts to deal with such situations [see (Kandel, 1986) and (Negoita and Ralescu, 1987) for two good introductions]. To date there does not seem to be much interest in directly applying these theories to stochastic control problems, but it is possible in the near future.

4.6.2. *Fuzzy Sets and Artificial Intelligence*

The intersection between the set of practical applications of fuzzy set methods and the set of practical AI applications is very large. In fact if we accept the definitions in Chapter 3, there is some justification for saying that the two sets are close to being identical.

Chapter 3 viewed a good definition of AI to be the collection of applications of automated computing devices that are based wholly or in part on heuristics. Heuristics in turn were defined as rules of thumb—reasoning methods that are generally true or generally lead to effective results. What could be called the AI approach asserts that computing devices cannot function nearly so effectively as human beings at certain tasks unless those devices are capable of dealing with forms of knowledge that include generalizations. However, AI as a field is better described as goal-specific than as method-specific.

Fuzzy set methods represent probably the first truly rigorous formalism for defining and analyzing vague concepts, especially concepts expressed relative to vague classes. Without a specific plan for application in tangible devices, fuzzy set theory asserts that extensions to mathematical structures permit modeling the

tendency of the human mind to reason with generalizations is ultimately useful. As a field fuzzy set theory is more method-specific than goal-specific.

The two fields are therefore essentially the same in rationale but differ primarily in emphasis, goal-specific or method-specific. It is difficult to draw other conclusions than that the two fields should complement each other. This does not necessarily imply that every AI application will use fuzzy set methods, but a large proportion of them should if the best tool is chosen. Going the other way, the proportion of applications of fuzzy set theory that are also AI applications depends on the definition chosen for this vaguely defined field, AI. If the heuristic-basis definition used here is accepted, then every application that makes any use of fuzzy set theory within a computing device qualifies as an AI application as well. This is because fuzzy set identifications are heuristics.

In any case fuzzy control clearly falls within the intersection of the two fields, AI and fuzzy set methods. Specifically fuzzy control is all cases within that intersection where the automated computing device interfaces directly with another physical device with the objective of automating its operation as well. It could equally well be called knowledge-based control.

CHAPTER 5

Classical Fuzzy Control Design and Implementation

Having discussed relevant aspects of conventional control theory, AI, and fuzzy set theory, it is now possible to bring these three together to discuss the fuzzy control design process in a fairly straightforward way. Many aspects are in fact so straightforward that they seem almost embarrassingly easy. However, other aspects of fuzzy control are still somewhat difficult to master. One example of this is that even though we began a discussion of fuzzy inference processes for approximate reasoning in the previous chapter, it may still not be easy to understand exactly how this would work in practice. On a still more practical level, the issue of tuning control devices that were designed almost totally on the basis of heuristics is a significant potential problem. One purpose of this chapter is to consider in greater detail each of the steps in the fuzzy control design process. One point of special significance in this regard will be to discuss an alternative perspective or interpretation of inference algorithms for fuzzy inference, and to compare this to the customary perspective in considerable detail. Chapter 5 concludes with a discussion of a few of the most significant examples of applications in the historical development of fuzzy control and with a few more points of comparison between fuzzy and conventional control methods.

5.1. High-Level System Design

From the view of an experienced problem solver, the true first step in a design process is usually the problem-definition phase, but for the purposes of our more strictly technical discussion here, the first step can be taken as the high-level system design. This is the only step in the design process that tends to be essentially the same in both fuzzy and conventional design methods. In both cases high-level system design is primarily concerned with defining input and output variables, that is it is concerned with the selection of the system attributes to measure and to manipulate, and with the ways of measuring them. Note: in the case of fuzzy control, we are speaking here of the base variable definitions only; the linguistic variable definitions come later in the process.

Regardless of the context, whether analyzing a natural phenomenon or designing an artificial system, high-level system definition typically proceeds in a rather *ad hoc* fashion. Only general systems theory, a field that regrettably is not widely known, has attempted to approach this matter in a way that is both rigorous and general. It is true that certain specific analytical methods have techniques for determining which variables are insignificant and therefore can be eliminated, and in some ways neural nets can be made to perform this function automatically, but in most cases, the designer or analyst still begins by considering what seems a reasonable way of viewing the system. Thus in the case of both fuzzy and conventional control methods, as well as in many other applications not related to control, this step is frequently based on heuristic thinking of the designer, and there is not necessarily anything wrong with that.

A few points should be emphasized specifically as they apply to control device design. First, as mentioned in Chapter 2, it is important to avoid confusion about what one means by an input or an output variable. In the case of feedback control, the inputs of the control device (the subsystem that one is in the process of designing) are typically the outputs of the plant, and can also be considered the outputs of the overall system. Similarly, the outputs of the control device typically take the form of adjustments to the inputs of the plant.

A second point relating to control applications is that the choice of variables cannot always be based strictly on how significant the attribute is but rather on how effectively or economically it can be measured or manipulated in practice. Hardware constraints must be considered. What types of sensors or actuators are available?

A similar and additional consideration applies to conventional control designs. Because nonlinear models are so much more difficult to use than linear ones in the case of conventional design methods, there may be circumstances for which the choice of variables depends in part on which variables would make it most possible to build an effective linear model of the system. This consideration is unimportant in fuzzy control design, and in this sense the designs are less constrained.

In any case the process of defining a system variable is more than just selecting an attribute of the system. One must also consider how one plans to measure or to scale the attribute. In the case of control applications, it is quite common to consider only a finite number of discrete settings, especially for output variables but often for input variables as well. In fact, the number of settings for a given variable is often quite small, sometimes less than a dozen. Also, if dynamic compensation is used, then a given attribute in the system is represented by more than one input variable. It is not always obvious how best to measure a derivative or integral even after determining the proportional element.

5.2. Naive Physics and Fuzzy Control Rules

(Raggett and Bains, 1992) contains an entry on the term naive physics: The “real physics” we are taught in school emphasizes “accurate enumeration.” But even before we learn that, we already know a naive physics which is based on our experiences, and takes the form of such heuristics as, “Things thrown up eventually come down: the harder they are thrown up, the longer they take to come back again.” This is how we normally reason about physical phenomena.

This concept of naive physics is precisely the key to understanding the rationale for the classical approach to fuzzy control. It is the basis of production rules or rule base of the control device, often called fuzzy control rules.

While heuristic knowledge expressed in the form of IF-THEN rules is in most ways very easy to grasp conceptually, in this section we discuss some specific considerations of this process related to fuzzy control. One question is the source of the heuristic knowledge for developing the rules, and another is the matter of rule formats.

As least from the point of view of the classical approach to fuzzy control design, there are only two reasonable sources for the heuristic knowledge to be expressed in the rules:

- One or more human operators already skilled in the specific control process in question
- The engineer or whoever is actually designing the fuzzy control device

There are advantages and disadvantages to each of these potential sources, and the choice in practice would depend on several factors, but particularly on the nature of the specific application and the experience base for it.

If the objective is to automate by fuzzy control methods a particular plant or process that was previously controlled manually, then it would seem most practical to make use of the expertise of one or more of the most skilled human operators as the primary source for the knowledge base. Such expertise would include both the knowledge in the rules and in the linguistic variable definitions to be discussed below. Although designers of fuzzy control devices should certainly be encouraged to use this approach as much as possible, this is a knowledge-engineering problem with its own particular set of challenges, and one must think carefully about how to proceed effectively.

For example, if the objective is to automate control of an industrial process previously controlled manually for many years, then although the expertise of the most skilled operators is potentially very useful, there may be several organizational problems that make it difficult to capture and use such expertise. People with industrial experience will know that in many cases industrial operators and engineers regrettably do not communicate very well. They may not “speak the same

language,” and they may not even respect each other very much. These organizational conditions seem to exist not just in Western countries, but often in developing countries and formerly Communist countries as well. One of the few places where these conditions are not common is in Japan. Because communication is a difficult aspect of any knowledge-engineering process even under the best of conditions, this type of organizational problem makes an already difficult process much more difficult still. For these reasons, there is a high probability that the fuzzy control device designer would forgo the use of operator expertise.

However, before giving up completely, one should consider one other possible way of using the knowledge of experienced operators. It has often been suggested by Sugeno and others that one approach may be simply to observe the behavior of these operators as they go about their control operations, then modeling that behavior either with formal or informal models. In some applications this may be as effective as consulting with the operators.

In a different type of application, the approach might be applied quite differently. For example, if one desired to use fuzzy control methods to automate a particular type of consumer appliance, then it may be appropriate to think more of the typical user than of the skilled operator. How would the typical consumer expect his automated microwave oven to behave in various circumstances? Or, alternatively, how does the typical user operate the nonautomated microwave that he currently owns? Thus the design process may be based to a large degree on market research. Note: While this is true for any consumer product, in the case of fuzzy control, consumer preferences and expectations may go into a knowledge base that becomes part of the product itself.

There are many circumstances when one wishes to automate a process for which no experience base exists in manual operation. Even when the process to be automated is not entirely new, it may have been structurally changed in some way that dramatically changes the ways it can be operated. Furthermore, although the ability of humans to learn, mainly by intuition and practice, how to control complex systems is very impressive, it is also true that there are limits even to what humans can do. Reaction times and attention spans are perhaps the two most serious constraints in this regard.

Whether for the lack of an established skill base or for other reasons, one may decide to rely on one's own knowledge as the sole source for the knowledge base when designing a fuzzy control device. Although the designer may not be as intimately familiar with the process in all its subtle variety as a skilled operator could be expected to be, the designer may be able to compensate for this lack in various ways. For example, by studying the plant or process in both structured and unstructured (“just play with it”) ways, the designer may be able to develop an extremely desirable balance between analytical and intuitive knowledge about the system. Such a balanced perspective is at the least an excellent place to start in developing any type of heuristic knowledge base.

On the question of formats for the fuzzy control rule base, many variations are possible. However, the differences tend to be so basic as to be almost of a trivial nature. The point is simply to use a format that makes it as simple as possible to keep the rule base well organized for easy development and use.

Term sets need not be the same for all linguistic variables in the rule base. It is possible for example for one variable to vary over seven terms, such as {NB, NM, NS, ZO, PS, PM, PB}, while another varies over only five, such as {NB, NS, ZO, PS, PB}. Note: Strictly for the sake of simplicity of explanation, we assume a standard seven-term set like the one just listed for all variables unless otherwise specified.

All sorts of conventions are possible for IF-THEN rules. Under almost any convention, conjunction (AND) is used, but one must decide whether to allow also the use of disjunction (OR) and logical negation (NOT) within the premise part of a rule. It is possible to avoid it, but the result would generally be a larger number of rules. The usual convention in fuzzy control was to allow disjunction and logical negation, and thereby keep the number of rules as small as possible.

Another fairly common convention in fuzzy control that is rarely assumed for expert systems requires each rule to contain every input variable in the antecedent part. If this convention is applied, then even if a given input has no bearing on the outcome of that particular rule, it must still be listed. However those who follow this convention usually allow themselves a type of shorthand in the form of the special term, ANY. For example in a fuzzy control device with four inputs—speed error (SE), speed change (SC), pressure error (PE), and pressure change (PC)—one of the rules in the part of the rule base governing the variable throttle change (TC) may be

```

IF      SE = (ZO OR PS) AND
        SC = PB AND
        PE = (NB OR NM OR NS OR ZO OR PS OR PM OR PB)
        AND
        PC = (NB OR NM OR NS OR ZO OR PS OR PM OR PB)
THEN   TC = NS.

```

However using the ANY form of shorthand, this can be abbreviated slightly as:

```

IF      SE = (ZO OR PS) AND
        SC = PB AND
        PE = ANY AND
        PC = ANY
THEN   TC = NS.

```

Of course in other conventions, the lines relating to PE and PC may be omitted altogether.

In any case, it is usually reasonable to allow only one variable in the consequent part of a rule. In fact where there is more than one output variable, it is sensible to break up the rule base into separate parts, each part containing all rules pertaining to a given output variable. Mamdani and Assilian did this in even the earliest fuzzy control demonstration, where they referred to the two parts as the “heater algorithm” and the “throttle algorithm,” respectively.

But in the end, it may be most practical not even to bother with an explicit IF-THEN format for the rules. It may be just as easy to express the entire rule base in the form of tables, one table for each output variable. If the number of input variables is m and the number of output variables is n , then we need n separate tables, each table in the form of an m -dimensional array. At first thought, this seems rather bulky, but actually it is not so bad when we remember that for purposes of the rule base, each variable has only approximately seven values, such as NB–PB. Note: In Chapter 6 we consider an approach to variable reduction in fuzzy control, which often allows the dimensionality of various tables to be reduced. For now, as a simple example, a control device with two inputs, SE (speed error) and SC (speed change), and only one output would require only one two-dimensional table that may resemble Table 5.1.

This tabular format is clearly easy to manipulate within the computer without modification, but, as a conceptual point, it also makes us think of the rule base differently. Looking at Figs. 4.5b and c, we note that the difference between the two views of a system is not so significant as it first seems. With the tabular format, it is quite simple to think of the rule base as Fig. 4.5b rather than 4.5c, that is, as a collection of functions—one function for each output.

This viewpoint is emphasized throughout the remainder of Chapter 5.

Given a control device with m inputs (including derivatives and integrals) and n outputs, we define n functions in the following way: Let

- T_{xi} represent the term set of input variable i for all $i = 1, 2, \dots, m$, and similarly let
- T_{yj} represent the term set of output variable j for all $j = 1, 2, \dots, n$

Typically T_{xi} or T_{yj} equals {NB, NM, NS, ZO, PS, PM, PB} or something similar. Then for each $j = 1, 2, \dots, m$ we define a function:

$$f_j : T_{x1} \times T_{x2} \times \dots \times T_{xm} \rightarrow T_{yj}$$

with values defined by the tables. This is all it takes to express the rule base.

This form of expression brings another issue into sharper focus. Must these functions be defined for all combinations of the term sets of the input variables? The answer is quite obvious: If the designer of the control device cannot conceive of a circumstance where a particular combination of input values might occur, then it is reasonable to leave the function undefined at that point. Note: The designer

may later learn that he/she was wrong. For now we place an appropriate symbol into the table to indicate undefined, and later we will consider how to handle these points during the inference process.

If the designer decides to define what action to take the tabular (or function-based) format is also convenient: We know by looking at the table defining the function whether it is defined at all points. If not, but we desire it to be, then we can correct it immediately. If we do not require the function to be complete, then we can easily handle that in the inference algorithms.

5.3. Linguistic Variable Design

There is perhaps no conclusive answer to the question of which should come first in the design process, the fuzzy control rules or the linguistic variable definitions. On the one hand, experience with other design approaches suggests that we had best define specifically what we mean by a term before developing rules using it. But there are two reasons why this requirement does not necessarily apply in this circumstance: First fuzzy control design is largely a heuristic design process, so it is quite different from more traditional design methods. In the philosophy of a heuristic design method, it is reasonable to begin with the general, then move to the specifics afterward, that is, to begin with vague statements, then elaborate on them later. Second from our perspective presented in this book, (fuzzy) quantification need not be considered equivalent to defining a term. Therefore it seems reasonable to state fuzzy control rules, then define the linguistic variables. However it is probably more reasonable for the two steps to progress at least somewhat in parallel. At the very least we must know what the term set for each variable will be before designing the control rules.

As we mentioned in Chapter 4, the best formalism for defining a linguistic variable is the fuzzy relation, not the fuzzy set *per se*. This has considerable bearing on the following sections, but for the practical problem we consider now, it is irrelevant whether we call it a fuzzy relation or a collection of fuzzy sets. If we use such terms as negative big (NB), negative medium (NM), etc., then we must explain more specifically the quantities to which they refer.

This brings us to a criticism often made of fuzzy set theory in general. It is all well and good, say the critics, to develop a methodology for manipulating fuzzy sets which supposedly represent vague concepts, but how does one come up with these sets in the first place? Actually, there are a variety of approaches, some more subjective than others, but in any case some degree of subjectivity is only to be expected. After all, fuzzy sets are a formalism for representing vague aspects of human reasoning. But to say that something is subjective is not the same as saying that it is arbitrary.

Table 5.1. Example of a Control Rule Table

		SC						
		NB	NM	SN	ZO	PS	PM	PB
SE	NB	PB	PB	PM	PM	PS	ZO	NS
	NM	PB	PB	PM	PS	ZO	NS	NS
	NS	PB	PM	PS	PS	ZO	NS	NM
	ZO	PM	PM	PS	ZO	NS	NM	NM
	PS	PM	PS	ZO	NS	NS	NM	NB
	PM	PS	PS	ZO	NS	NM	NB	NB
	PB	PS	ZO	NS	NM	NM	NB	NB

The exact approach to use in defining a fuzzy set to represent a given concept depends on many factors, including at least the following:

- General nature of the application area (overall context)
- Specific characteristics of the particular application (system complexity, sensitivity, etc.)
- Whether the fuzzy set is to represent the vague concept as viewed by only one particular person or by several people
- Preferences of the developer, including the degree of sophistication in methods and prestated restrictions (e.g., fuzzy numbers, TFNs only, etc.).

Variations in these factors result in many approaches, but for our purposes here, it is sufficient to consider only the most straightforward approaches that refer to the classical approach to fuzzy control.

Despite the subjective nature of the process, the definition of a fuzzy set is not arbitrary; in fact it can be quite simple, especially in fuzzy control applications if we limit ourselves to fuzzy numbers with piecewise linear membership functions. This means TrFNs or TFNs.

To relate a given vague concept, such as the speed_error is negative_medium (SE = NM), to a fuzzy number with a piecewise linear membership function requires the thoughtful answer to only three questions:

- What is the value below which I am absolutely certain that the concept does not apply?
- What is the single value (if it is required to be a TFN) or the interval (if it can be a general TrFN) that best represents the point(s) at which I am absolutely certain that the concept applies?
- What is the value above which I am absolutely certain that the concept does not apply?

The extreme cases, such as NB or PB, in a typical fuzzy control design are simpler because we can omit either the first or last question and assume that there is no lower or upper limit, as appropriate. In fact if we make the customary assumption that (in our formalism):

$$\sum_{t \in T} R(t,x) = 1 \quad \text{for all } x \in \mathcal{R}$$

then there are fewer questions to answer.

This process must be completed for each input variable (including derivatives and integrals) and each output variable. If the design is based on the expertise of several people, such as several experienced operators, then there must be some method for representing their concepts collectively. One way would be to take means or medians for the answers to the preceding questions. However, in some situations, it may be more reasonable to form a consensus in some way such as by a fuzzy Delphi method.

5.4. Perspective on Inference Algorithms for Fuzzy Control

Chapter 4 introduced an interpretation of the inference process for fuzzy control and other applications of approximate reasoning that differed from the standard interpretation. The next section further compares the two interpretations, showing that they lead to equivalent results under min-max operators and reasonable assumptions. That section also explains why we view the alternative interpretation as much easier to understand than the standard one and why it leads us to think more naturally in terms of efficient algorithms.

Before preceding this section explains our alternative interpretation in greater detail.* For readers with a limited background in mathematics, this section and the next may initially seem abstract, but do not give up too quickly. A review of our numerical example at the end of this section and more careful reflection on what the notation actually means will clarify matters.

We review what is necessary to specify a simple fuzzy system of the type that forms the basis of a typical fuzzy control device. The easiest way to describe such

*The level of detail is such that anyone with moderate systems design and programming skills should be able to sit down almost immediately and write a program to implement our interpretation of approximate reasoning for any given specific situation. We assume that such a language as Pascal, with enumerated type and other modern features, is used because this assumption makes the task of writing such a program easier both to explain and to do.

a system is to state that its two main components are a collection of crisp functions and a collection of fuzzy relations. The crisp functions are nonnumerical, defined on the term spaces and representing the control rules or other production rule system, as discussed in Section 5.2. Fuzzy relations relate the meanings of the various input and output term spaces to their respective base variable spaces, as described in Section 5.3. Thus the following conceptual equation is only a slight simplification:

$$\begin{array}{c} \text{Crisp functions} \\ + \text{Fuzzy relations} \\ \hline \text{Fuzzy system specification} \end{array}$$

However this still omits a few important details. An actual inventory of what we must identify to describe completely a classical fuzzy control device or other similar type of fuzzy system with m inputs and n outputs follows:

1. the m input and n output term sets.
2. n sets of possible crisp output values (the base output space, often finite); the base input space can be considered, though this may not be essential.
3. n crisp functions on the overall input term space, each to an output term set.
4. $m + n$ fuzzy relations, each relating the quantitative context to the linguistic (term-set). Context for a given variable.
5. the details of the fuzzy inference engine and defuzzification method (i.e., which t-norms/t-conorms to use for conjunction/disjunction, which defuzzification scheme to use, etc.).

There is no reason to concern ourselves too much with Item 5 at this time. Let us assume for the moment that we typically use min-max operators and COG defuzzification and can easily specify otherwise where necessary. As a logical next step, let us consider a detailed notational scheme for representing Items 1–4. We also consider notation for certain types of intermediate values that become useful later.

5.4.1. Knowledge Representation Formats for a Simple Fuzzy System with m Inputs and n Outputs

5.4.1.1. Terms and Term Sets. Inputs: For all $i = 1, 2, \dots, m$, let T_{xi} be the term set of the linguistic variable for input i . For example in a given application we may say that $T_{x2} = \{\text{NB}, \text{NM}, \text{NS}, \text{ZO}, \text{PS}, \text{PM}, \text{PB}\}$. Depending on the design T_{xi} could be the same for all inputs i or different. In any case we typically write elements as $t_{xi} \in T_{xi}$. Let

$$T_x = T_{x1} \times T_{x2} \times \dots \times T_{xm} \text{ and let } t_x = (t_{x1}, t_{x2}, \dots, t_{xm})$$

Outputs: Similarly let T_{yj} represent term sets for all outputs $j = 1, 2, \dots, n$. If at some time we assume j to be fixed, then we represent elements as $t_y \in T_{yj}$.

5.4.1.2. Base Variable Values and Spaces. **Inputs:** Let X_i be the set of all values that the i th input is allowed to take on the base variable. For now it makes little difference whether we consider X_i to be all reals or some finite subset of the reals. For example if the third input almost never goes below -15 or above 15 units and if 0.5 unit is clearly sufficient precision, then we can say that $X_3 = \{-15.0, -14.5, -14.0, \dots, 14.5, 15.0\}$; for now we can just as easily consider that $X_3 = \mathcal{R}$.

In any case let $X = X_1 \times X_2 \times \dots \times X_m$, the Cartesian product of these sets. Assume at any given instant that the input to the system is the crisp vector $x \in X$. In other words, $x = (x_1, x_2, \dots, x_m)$, where $x_i \in X_i$ for $i = 1, 2, \dots, m$.

Outputs: For $j = 1, 2, \dots, n$, let Y_j be the set of values that the j th output is allowed to take. But as opposed to the inputs we require Y_j in each case to be an explicit (preferably finite) subset of the reals. This requirement allows us to defuzzify result more efficiently. The definition of the subset depends on the nature of the output—its range and precision.

Our final objective is to produce crisp outputs $y_j \in Y_j$ for $j = 1, 2, \dots, n$. However, because we can calculate the final outputs independently, it is rarely necessary to view these as a vector or their space as a Cartesian product.

When the value of j is assumed to be fixed, we represent arbitrary elements as $y_b \in Y_j$.

5.4.1.3. Crisp Functions (Rules). The control rules are defined in tabular form as represented by functions; thus for each $j = 1, 2, \dots, n$, we define the function:

$$f_j : T_{x1} \times T_{x2} \times \dots \times T_{xm} \rightarrow T_{yj},$$

or more simply as:

$$f_j : T_x \rightarrow T_{yj}.$$

This can be represented in the computer memory as one m -dimensional array for each output, ranging in each dimension on a term set and also taking values on a term set. Enumerated types make this simple to represent neatly. Although this may seem imposing, in typical applications, this does not actually require much memory.

These functions can be undefined at some points if the rule base does not cover all possible combinations of input terms. As previously stated there is no serious consequences to having some points undefined in the statement of these functions.

When building functions we must think only of control rules in this way—defined on crisp values of the term sets. However we should keep in mind that the extension principle makes it very easy to extend the definition of these functions to fuzzy subsets of term sets. Thus we can easily think as follows:

$$f_j : \mathcal{F}(T_x) \rightarrow \mathcal{F}(T_{yj})$$

where $\mathcal{F}(A)$ indicates the fuzzy power set of set A .

5.4.1.4. Fuzzy Relations. Inputs: For each $i = 1, 2, \dots, m$, let R_{xi} represent a fuzzy relation expressed by:

$$R_{xi} : T_{xi} \times X_i \rightarrow [0, 1]$$

Outputs: For each $j = 1, 2, \dots, n$, let R_{yj} represent a fuzzy relation expressed by:

$$R_{yj} : T_{yj} \times Y_j \rightarrow [0, 1]$$

We nearly always assume the constraint on these fuzzy relations that the fuzzy subset of the reals formed by applying each of these fuzzy relations to each term in the term set is a fuzzy number. As explained earlier it is usually reasonable to assume further that this must be a particular type of fuzzy number, such as a TFN. Thus it is typically possible to use a concise parametric form of expression for these fuzzy relations, either on paper or in the computer memory. For example if TFNs are assumed, then each fuzzy relation can be expressed as a two-dimensional array or table, ranging in one dimension through the values of the term set (such as NB to PB), and in the other dimension from 1–3, to represent the parameters of the (a_1, a_2, a_3) form of the TFN expression. On paper a possible example is

$$\begin{aligned} \text{NB} &: (-\infty, -12, -8) \\ \text{NM} &: (-12, -8, -4) \\ \text{NS} &: (-8, -4, 0) \\ \text{ZO} &: (-4, 0, 4) \\ \text{PS} &: (0, 4, 8) \\ \text{PM} &: (4, 8, 12) \\ \text{PB} &: (8, 12, \infty) \end{aligned} \tag{5.1}$$

In a computer program the enumerated type feature of languages like Pascal is quite convenient for defining and manipulating such tables. Naturally we cannot represent infinity in the computer as a real value, but it is sufficient to substitute a very large value, such as 10^{20} .

We just considered all of the things we must specify to define a typical fuzzy control design or other similar fuzzy system. Now, as one final preliminary before discussing the approximate reasoning process itself, let us consider the types of intermediate values we can expect to encounter during that process.

5.4.1.5. Intermediate Values. As we will see momentarily, in our view it is quite sensible to break down the process of approximate reasoning into four logically distinct steps with values passed from each step to the next. Thus, there are three types of intermediate variables to consider. All three types are fuzzy rather than crisp variables. That is, their values in each case will be fuzzy sets.

5.4.1.6. Fuzzy Linguistic Values for Inputs. For each $i = 1, 2, \dots, m$, let x'_i be some particular fuzzy subset of T_{xi} (in each case we will identify which particular fuzzy subset in discussing the inference process). Thus x'_i takes values in $\mathcal{F}(T_{xi})$, the fuzzy power set of the i th term set. The actual value of x'_i can be represented by its membership function $x'_i : T_{xi} \rightarrow [0, 1]$, or we can represent it by the notation described in Chapter 4. For example under some given circumstances in some given fuzzy system, we may end up stating that $x'_3 = 0.23/\text{NM} + 0.77/\text{NS}$ for the third input of the system.

5.4.1.7. Fuzzy Linguistic Values for Outputs. In precisely the same way, for any $j = 1, 2, \dots, n$, let y'_j represent a particular fuzzy subset of T_{yj} . Thus y'_j takes values in $\mathcal{F}(T_{yj})$, and it can be represented by the membership function $y'_j : T_{yj} \rightarrow [0, 1]$ or by any other useful format.

5.4.1.8. Fuzzy Subsets of Real Output Spaces. For each $j = 1, 2, \dots, n$, let y''_j represent a particular fuzzy subset of Y_j . Then these y''_j variables take values in $\mathcal{F}(Y_j)$, the fuzzy power sets of the base output spaces. We can represent y''_j by its membership function form $y''_j : Y_j \rightarrow [0, 1]$.

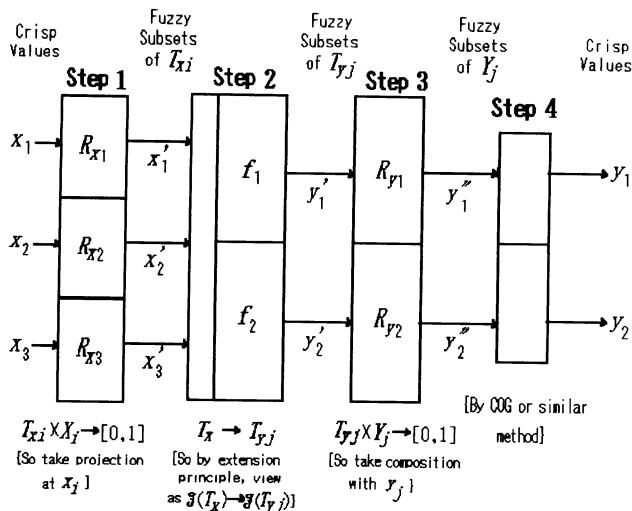


Figure 5.1. Alternative interpretation of the fuzzy inference process.

We are now ready to describe in detail our view of the approximate reasoning process as it applies to a typical fuzzy control device design. The four steps of this process, described in general, are illustrated in Fig. 5.1 for the case of a system with three inputs and two outputs.

5.4.2. Approximate Reasoning Process for Simple Fuzzy System with m Inputs and n Outputs

5.4.2.1. Step 1. Fuzzification. The goal of this step is to translate crisp input values into fuzzy linguistic values. This is quite simple to do because it simply means taking the projection of the corresponding fuzzy relation at the actual crisp input value to the system in each case. A pseudocode representation of the entire process of Step 1 is

```
For all  $i = 1, 2, \dots, m$  do
  For all  $t \in T_{xi}$  do
    Calculate  $x'_i(t) = R_{xi}(t, x_i)$ . (5.2)
```

In the particular case that TFNs are assumed, this algorithm can be stated in greater detail as:

```
For all  $i = 1, 2, \dots, m$  do
  For all  $t \in T_{xi}$  do
    Calculate  $x'_i(t) = \max\{0,$ 
 $\min[(x_i - aR_{xi}[t,1]) / (aR_{xi}[t,2] - aR_{xi}[t,1]),$ 
 $(aR_{xi}[t,3] - x_i) / (aR_{xi}[t,3] - aR_{xi}[t,2])\}.$ 
```

If the third input of some particular fuzzy control device is the fuzzy relation R_x^3 represented by Eq. 5.1 and at some instant the crisp input value $x_3 = -3$, then:

$$x'_3(t) = \begin{cases} 0.75, & t = \text{NS} \\ 0.25, & t = \text{ZO} \\ 0, & \text{elsewhere} \end{cases}$$

In other words $x'_3 = 0.75/\text{NS} + 0.25/\text{ZO}$.

5.4.2.2. Step 2. Inference. The goal of this step is to infer fuzzy linguistic output values from the linguistic input values obtained in Step 1. Having noted first, that the rules can be expressed as functions and second, that the extension principle shows how to interpret these functions relative to fuzzy variables, we can now say that all we need to do in Step 2 is calculate

$$y'_j = f_j(x'_1, x'_2, \dots, x'_m) \quad \text{for all } j = 1, 2, \dots, n$$

This statement is simple enough, but it remains rather abstract. To be a little more concrete, we recall that we actually wish to express the y'_j values by their

corresponding membership functions. This leads us to consider the following pseudocode:

```
For all  $j = 1, 2, \dots, n$  do
  For all  $t_y \in T_{yj}$  do
    Calculate  $y'_j(t_y)$ .
```

But we must still specify how to calculate these $y'_j(t_y)$ values. To address this point, we consider once again what we meant by this tabular expression of the control rules in the form of functions. If we were to go the other way for a moment, and tried to find a way to convert the functions back into collections of IF-THEN rules, then one way to do so would be to consider each point in a table as a single rather simple rule. That is for each output, the function f_j generates one rule for each point in T_x for which f_j is defined. This takes the following form:

```
IF  $x_1 = t_{x1}$  AND  $x_2 = t_{x2}$  AND . . . AND  $x_m = t_{xm}$ 
THEN  $y_j = f_j(t_x)$ 
```

As always we can consider all of the rules in the collection to be joined by disjunction: Rule_1 OR Rule_2 OR . . . There are typically several values of t_x for which $f_j(t_x)$ has the same value. When considering only one particular output term $t_y \in T_{yj}$, we ignore all rules corresponding to points where $f_j(t_x) \neq t_y$ and consider only the remaining rules as being joined by ORs. Recalling discussions on fuzzy inference in Chapter 4, if the min-max operators are used, then we must calculate $y'_j(t_y)$ as follows:

$$y'_j(t_y) = \underset{f_j(t_x)=t_y}{\text{Max}} \left[\underset{i=1}{\overset{m}{\text{Min}}} x'_i(t_{xi}) \right] \quad \text{for all } t_y \in T_{yj} \text{ and for all } j = 1, 2, \dots, n \quad (5.3)$$

This definition works fine just as it stands even if f_j is not defined for all points in T_x .

5.4.2.3. Step 3. Defuzzification, Part 1. The goal of this step is to translate fuzzy linguistic output values into fuzzy subsets of the output base variable space. Just as Step 1 was based on the fuzzy relations for the inputs, this step is based on the fuzzy relations for the outputs; however it is not quite so simple as Step 1. Given a crisp value and a fuzzy relation, we translate from one domain to the other just by taking a projection, but if we are given a fuzzy value and a fuzzy relation, we must define a form of composition. Therefore let us say that for all $j = 1, 2, \dots, n$ and for each $y_b \in Y_j$:

$$y''_j(y_b) = y'_j(\bullet) \circ R_{yj}(\bullet, y_b)$$

Under the max-min rule of fuzzy composition, this means

$$y_j''(y_b) = \underset{t_y \in T_{yy}}{\text{Max}} [\min\{y_j'(t_y), R_{yy}(t_y, y_b)\}]$$

for all $y_b \in Y_j$ and for all $j = 1, 2, \dots, n$ (5.4)

5.4.2.4. Step 4. Defuzzification, Part 2. The goal of this final step is to translate the fuzzy subsets of the reals obtained in Step 3 into the final, crisp real values. Note: In this step there is no difference between our alternative perspective and the standard one. If the COG method is applied to our notation, then for all $j = 1, 2, \dots, n$:

$$y_j = \frac{\sum_{y_b \in Y_j} [y_b \cdot y_j''(y_b)]}{\sum_{y_b \in Y_j} y_j''(y_b)}$$

rounded to the nearest value in Y_j .

This ends the actual explanation of the algorithm.

A few illustrations may be useful. Appendix B contains a Pascal program for a device with three inputs and two outputs. Here, we consider a somewhat simpler, though still quite realistic, example calculated manually.

Consider a system with only two inputs and one output. Furthermore, in each case the term set is somewhat small. To begin with the term sets, let us suppose that the designer believed that the following would suffice.

$$\begin{aligned} T_{x1} &= \{\text{NE, ZO, PS, PB}\}, \\ T_{x2} &= \{\text{NB, NS, ZO, PS, PB}\}, \\ T_y &= \{\text{NB, NS, ZO, PO}\} \end{aligned}$$

where NE = negative, PO = positive, and all other abbreviations take their standard meanings in fuzzy control.

The control rules are thus easily expressed as a two-dimensional table, only four-by-five. Let us suppose that the designer arrived at the function expressed in Table 5.2a.

In all cases TFNs are used, and the fuzzy relations for defining the linguistic variable can be expressed in the (a_1, a_2, a_3) format. Let us suppose that the designer decided on the relations thus shown in Table 5.2b. Furthermore the base variable value set is best defined as a finite set in the case of the output. Assume the designer thought it reasonable for this set to be

$$Y = \{-12, -11, -10, \dots, 5, 6, 7\}$$

There is no need to define explicitly X_1 and X_2 in this way.

Now suppose that at a given point in time, system inputs are $x_1 = -2$ and $x_2 = 7$. What is the final (crisp) output?

Table 5.2. Control Rules (Crisp Function) for a Numerical Example

		t_{x2}					
		NB	NS	ZO	PS	PB	
$f(t_{x1}, t_{x2})$		NE	PO	PO	ZO	ZO	NS
t_{x1}		ZO	PO	ZO	ZO	NS	NS
PS		ZO	ZO	NS	NS	NB	NB
PB		NS	NS	NB	NB	NB	NB

Step 1: $x'_1 = 0.5/\text{NE} + 0.5/\text{ZO}$
 $x'_2 = 0.25/\text{PS} + 0.75/\text{PB}$

Step 2: $y'(\text{NB}) = \max[\min(0, 0.75), \min(0, 0), \min(0, 0.25), \min(0, 0.75)]$
 $= 0$
 $y'(\text{NS}) = \max[\min(0.5, 0.75), \min(0.5, 0.25), \min(0.5, 0.75), \min(0, 0), \min(0, 0.25), \min(0, 0), \min(0, 0)]$
 $= 0.5$
 $y'(\text{ZO}) = \max[\min(0.5, 0), \min(0.5, 0.25), \min(0.5, 0), \min(0.5, 0), \min(0, 0), \min(0, 0)]$
 $= 0.25$
 $y'(\text{PO}) = \max[\min(0.5, 0), \min(0.5, 0), \min(0.5, 0)]$
 $= 0$

Table 5.3. Fuzzy Relations for Variables in
Table 5.2

R_{x1}	R_{x2}
NE : $(-\infty, -4, 0)$	NB : $(-\infty, -7, -4)$
ZO : $(-4, 0, 3)$	NS : $(-7, -4, 0)$
PS : $(0, 3, 8)$	ZO : $(-4, 0, 4)$
PB : $(3, 8, \infty)$	PS : $(0, 4, 8)$
	PB : $(4, 8, \infty)$
R_y	
NB : $(-\infty, -10, -5)$	
NS : $(-10, -5, 0)$	
ZO : $(-5, 0, 5)$	
PO : $(0, 5, \infty)$	

Thus, $y' = 0.5/NS + 0.25/ZO$

Step 3: $y''(-12) = \max[\min(0, 1), \min(0.5, 0), \min(0.25, 0), \min(0, 0)] = 0$
 $y''(-11) = \max[\min(0, 1), \min(0.5, 0), \min(0.25, 0), \min(0, 0)] = 0$
 $y''(-10) = \max[\min(0, 1), \min(0.5, 0), \min(0.25, 0), \min(0, 0)] = 0$
 $y''(-9) = \max[\min(0, 0.8), \min(0.5, 0.2), \min(0.25, 0), \min(0, 0)] = 0.2$
 $y''(-8) = \max[\min(0, 0.6), \min(0.5, 0.4), \min(0.25, 0), \min(0, 0)] = 0.4$
 $y''(-7) = \max[\min(0, 0.4), \min(0.5, 0.6), \min(0.25, 0), \min(0, 0)] = 0.5$
 $y''(-6) = \max[\min(0, 0.2), \min(0.5, 0.8), \min(0.25, 0), \min(0, 0)] = 0.5$
 $y''(-5) = \max[\min(0, 0), \min(0.5, 1), \min(0.25, 0), \min(0, 0)] = 0.5$
 $y''(-4) = \max[\min(0, 0), \min(0.5, 0.8), \min(0.25, 0.2), \min(0, 0)] = 0.5$
 $y''(-3) = \max[\min(0, 0), \min(0.5, 0.6), \min(0.25, 0.4), \min(0, 0)] = 0.5$
 $y''(-2) = \max[\min(0, 0), \min(0.5, 0.4), \min(0.25, 0.6), \min(0, 0)] = 0.4$
 $y''(-1) = \max[\min(0, 0), \min(0.5, 0.2), \min(0.25, 0.8), \min(0, 0)] = 0.25$
 $y''(0) = \max[\min(0, 0), \min(0.5, 0), \min(0.25, 1), \min(0, 0)] = 0.25$
 $y''(1) = \max[\min(0, 0), \min(0.5, 0), \min(0.25, 0.8), \min(0, 0.2)] = 0.25$
 $y''(2) = \max[\min(0, 0), \min(0.5, 0), \min(0.25, 0.6), \min(0, 0.4)] = 0.25$
 $y''(3) = \max[\min(0, 0), \min(0.5, 0), \min(0.25, 0.4), \min(0, 0.6)] = 0.25$
 $y''(4) = \max[\min(0, 0), \min(0.5, 0), \min(0.25, 0.2), \min(0, 0.8)] = 0.2$
 $y''(5) = \max[\min(0, 0), \min(0.5, 0), \min(0.25, 0), \min(0, 1)] = 0$
 $y''(6) = \max[\min(0, 0), \min(0.5, 0), \min(0.25, 0), \min(0, 1)] = 0$
 $y''(7) = \max[\min(0, 0), \min(0.5, 0), \min(0.25, 0), \min(0, 1)] = 0$
 Thus, $y'' = 0.2/-9 + 0.4/-8 + 0.5/-7 + 0.5/-6 + 0.5/-5 + 0.5/-4 + 0.5/-3 + 0.4/-2 + 0.25/-1 + 0.25/0 + 0.25/1 + 0.25/2 + 0.25/3 + 0.2/4$

$$\text{Step 4: } y = \text{Rounded} \left(\frac{-16.25}{4.95} \right) = \text{Rounded} \left(\frac{-3.28}{y} \right)$$

$$= -3.$$

Thus the final output is $y = -3$.

There are two other points about this process that are significant, but which can be discussed very briefly. First in nearly all applications, this process is implemented based on a set time cycle; thus time is effectively dealt with as a discrete variable. Naturally the length of the time cycle depends on the reaction time requirements of the application; it must also fall within the capabilities of the hardware used. Second unlike the typical expert system application, fuzzy control designs, especially in the form presented here, make no use of rule chaining, and therefore there is no need to think about the problems of backward or forward chaining as they relate to fuzzy inference. This is part of what we mean by a simple fuzzy system.

5.5. Comparison of Our Perspective with the Standard One

We refer to the approach presented in the preceding section as an alternative perspective or alternative interpretation of the fuzzy inference process. We now describe how this alternative interpretation differs from the standard interpretation of fuzzy inference or approximate reasoning.

Although there is considerable variety in the literature on fuzzy systems, discussions of the fundamental concepts and rationale of approximate reasoning nearly always center around the basic conceptual structure first fully articulated by Mamdani and his colleagues in early papers on fuzzy control [see (Mamdani and Assilian, 1975)]. The popularity of that conceptual structure may also owe something to several papers by Zadeh (see (1975)] on a concept for linguistic variables that in many ways complemented the Mamdani perspective. Certainly many fuzzy control designs differ from the Mamdani approach in terms of their details, and there is also some debate on whether it is logically sound to use the term fuzzy implication as Mamdani does, but it is nonetheless true that most discussions of the basic concepts for approximate reasoning are structured around the presentation of Mamdani *et al.* Therefore we refer to that as the standard interpretation.

On the other hand, we make no claim that our alternative perspective is altogether unique. Throughout the history of fuzzy control and fuzzy systems research, there have been a few perspectives that have differed fundamentally from the standard one. Holmblad and Østergaard [see (1982)], in describing their control design for the cement kiln, presented a very straightforward description of the principles of fuzzy control, centering on a “grade of fulfillment [sic]” concept for the rules. More interesting still Sugeno in many of his earlier overviews of fuzzy

control, especially those in Japanese, such as (Sugeno, 1987), suggested not one, but three different fuzzy inference methods (*suironhou*). All three of these are presented in a manner that is quite straightforward and appealing to an intuitive appreciation of the concepts of approximate reasoning. Presentations in the spirit of Holmblad and Østergaard or Sugeno offer instructional alternatives that are arguably much easier for a beginning student to understand than the standard perspective; however these presentations are typically stated only informally or semiformal, and also they rely perhaps too heavily on an intuitive discussion of what one means by linguistic rules.

We designed the perspective that we presented in Section 5.4 to capture the intuitive appeal and ease of understanding found in the presentations just described, but in a manner that is somewhat more formally stated. We also believe that this perspective makes the most rational applications of such basic fuzzy set concepts as fuzzy relations, the extension principle, fuzzy composition, etc. to the approximate reasoning process.

Since the purpose of this section is to compare the alternative perspective to the standard one, the remainder of this section can be divided roughly into three parts. First, we restate the standard interpretation by our notation, and also restate the alternative interpretation in a similarly packaged format. Second, we prove that under reasonable assumptions, the two interpretations lead to identical results. And third, we argue that the alternative interpretation naturally leads to both greater clarity of thought and more efficient implementations than the standard one. Although complete mastery of this section is not essential for understanding the remainder of the book, all readers will find it worthwhile at least to reflect on its main points, and some may find it quite interesting.

5.5.1. Standard Perspective as Expressed in Our Notation

In one part of Section 4.5, we considered briefly the nature of the standard interpretation of fuzzy inference, but we focus there only on one rule at a time and discuss the simplified case of only one input variable. Our discussion began by interpreting generalized *modus ponens* by what was called the compositional rule of inference. That in turn was interpreted according to both what is sometimes called the Mamdani interpretation of implication and the max-min rule of fuzzy composition. The result of all of this was Eq. 4.17. This is a good starting point for considering how to state the standard interpretation in the notation used in the previous section. For convenience we repeat Eq. 4.17 here. The only change we make to it is that we now replace the μ -form with the abbreviated form for indicating a membership function:

$$B'(y) = \underset{x \in X}{\text{Max}} \{\min[A'(x), A(x), B(y)]\} \quad (5.5)$$

If for the moment we consider a simple system with only one input, then T_x can be taken to mean a simple set of linguistic terms, and $t \in T_x$ is a single term. If there is also only one output, then all of the rules can be summarized by a single function $f: T_x \rightarrow T_y$, where T_y is also a simple set of terms. Plainly then there are at most $|T_x|$ rules, each of the form:

IF t THEN $f(t)$

where $t \in T_x$.

If we start simply by considering only one rule at a time, then let us fix on one particular $t \in T_x$ for which $f(t)$ is defined. The generalized *modus ponens* can then be stated as:

$$\begin{array}{c} t \Rightarrow f(t) \\ \text{more or less } t \\ \hline \text{more or less } f(t), \end{array}$$

This can be interpreted in a form similar to Eq. 5.5. This equation produces a fuzzy subset of the base output space. This corresponds in our notation to the value of y'' . To avoid confusion with the use in our notation of y to indicate the crisp output, let us replace the y in Eq. 5.5 with y_b . Thus the left-hand side of Eq. 5.5 can be interpreted as $y''(y_b)$.

The most difficult part of doing this is to determine what the $A'(x)$ term in Eq. 5.5 really means. First we recall that the x in Eq. 5.5 is not the actual crisp input value, as it was in our notation, but rather it is the index for the maximization, and it ranges through the base input space; thus we use the symbol x_b instead of x . However this still does not tell us what $A'(x_b)$ means when A' is an indication of the fact that more or less A . This can be interpreted as:

$$\min\{A(x_b), A(x)\} \quad \text{for all } x_b \in X$$

where x is the actual input. But clearly:

$$\begin{aligned} & \min[\min\{A(x_b), A(x)\} A(x_b), B(y_b)] \\ &= \min[A(x), A(x_b), B(y_b)] \end{aligned}$$

Now the connection between our approach of representing a linguistic variable by a fuzzy relation and the standard approach of representing one by a collection of fuzzy sets and associated terms is a fairly minor point. The transformation can be represented as:

$$R_x(t_x, x_b) = t_x(x_b)$$

and similarly for the output variables. Thus Eq. 5.5 can be restated as:

$$y''(y_b) = \max_{x_b \in X} (\min[R_x(t, x), R_x(t, x_b), R_y(f(t), y_b)]) \quad \text{for all } y_b \in Y$$

However so far this applies only when just one rule is considered. In this one-input-one-output case, the entire rule base can easily be generated by allowing t to vary over T_x , for all values where $f(t)$ is defined. As always the rules are considered to be joined by disjunction, and disjunction is represented here by taking the maximum. So to take all rules together, we can write the following if we add the convention that $R_y(f(t), \bullet) = 0$ if $f(t)$ is undefined.

$$y''(y_b) = \max_{t \in T_x} \left\{ \max_{x_b \in X} (\min[R_x(t, x), R_x(t, x_b), R_y(f(t), y_b)]) \right\} \quad \text{for all } y_b \in Y \quad (5.6)$$

To extend this definition to m inputs and n outputs requires the following changes:

- T_x is now viewed as the Cartesian product $T_x = T_{x1} \times T_{x2} \times \dots \times T_{xm}$, where T_{xi} is the term set for the i th input.
- t_x is now a vector $(t_{x1}, t_{x2}, \dots, t_{xm})$.
- X is now the Cartesian product $X_1 \times X_2 \times \dots \times X_m$, that is, the Cartesian product of the individual base input spaces.
- x_b is now the vector $(x_{b1}, x_{b2}, \dots, x_{bm})$.
- $\min[R_x(t, x), R_x(t, x_b)]$ is replaced with

$$\min_{i=1}^m \{\min[R_{xi}(t_{xi}, x_i), R_{xi}(t_{xi}, x_{bi})]\}$$

- The outputs can still be considered one at a time for $j = 1, 2, \dots, n$. However we must now specify y_j'' instead of y'' , f_j instead of f , and R_{yj} instead of R_y . Also y_b is now considered an element of Y_j for whatever j we are considering at the moment.

With all of these changes, Eq. 5.6 takes the following form:

$$y_j''(y_b) = \max_{t_y \in T_{yj}} \left\{ \max_{x_b \in X} \left[\min \left(\min_{i=1}^m \{\min[R_{xi}(t_{xi}, x_i), R_{xi}(t_{xi}, x_{bi})]\}, R_{yj}[f_j(t_x), y_b] \right) \right] \right\}$$

for all $y_b \in Y$ and for all $j = 1, 2, \dots, n$ (5.7)

where $R_{yj}(\text{undefined}, \cdot) = 0$.

Let us refer to the right-hand side in Eq. 5.7 as $S_j(x, y_b)$ to represent the results for $y''(y_b)$ for a given $j \in \{1, 2, \dots, n\}$ and a given $y_b \in Y_j$ under the standard

interpretation when the system (crisp) inputs are x . The crisp output values can be calculated from this by the COG or some similar method.

5.5.2. A Similarly Packaged Format for the Alternative Perspective

As another preliminary to proving that the two interpretations are equivalent in results, it is useful to present the alternative perspective in a form similar to what the $S_j(x, y_b)$ form represents for the standard interpretation. Because $S_j(x, y_b)$ is a form for computing y_j'' values from x_i values, this is equivalent to combining Steps 1–3 in our interpretation. Note: We generally avoid combining steps because it reduces the clarity of the approach, but we can easily perform this combination when it is necessary to do so just by manipulating Eqs. 5.2–5.4. From Eqs. 5.3 and 5.4, we have

$$y_j''(y_b) = \underset{t_y \in T_{yj}}{\text{Max}} \left\{ \underset{f_j(t_x) = t_y}{\text{Max}} \left[\underset{i=1}{\overset{m}{\text{Min}}} x'_i t_{xi} \right], R_{yj}(t_y, y_b) \right]$$

for all $y_b \in Y_j$ and for all $j = 1, 2, \dots, n$

By Eq. 5.2 this results in:

$$y_j''(y_b) = \underset{t_y \in T_{yj}}{\text{Max}} \left\{ \underset{f_j(t_x) = t_y}{\text{Max}} \left[\underset{i=1}{\overset{m}{\text{Min}}} R_{xi}(t_{xi}, x_i) \right], R_{yj}(t_y, y_b) \right\}$$

$$\text{for all } y_b \in Y_j \quad j = 1, 2, \dots, n \quad (5.8)$$

Let $A_j(x, y_b)$ represent the right-hand side of Eq. 5.8.

5.5.3. Equivalence of Results

We are now ready to begin proving that $S_j(x, y_b)$ equals $A_j(x, y_b)$ and thus that the two interpretations yield equivalent results. We begin by presenting a lemma.

Lemma. *Given any two sets X and Y and functions:*

$$f : X \rightarrow Y$$

$$g : X \rightarrow \mathcal{R}^+$$

$$h : Y \rightarrow \mathcal{R}^+$$

where \mathcal{R}^+ denotes the nonnegative reals. The following is true:

$$\underset{x \in X}{\text{Max}} [\min\{g(x), h(f(x))\}] = \underset{y \in Y}{\text{Max}} \left[\min \left\{ \underset{f(x)=y}{\text{Max}} g(x), h(y) \right\} \right]$$

Proof. Define function $F : X \times Y \rightarrow \mathcal{R}^+$ as:

$$F(x, y) = \begin{cases} \min[g(x), h(y)], & \text{where } f(x) = y \\ 0, & \text{where } f(x) \neq y \end{cases}$$

Then:

$$\underset{x \in X}{\text{Max}} [\min\{g(x), h(f(x))\}] = \underset{x \in X}{\text{Max}} [F(x, f(x))]$$

because there are no terms for which $y \neq f(x)$. For any given $x \in X$, $F(x, y)$ clearly attains its maximum where $f(x) = y$ by the definition of F . Thus:

$$\begin{aligned} \underset{x \in X}{\text{Max}} [F(x, f(x))] &= \underset{x \in X}{\text{Max}} \left[\underset{y \in Y}{\text{Max}} \{F(x, y)\} \right] \\ &= \underset{y \in Y}{\text{Max}} \left[\underset{x \in X}{\text{Max}} \{F(x, y)\} \right] \\ &= \underset{y \in Y}{\text{Max}} \left[\underset{f(x)=y}{\text{Max}} \{\min[g(x), h(y)]\} \right] \\ &= \underset{y \in Y}{\text{Max}} \left[\min \left\{ \underset{f(x)=y}{\text{Max}} g(x), h(y) \right\} \right] \end{aligned}$$

■

Now we are ready for the proof of equivalence of results.

Theorem. Given a crisp input vector $\mathbf{x} = (x_1, x_2, \dots, x_m)$ such that $x_i \in X_i$ for all $i = 1, 2, \dots, m$, then for any given $j \in \{1, 2, \dots, n\}$ and any given $y_b \in Y_j$, the following is true:

$$S_j(\mathbf{x}, y_b) = A_j(\mathbf{x}, y_b)$$

where functions S_j and A_j are as defined in Eqs. 5.7 and 5.8, respectively.

Proof. Assume for the moment that we fix on only one particular $t_x \in T_x$, which means that t_{xi} is fixed for all $i = 1, 2, \dots, m$. Therefore:

$$\text{let } g_i(x_i) = R_{xi}(t_{xi}, x_i) \quad \text{for all } i = 1, 2, \dots, m$$

Also if we are assuming that t_x is fixed, then $R_{yj}(f_j(t_x), y_b)$ becomes a constant so label it C . Thus we can rewrite

$$\underset{x_b \in X}{\text{Max}} \left[\min \left(\underset{i=1}{\overset{m}{\text{Min}}} \{ \min[R_{xi}(t_{xi}, x_i), R_{xi}(t_{xi}, x_{bi})] \}, R_{yj}(f_j(t_x), y_b) \right) \right]$$

as:

$$\begin{aligned} & \underset{x_b \in X}{\text{Max}} \left[\min \left(\underset{i=1}{\overset{m}{\text{Min}}} \{ \min[g_i(x_i), g_i(x_{bi})] \}, C \right) \right] \\ &= \min \left[\underset{x_b \in X}{\text{Max}} \underset{i=1}{\overset{m}{\text{Min}}} \{ \min[g_i(x_i), g_i(x_{bi})] \}, C \right]. \end{aligned} \quad (5.9)$$

Because we assumed that $x_i \in X_i$ for all $i = 1, 2, \dots, m$, clearly there is one $x_b \in X$ for which $x_b = x$. Therefore:

$$\underset{x_b \in X}{\text{Max}} \left(\underset{i=1}{\overset{m}{\text{Min}}} \{ \min[g_i(x_i), g_i(x_{bi})] \} \right) \geq \underset{i=1}{\overset{m}{\text{Min}}} g_i(x_i) \quad (5.10)$$

But it is also obvious that:

$$\min[g_i(x_i), g_i(x_{bi})] \leq g_i(x_i) \quad \text{for all } i = 1, 2, \dots, m$$

and so:

$$\underset{x_b \in X}{\text{Max}} \left(\underset{i=1}{\overset{m}{\text{Min}}} \{ \min[g_i(x_i), g_i(x_{bi})] \} \right) \leq \underset{i=1}{\overset{m}{\text{Min}}} g_i(x_i) \quad (5.11)$$

Obviously, the two inequalities, 5.10 and 5.11 imply an equality, and applying this equality to Eq. 5.9 results in:

$$\begin{aligned} & \underset{x_b \in X}{\text{Max}} \left[\min \left(\underset{i=1}{\overset{m}{\text{Min}}} \{ \min[g_i(x_i), g_i(x_{bi})] \}, C \right) \right] \\ &= \min \left[\underset{i=1}{\overset{m}{\text{Min}}} g_i(x_i), C \right] \end{aligned}$$

Reinterpreting this and substituting back into the definition of $S_j(\mathbf{x}, y_b)$ for all $t_x \in T_x$ yields:

$$S_j(\mathbf{x}, y_b) = \underset{t_x \in T_x}{\text{Max}} \left\{ \min \left[\underset{i=1}{\overset{m}{\text{Min}}} R_{xi}(t_{xi}, x_i), R_{yj}(f_j(t_x), y_b) \right] \right\}.$$

With the following substitutions:

Lemma Form	Equivalent Form
X	$T_x = T_{x1} \times T_{x2} \times \dots \times T_{xm}$
Y	T_y
$x \in X$	$t_x = (t_{x1}, t_{x2}, \dots, t_{xm}) \in T_x$
$y \in Y$	$t_y \in T_y$
$f(x)$	$f_j(t_x, y_b)$ because y_b is fixed
$g(x)$	$\underset{i=1}{\overset{m}{\text{Min}}} \{R_{xi}(t_{xi}, x_i)\}$ because x is fixed
$h(y)$	$R_{yj}(t_y, y_b)$ because y_b is fixed

We know by the lemma that:

$$\begin{aligned} & \underset{t_x \in T_x}{\text{Max}} \left\{ \min \left[\underset{i=1}{\overset{m}{\text{Min}}} R_{xi}(t_{xi}, x_i), R_{yj}(f_j(t_x), y_b) \right] \right\} \\ &= \underset{t_y \in T_y}{\text{Max}} \left[\min \left\{ \underset{f_j(t_x)=t_y}{\text{Max}} \left[\underset{i=1}{\overset{m}{\text{Min}}} R_{xi}(t_{xi}, x_i) \right], R_{yj}(t_y, y_b) \right\} \right] \\ &= A_j(\mathbf{x}, y_b) \end{aligned}$$

■

Directly speaking, the only assumption that is not completely obvious is that $x_i \in X_i$ for all $i = 1, 2, \dots, m$. The alternative interpretation does not require us to define input base variable spaces explicitly, but the standard interpretation does. When comparing the two this becomes a significant point because this assumption is used in our proof. For example suppose that in designing a fuzzy control device according to the standard interpretation, the designer decides that a particular input variable i can be expected to range only from -15 to 15 with precision of measurement 0.5 so that in the design X_i can be assumed to be $\{-15.0, -14.5, -14.0, \dots, 14.5, 15.0\}$. Then, strictly speaking, in order for the two interpretations to be equivalent in results, we must assume that the actual input value x_i is a member of this definition of X_i . However as a practical matter, this assumption has little bearing. In defining X_i the designer made certain assumptions about the range and precision of measurement of the input, and if those assumptions are valid, then it is also valid that no harm would come from always rounding the actual input value to the closest value in X_i before processing it.

But indirectly speaking, a much more significant assumption is implicit here. In proving that $A_j(\mathbf{x}, y_b) = S_j(\mathbf{x}, y_b)$, we show that the two approaches are equivalent in results only when classical (min-max) operators and the Mamdani interpretation of implication are assumed. To prove these results in a more general way would be a much more difficult task which will not be attempted here, partly because we intend to use the classical operators throughout the remainder of the book anyway.

5.5.4. Conceptual Comparison of the Two Perspectives

Finally, we explain why we believe the alternative interpretation leads to greater clarity of thinking than the standard one. Certainly, this is a subjective matter, but we consider the following points:

- Under the standard interpretation, we are asked to think of the control rules as acting on and producing entities that are very difficult to grasp conceptually. Under the alternative interpretation, it is only necessary to think of the control rules as acting on fuzzy subsets of small sets of terms, and this is very easy to do provided that one thoroughly understands the extension principle.
- To make matters even much worse, it is customary under the standard interpretation to combine all of the input linguistic variable definitions, all of the control rules, and the output linguistic variable definitions by composition, and it is often treated in this way even in introductory material. This way of thinking seems suited only to those who are so hurried to obtain results that they cannot reflect on what they are doing. However proceeding in this manner draws attention from the preceding problem. The alternative interpretation reduces the process to comprehensible steps with comprehensible outputs from each step.
- If we take the standard interpretation literally, it leads to computer implementations much less efficient than they could be. The alternative interpretation leads us to think naturally of very efficient implementations.

Considering the first point in greater detail, in the standard interpretation of fuzzy inference, control rules act on vectors of unions of sets of truncations (restrictions) of fuzzy numbers. We emphasize this: Control rules do not act on fuzzy numbers nor on truncations of fuzzy numbers nor on sets of truncations of fuzzy numbers nor on unions of sets of truncations of fuzzy numbers; these act on vectors of unions of sets of truncations of fuzzy numbers. And these produce similar entities as outputs. This is a much more difficult way of approaching the problem than ours.

Concerning our last point, applying control rules under the alternative interpretation as expressed in Eq. 5.3 suggests that a program for implementing fuzzy inference must loop through the T_x . However we must remember that with the

tabular representation of the control rules, this has the effect of looping through the rules. At no point does the alternative interpretation suggest that looping through the Cartesian product of the input base variable value sets, and in fact it does not even require us to define those sets. The standard interpretation expressed in the $S_j(x, y_b)$ form suggests that we must loop through the Cartesian product of the input base variable value sets as well as through T_x . Strictly speaking this implies that the algorithm is of exponential order in time on the number of input variables in either case, but it is much more serious in the case of the standard interpretation.

In light of these problems, why is preference generally given to the standard interpretation? Why in particular does the literature place so much emphasis on calculation over the base variable value space throughout the entire reasoning process? It is possible to find justifications for this: Berenji (1992) discusses modeling noise in sensor readings by fuzzy numbers too, in addition to fuzzy numbers representing vagueness in terms. This justifies the necessity of calculating over the base variable space, but this idea is very much a new twist, not at all a part of the classical concepts of early fuzzy control. Incidentally we question this idea; why is it more appropriate to model stochastic phenomena by fuzzy sets than to model vagueness of concept by probability theory?

Perhaps part of the real reason is that fuzzy control was developed mostly by people trained primarily as engineers, and such training often produces a strong bias toward quantification, a tendency to believe that all analytical reasoning must relate to real or complex number spaces. Early formal presentations of the concept of linguistic variables referred to “a semantic rule” associating linguistic values with their meanings, which are fuzzy subsets of the base variable universe. This is straying again into the philosophical realm, and no disrespect is intended, but giving meaning is not equivalent to quantifying (either fuzzily or crisply), though it is possibly typical for someone with an engineering mindset to believe it is. Before Gabriel Daniel Fahrenheit introduced his standardized scale in the early eighteenth century, before even the development of thermometers in the seventeenth century, the statement, “It’s a hot day today,” had meaning.

Fuzzy set theory advocates sometimes state that traditional AI has been concerned with symbolic processing to a fault. Perhaps this results in a tendency to think of symbolic processing negatively and to express applications of fuzzy set theory so that they seem to be completely unrelated to symbolic processing, though actually they are very much related.

5.6. General Notes on Implementation

In the present age, the age of digital electronics, there is nothing particularly difficult about implementing fuzzy control designs even with fairly standard components. There is something just slightly ironic about this. Fuzzy control design

is based on fuzzy set theory and a nonbinary view of logic, in fact on a continuous view of logic. Yet fuzzy control implementations are normally based primarily on digital technology, and it is difficult to imagine these implementations using strictly analog electronics. On the other hand, conventional control designs are based on conventional mathematics, which in turn is based on conventional two-valued logic. Yet conventional control theory reached its mature stage of development during the age of analog electronics, and many of the conventions of conventional control still make the most sense when viewed from an analog perspective.

Until the mid or late 1980s, most applications of fuzzy control outside of the laboratory were to large-scale, high-cost systems, such as industrial plants or transportation systems, where it was economical to connect a dedicated minicomputer or even a mainframe computer to the system. In most cases such a computer was already there to implement digital process control, data logging, and so on. By the late 1980s when Japanese industry in particular began to investigate the possibility of implementing fuzzy control in small, low cost systems, such as consumer appliances, microprocessor technology was already highly developed and providing quite a high degree of calculation power at a very low cost. Thus fuzzy control design as a technology did not develop in the absence of powerful digital computing devices.

Digital-based control applications also require sensors and actuators and interfaces for connecting these to digital-computing device, but these technologies were also either already highly developed or developing in parallel with the microprocessor technology. Furthermore the costs of the various other types of devices had also fallen dramatically.

One issue that has not been totally resolved is the question of how much special purpose hardware and software are useful for the further development of fuzzy control technology. On the one hand much of the development to date has been based only on the programming of general purpose hardware. For the typical application there is nothing particularly difficult about doing this if we are capable of programming a microprocessor and clearly understand either the standard or the alternative interpretation of the fuzzy inference process. Furthermore if we use as efficient an algorithm as possible and there are no exceptionally stringent reaction-time requirements implicit in the application, then general purpose hardware is typically fast enough for most applications developed to date.

In fact, one could argue that one of the advantages of fuzzy control and fuzzy expert system approaches is that they are probably generally less dependent on the development of special purpose hardware than are neural net approaches. It seems likely that the future of low cost, high quality voice recognition or handwriting recognition will depend more on the development of low cost, massively parallel hardware designed on a special purpose basis than on the further refinement of neural net algorithms. It should be viewed as an advantage that applications of

approximate reasoning by fuzzy inference are not constrained by a similar hardware bottleneck.

However, there seems to be increasing interest in both special purpose hardware and special purpose software for fuzzy control development, and there are certain arguments one could make in support of this development. In the case of the special purpose fuzzy hardware, one of the strongest arguments is that it probably extends the potential for further development of fuzzy control in application areas where very quick reaction times are necessary. One specific example of such an area that appeared in a European survey presented by (Kasper and Zimmermann, 1994) is propulsion engineering.

For this and other reasons, there is considerable interest in North America, Japan, and Europe in developing fuzzy inference chips, as reported for example in (Johnson, 1994). There is even a term for measuring the performance of these chips, for mega fuzzy logic instructions (or inferences) per second (MFLIPS), but unfortunately standards for how a chip is actually measured in this regard remain somewhat ambiguous.

Another question is whether a fuzzy inference chip should be totally digital or primarily analog in the basis of its design. Analog chips, as surveyed by (Yamakawa, 1994), are interesting because analog signals seem a more natural way to represent membership grades and because in some applications, they may eliminate the need for analog/digital converters in connection with the sensors or digital/analog converters with the actuators. However analog fuzzy inference chips have the same drawbacks as other analog computing device: There are practical limits to their level of precision, it is difficult to develop general formats for programming them, and their accuracy may be influenced by ambient conditions. For these reasons there continues to be more interest in digital fuzzy inference chips, as surveyed by (Togai, 1994).

There are two major justifications for special purpose software tools for developing fuzzy control devices. First such tools lower the level of both programming skill and systems-design experience required to develop fuzzy control devices. Second these tools often have well-developed features for assisting in the difficult process of tuning a fuzzy control device. For some time now such software has been developed specifically for use in conjunction with a particular brand of special-purpose fuzzy inference hardware. Examples can be found in (Chiu and Togai, 1988), as well as the more recent surveys on fuzzy inference chips previously mentioned. In a slightly different vein, there is also now Computer Assisted Instruction (CAI) software for developing an understanding of how to design a fuzzy control device; some of the packages developed by CSD, Ltd., of Kawasaki and the disks that accompany many of the recent books on fuzzy logic are good examples.

There are primarily two disadvantages to using special-purpose hardware or software. While cost for such products are dropping, they may still be significant

in some applications and development environments. Another disadvantage is that even though developers of such products sometimes try to make them adaptable, there are always limitations and using a special purpose product always greatly reduces the freedom one can exercise in design. Some chips or software assume only triangular fuzzy numbers, which is fine if we intended to use TFNs anyway, otherwise it is a bothersome constraint. Other chips require symmetrical TFNs. If the chip is based on the classical max-min operators and we wish to use some other interpretation, then there is a more significant problem. Such problems are always good reasons for going back to doing things “from scratch” using general purpose hardware and software, provided that one is capable of doing that and the application requirements permit it.

5.7. Tuning of Fuzzy Control Devices

As related in Chapter 2, conventional control theory does have its drawbacks, but it also has some very significant strengths. One of these strengths is that, assuming it really is possible to develop an accurate model of the plant, conventional control theory provides highly developed techniques for analyzing how a given control device design can be expected to perform before it is even built. Once built, if the control device does not perform up to expectations according to some criterion, then conventional control theory allows us to analyze how much the device must be modified in order to measure up. Beyond any doubt, these are extremely powerful and useful capabilities. They make the development process much more structured and economical than it would otherwise be.

It is difficult for fuzzy control to compete with such advantages, although it does offer others. This is particularly true of classical approaches to fuzzy control, based almost entirely on the heuristic knowledge of human sources, but it is also true to a large degree of even more recent approaches. To say that the fuzzy control design process is based largely on heuristics and therefore is at least to some degree an *ad hoc* process is not to condemn it in any general sense. However it is always difficult to know how well a heuristic-based design will work until we actually try to implement it. It may also not be entirely obvious how to modify such a design if it fails to function as well as expected.

If the initial design process is *ad hoc*, then how much more *ad hoc* is the tuning process likely to be? Designs based on heuristics are not arbitrary, but they are necessarily subjective, and if there is an error somewhere in a collection of subjective judgments, then how do we diagnose it? These questions are quite difficult to answer, particularly in a general sense.

But as ominous as this may sound, there is no need to be excessively pessimistic about fuzzy control. Even without developing special new tools, two points typically keep the problems within bounds. First experience has shown, as attested

throughout the now vast literature on the subject, that fuzzy control designs tend to be naturally highly robust. There is a tendency for the overall performance to be very insensitive to minor variations or minor errors in the design. In a sense we expect this: Most decisions in daily life are based on subjective judgments, yet most decisions, most of the time somehow turn out to be generally acceptable. This point slightly reduces anxiety over how well the initial design will function once built. It may also reduce the need for any major amount of tuning.

The second point is that many fuzzy control devices are not so complex that diagnosis becomes a truly major undertaking. If there is a particular aspect of performance that is not as good as it should be, then it is typically possible by a powerful combination of intuition and informal analysis to determine what modification could be made to correct the problem. Unless there is something fundamentally wrong with the high-level system design, the modifications would take the form of adjustments to the rule base or to the linguistic variable definitions. Admittedly changes to one part of a complex system have the potential of damaging performance in other unforeseen ways; therefore extensive additional testing is obviously necessary. Once again the natural robustness of fuzzy control designs typically prevent extreme difficulties.

Furthermore, some special tools have been developed both for predicting performance and for the tuning of control devices designed by fuzzy control methods (see Chapters 6 and 11; probably even more of these tools will be developed in the future. The most significant trend has been to incorporate various machine-learning techniques into the fuzzy control design process with generally good results.

Tuning methods and methods of predicting performance continue to be the weakest aspects of the fuzzy control design process, especially when compared to conventional control. However these problems are not debilitating, and in many cases they are more than compensated for by the many relative strengths of fuzzy control methods.

5.8. A Few Examples of Classical Fuzzy Control Applications

Over the past 10 years, and particularly last five, the number of fuzzy control applications has increased so significantly that it is no longer feasible to produce anything close to a comprehensive survey; even a research committee on “Fuzzy Logic and Knowledge/Information Processing,” one of the fuzzy research groups sponsored by Japanese industry, abandoned its attempt. However we consider briefly some of the most historically significant fuzzy control applications, either by specific examples or by general classes, in this section. Because Chapter 5 discusses the classical approach to fuzzy control, we focus as much as possible on

designs that more or less followed that approach while at the same time giving as broad a view as possible of the overall scope of applications.

5.8.1. Model Steam Engines and the Pioneers at Queen Mary College

There is nearly universal agreement that the first actual implementation of fuzzy control should be credited to the work of Mamdani and others at Queen Mary College in London during the early 1970s. Many already highly developed concepts of fuzzy control originated from their work.

As their model plant Mamdani *et al.* took a small steam engine whose boiler pressure and output speed they hoped to control by adjusting boiler heat input and throttle adjustment. The actual fuzzy control device inputs were called:

- PE (pressure error),
- SE (speed error),
- CPE (change in pressure error), and
- CSE (change in speed error).

They called the control device outputs:

- HC (heat change) and
- TC (throttle change).

Thus they used a heuristic equivalent of PD control, though surprisingly they did not use that term. The number of terms in the term sets for each variable ranged from 5–8, and the terms were already of the nature of NB, PS, etc. The base variable value sets were taken to be discrete, with cardinalities ranging from 5–15.

The rule base, in the form stated, contains 24 IF-THEN rules, 15 for the “heater algorithm” and nine for the “throttle algorithm.” The rule format uses the convention that the antecedent of each rule must mention all four input variables, but the researchers use disjunction and logical negation within the rules as well as the term ANY. They implemented actual control with a computer connected to the steam engine and also implemented what they called a fixed digital controller by the same means for purposes of comparison. Their results show that the fuzzy controller reacted much more rapidly and stably than the other digital controller.

Although Mamdani published several papers about that time, alone and with various coauthors, explaining many different views of fuzzy control, the paper considered the best overview of the fuzzy control implementation itself is (Mamdani and Assilian, 1975), which presents eloquent statements of the rationale for developing a fuzzy control methodology. In the introduction the authors write

To the control engineer quantitative languages supporting arithmetic are the natural ones. To support the translation of vaguer, non-numeric statements that might be made about a control strategy we needed a semi-quantitative calculus.

Zadeh's (1973) fuzzy logic seemed to provide a means of expressing linguistic rules in such a form that they might be combined into a coherent control strategy.

In the conclusion they note, "The power of the approach derives from the fact that it is possible to translate into an algorithm an entirely unstructured set of heuristics expressed linguistically."

Some incidental points in this paper are also interesting. To attack the problem of tuning, the authors mention a data-logging and rule-tracing procedure that seems to have worked something like the features found in some expert system shells. In hindsight though, they were wrong about what should be adjusted in the tuning process: They felt that only rules should be changed, not the linguistic variable definitions, whereas the current trend is close to the opposite. The only unfortunate point is that they used what we call the standard interpretation of the fuzzy inference process.

5.8.2. *Cement Kiln*

A few years passed before fuzzy control left the laboratory to be implemented for truly practical applications. One of the earliest of these was the control of a cement kiln in Denmark, as presented by (Holmblad and Østergaard, 1982). This is quite impressive for a number of reasons.

The efficient operation of a continuous feed cement kiln, as the authors describe it, seems to be a very challenging problem even for human operators. Due to the time-varying, nonlinear behavior of the plant and the relatively few measurements available, it was previously impossible to automate the control of anything more than a few secondary variables. In the introduction, the authors state

There is a striking contrariety between the difficulties encountered in establishing adequate mathematical descriptions of the kiln process on the one side, and on the other side the relative ease by which human beings can be trained and in a relatively short time become skilled kiln operators.

In this connection it may be observed that a mathematical model approach builds on descriptions mostly using absolute, numerical quantities, whereas human operators more seem to think and act according to approximate relationships involving vaguely defined, linguistic quantities like "high," "small," "OK," etc.

As remarkably valid evidence of this point, the authors next cite U.S. textbook for cement kiln operators where heuristics are stated in a format that is effectively that of fuzzy IF-THEN rules.

Wet process cement kilns are used to produce cement clinkers by gradually heating (eventually to about 1430° C) limestone, clay, and sand, which were first mixed with water to form a slurry. These clinkers are later ground finely to produce the actual cement, but we are concerned here only with the production of clinkers.

The continuous feed kiln is a very large steel cylinder, about 165 m long and 5 m in diameter in this case, tilted slightly from the horizontal. It is rotated by a motor at a rate from only 1–2 RPM. The slurry is continuously fed in at the higher end of the cylinder, and due to the slow rotation of the cylinder, the material gradually works its way to the lower end and out over a period of about 3–4 hours. As with many such processes, the flow of heat is counter to the flow of material: The hot gasses from a coal burner are fed into the lower end of the cylinder, then drawn through by an exhaust fan at the high end.

At first thought one might guess that several variables in this process can be easily adjusted for overall control, but actually this is not so. Because it is a slow, continuously fed process, at any given moment there are large amounts of material in each of the many stages of processing. Thus it is simply ineffective to attempt to control the process by adjusting the kiln rotation rate or the slurry feed rate. The variables that are usually adjusted are

- Coal feed rate (into the burner) and
- Exhaust gas damper adjustment (near the gas exhaust fan at the higher end of the cylinder).

But even these must be adjusted with great care because they also influence the material in many stages of processing.

It also may seem that many variables of the process can be measured for input to the control device, but again this is not true. For one it is nearly impossible to install sensors inside a very hot, rotating cylinder filled with abrasive material. The following measurements that can be used are of a much less direct nature:

- Percentage of oxygen in the exhaust gases
- Percentage of carbon monoxide in the exhaust gases
- Percentage of nitrogen compounds in the exhaust gases
- Temperature of the exhaust gases
- Torque required to drive the kiln (by the motor)
- Specific gravity of clinkers just completed (kg/l)

There are considerable time delays in the measurement of all of these variables, but the delays are especially large in the case of the clinker specific gravity, which indicates conditions in the kiln 1–2 hours earlier. This is unfortunate because in other respects, this is a very useful measurement because it indicates both the temperature within the actual burning zone and the quality of the clinkers.

All things considered this is an incredibly difficult control problem, as are many industrial processes. Yet even at that early stage in the development of fuzzy control technology, Holmblad and Østergaard were able to control the process with only 27 fuzzy control rules. Actually, they designed the control to operate in two modes, based on whether, by a separate algorithm, the computer detected that the system seemed to be running stably or unstably. They implemented the control rules in

their own version of fuzzy control language (FCL) on a minicomputer connected to the system. Their FCL was curious because propositions are stated by first giving the linguistic value, then the variable name in parenthesis, such as LOW(O₂) to indicate the oxygen level is too low, but it was very practical. They did not require all input variables to be stated as conditions of the rules, but other than that, the general structure of their rules was quite similar to Mamdani and Assilian. The FCL was designed to allow 12 possible linguistic terms, but apparently each variable used only a subset of these. In addition to the usual sounding large_negative, medium_negative, etc., they also allowed high, OK, and low.

Fortunately their interpretation of fuzzy inference was essentially what we call the alternative interpretation. Specifically, they attach a “degree of fulfillment [sic]” to the condition part of each rule.

5.8.3. Sendai Subway Trains

One of the practical applications of fuzzy control that has had the highest visibility is the automatic train operation (ATO) of the Sendai Subway System, developed by Hitachi, Ltd. automatic train operation systems that control acceleration and braking in mass transit trains automatically by computers are not a new idea. Although typically, at least one human operator must still be aboard each train to take over in case of a breakdown or other emergency, ATOs offer the potential for overall higher levels of safety, comfort, efficiency, punctuality, accuracy of stop position, and freedom from stress than might be possible for a human operator alone.

There is some potential for ATO development based on conventional control design, and in that particular sense perhaps a fuzzy ATO is not quite so impressive as a fuzzy cement kiln. But in other ways, this application is very impressive, particularly in that it was the first time that the safety of large numbers of people was entrusted, even in part, to a fuzzy control device. Furthermore it is probably only partially true to say that an effective ATO device can be based on conventional control. It is very difficult to design a conventional control device to deal with more than one or two criteria at a time. For example, a conventional control device could be designed to stop a train in the correct position, but it could probably not be expected to do that and simultaneously maximize comfort, efficiency, and punctuality.

Several papers have appeared on the Sendai subway fuzzy control systems, both in English and Japanese, but a definitive one is still (Yasunobu *et al.*, 1983), written when Hitachi was progressing toward the actual detailed designs. One particularly interesting aspect of this system is that inputs to the fuzzy control rules are various fuzzy performance indices that are calculated indirectly from measured variables. It is an early example of a hierarchical structure in fuzzy control design. That is directly sensed or measured variables, such as

Speed,
Position marker signals, and
Signals from a supervisory program in the host computer,
are used to calculate such fuzzy indices as,
Safety performance,
Comfort performance,
Traceability performance (which essentially means speed error), and
Stop gap performance (how close the train will come to stopping at the correct point in the next station).

These indices (which take as values fuzzy numbers) are then fed to the fuzzy control rules which determine the adjustment to just one output variable, namely

Control notch position.

The discrete control notches control the acceleration of the train, either powering the motors in the case of a positive notch or applying the brakes in the case of a negative one.

The fuzzy control rules are broken up into two parts for two modes of operation. One mode is for accelerating out of the station and running at more or less constant speed between stations. The second mode is entered at a set point when approaching the next station, and it controls the approach in such a way as to stop on target, while still maintaining comfort and safety.

Clearly, this is the type of application for which the controller must respond fairly rapidly. Therefore the discrete cycle time is on intervals of 100 msec.

Yasunobu *et al.* compared performance to both human operators and conventionally designed controllers by means of simulation. They found that if a conventional controller was designed to stop on target, then the stop gap was about the same as for the fuzzy controller, but the conventional controller tended to change the control notch position more frequently, thus causing an uncomfortable ride. Interestingly, the comparison with the human controller was somewhat the reverse: Experienced human operators tended to adjust the control notch less frequently than the fuzzy controller, but the result was still an uncomfortable ride because the human adjustments were too infrequent and drastic when they did occur. By actual experience of riding the Sendai subway, one can certainly notice a feeling that the acceleration and braking are much smoother than on most trains, and certainly much more dependably so. The result is a comfortable ride even when standing.

5.8.4. Consumer Appliances in Japan

As a result of highly visible fuzzy control applications, most notably the Sendai subway fuzzy control system, the interest in fuzzy control in Japan became more widespread during the late 1980s. That is the engineering community at large

became quite familiar with the basic concepts, and the general public began to have some sense of name recognition for the word fuzzy.

This interest, combined with a number of other factors, led to a boom in consumer products advertised as using fuzzy control in 1990 and 1991. The range of “fuzzy products” was very wide, including major appliances (refrigerators, washing machines, air conditioners, etc.), minor appliances (rice steamers, vacuum cleaners, etc.), and home electronics (especially video cameras). In all of these product lines, every major Japanese electrical manufacturer offered products, often several models of each type, that were advertised as “fuzzy.” The result was that especially during 1991 if one walked into the electrical appliance section of any Japanese department store, large or small, it was possible to look around and see stickers attached to nearly every piece of merchandise with the word “fuzzy” written on it, usually nothing more, just the word “fuzzy.” Models without such stickers were usually heavily discounted.

By 1992, products that were simply “fuzzy” were already beginning to seem like old hat. By that time, products with “neuro-fuzzy” or “neuro & fuzzy” stickers were appearing, and it was the “fuzzy”-only products that were being discounted. This trend continued into 1993, and then eventually the “neuro-fuzzy” products started to seem old so that even that label is rarely used as an advertising slogan anymore. Furthermore, during the recent, long-term Japanese recession, some of the research groups formed in better days in cooperation between academia and industry were broken up due to loss of industrial support. While Japanese consumer product manufacturers have not abandoned fuzzy control methods, it is difficult to know how much such methods are still used because they are rarely discussed, and we cannot tell what sort of algorithms drive a microprocessor-based product.

Plainly, there are several lessons one can learn from observing this boom-and-bust phenomenon, some positive and some negative. Clearly, the Japanese manufacturers proved that given low cost microprocessors and low cost sensors, it is very much possible to implement low cost fuzzy control of actual products. Furthermore, they did so mostly without using fuzzy inference chips or other special purpose hardware, which is significant.

Unfortunately it is more difficult to judge how much more useful those products actually were by virtue of having fuzzy control. If over a period of years people continue to buy Product A more frequently than Product B, and if no special advertising gimmicks or fads are involved in this choice, then one can conclude that Product A is the more useful product. However, if people buy a product for the first time, knowing nothing more about it than that it is the new thing that everyone is talking about these days, then this proves a great deal about the power of advertising, but almost nothing at all about the relative utility of the product. We have yet to see any consumer review done on fuzzy products with even a pretense of objectivity and freedom from hype.

Therefore judgments on how useful these products were is subjective and based on consideration of specific products. Our evaluation based on only a few specific products is that these are probably more useful. In particular it seems that more features were more fully automated in each product with fuzzy control than one would have expected of a similar product that was microprocessor-based but without fuzzy control. However Japanese skeptics sometimes pointed out that low-cost sensor technology was progressing at the same time, so it is difficult to know how much the various improvements in automation are due to fuzzy control and how much they are due to more and better types of sensors that had fallen enough in cost to be placed in consumer products for the first time. Nonetheless we still believe that fuzzy control led to actual improvements in product utility.

There is little solid information on how much simpler the control systems of these products were to design using fuzzy control methods. However partly based on how quickly the various models came out, it seems clear that it was not unusually difficult to learn fuzzy control design.

Much speculation was made, particularly in 1991 and 1992, about why fuzzy control products developed to such a degree in Japan, but hardly at all in other countries. Now that the boom is over, it is perhaps possible to reflect on this more rationally. Many looked for very direct reasons of a deeply cultural nature; "Western minds inherently reject fuzziness, Japanese minds inherently embrace it," or whatever. At one time we were somewhat sympathetic of that sort of view, but we have since seen it so much overstated that it seems now very objectionable. For one thing it could be interpreted to support the favorite self-stereotype of the Japanese, that "We Japanese are not logical, not even rational." The real reasons are probably much more complex, and some of them may relate to culture but in a more subtle way than this one simple idea.

Japanese industry has been praised for its engineering standards in the area of consumer products over the past few decades, but it probably deserves even more praise than it receives. The United States for example is familiar with Japanese automobiles (since 1980, mostly with the higher cost models) and consumer electronics and cameras from Japan, but we are not so familiar with some other types of products such as home appliances. Yet, it is in the area of home appliances, especially minor appliances, that the Japanese industrial attitude is most remarkable; the attitude that no product is so small or so mundane that it does not deserve to be designed with the best engineering and the best new ideas possible.

As a specific example of this, when microprocessors first became available during the 1970s, there was a significant difference in how they were viewed by U.S. industry and Japanese industry. Although the United States was first in using microprocessors in the original personal computer systems, in other areas Americans seem to have still been thinking too big about where to apply microprocessor technology. Most of the talk was about high cost engineering devices, such as surveying equipment, and for automobiles once noise-shielding technology devel-

oped enough to allow that. By contrast at that early stage, Japan was seriously talking about putting microprocessors in washing machines, rice steamers, and many other minor products. Some microprocessor-based products of this nature did appear about 1980, and by the late 1980s a fairly high percentage of Japanese appliances for the domestic market had microprocessors inside, even very small and mundane sorts of appliances.

This is not only an example of how easily Japanese industry thinks of using new technology in home appliances and other products, but it also provided a practical base to work from in applying fuzzy control. If an appliance was designed to contain a microprocessor and various sensors anyway, then in most respects it is only a minor design change to implement fuzzy control as well by using this hardware.

Of course, none of this would have been possible if the market did not cooperate, and in this respect Japanese are quite cooperative consumers. The simple word *shinseihin* (new product) can be used to sell readily almost anything in Japan. Some of the stereotypes are partially valid, and some are complete nonsense, but in general it is true that Japanese are quite conformist, quite status-conscious, and quite interested in new fads for almost everything. They are also less likely than Westerners to have strong anti-technology, anti-big business, or nostalgic sentiments. Japan is not only highly urbanized, but the majority of people are crowded into just the top few metropolitan areas, all of which are remarkably similar in atmosphere. The point is that there are very many reasons why a new idea related to consumer products might succeed first in Japan, without the need to talk about notions of fuzzy-brained vs. crisp-brained cultures or anything equally bizarre.

As relates to the English-speaking world, there is one other point that deserves mentioning. In the earliest publications in the Japanese language about fuzzy set theory and applications, the borrowed word “fuzzy” was not used. Rather it was called *aimai*, the Japanese word for vague. More specifically, *Sanseido’s Daily Concise Japanese-English Dictionary* gives the English definition of *aimai* as, “vague; ambiguous; evasive; unreliable; uncertain.” Thus the Japanese wrote “vague sets,” “vague engineering,” etc. in their own language. But none of the ideas were catching on in Japan at that time outside of a very small group of researchers. Soon these researchers made the decision to stop using the word *aimai*, and to replace it with the borrowed word “fuzzy”, a word that few Japanese understood.

It is only speculation, but one wonders how important this change in terminology was. Would the Japanese have been so willing to buy *aimai* washing machines, microwaves, and vacuum cleaners, as they were to buy “fuzzy” ones? To Japanese consumers in 1991 “fuzzy” was still a delightfully exotic word which they only partially understood, and in Japan, exotic sounding, partially understood words borrowed from European languages tend to be very chic. To an English-speaking person, on the other hand, it is a provocative use of language to call a field of knowledge or a type of electrical appliance “fuzzy.” Zadeh certainly must have had his reasons for choosing the term, and it was certainly his prerogative to choose.

But for anyone to criticize the English-speaking countries for a supposed relative lack of interest without at least considering terminology as a possible factor is being critical in a very unreasonable way.

This is a particularly significant point when considering taking fuzzy control out of the laboratory. One must remember that the industrial environment is quite different from academia. In many cases, an engineer may be required to obtain approval from management, probably from multiple levels of management, before proceeding very far with a new approach. And management often makes decisions more on the basis of how good something sounds than on its actual technical merits. Naturally, this is all the more true when it applies to consumer products, and one intends to advertise or even just to publicize the nature of the product.

Finally, although it is true that engineering applications of fuzzy set theory have experienced considerable popularity in Japan, it is erroneous to assume that there is anything close to consensus on this point. Among Japanese scholars in such fields as electrical engineering and systems science, there are many who are so skeptical as to be openly antagonistic toward fuzzy set theory. As for the broader public, even in 1991 few Japanese understood what fuzzy control was except as a trendy new, high-tech buzzword; and since that time it has slipped from public consciousness even in that sense. After calmer reflection than prevailed a few years ago, we believe that one will find fewer and fewer reasons to believe that cultures vary significantly in their basic perceptions of vagueness.

5.8.5. Model Helicopter Flight Control

In the 1990s it has become more difficult to develop a revolutionary or even dramatic application of fuzzy control. So much has been done already. There is to some degree an attitude now in the Japanese research community that knowledge about the fuzzy control design process is complete enough so that true researchers hardly need bother with it at all anymore. However, there is still the need for doing work in how fuzzy control methods can be modified by combinations with other new ideas, and this work is continuing. In addition to that, a few researchers simply wish to continue to push the limits of what can be done with fuzzy control by finding the most difficult applications imaginable.

It is in that spirit that Sugeno and others at Tokyo Institute of Technology in a long-term project have been applying fuzzy control methods for use in flight control of unmanned helicopters. It is generally agreed that flight control of a helicopter is a much more difficult problem than flight control of a fixed wing aircraft, even for human pilots. It has been generally viewed until recently as nearly impossible to develop a truly useful automated helicopter flight control system for all modes of flight. This is precisely the reason that the Tokyo Institute of Technology group is so interested in attacking the problem.

A paper by Saitoh, Hirano, and Sugeno (Saitoh *et al.*, 1993) is a concise report of their results as of that time. It briefly describes the modular structure of their system with an upper module for determining medium level goals such as flight direction, speed, etc., and a lower module for directly controlling the flight surfaces of the helicopter. Thus, it is hierarchical in a manner quite similar to the Sendai subway control system. They report some rather satisfactory results, both based on simulation and on actual operation of a small helicopter by remote control.

5.9. A Further Comparison of Fuzzy and Conventional Control

It is not a primary goal of this book to compare fuzzy control design methods with the conventional methods, but we offer a few general comments. As discussed in Chapter 2, conventional control design methods are very highly developed, and they work remarkably well for relatively simple systems where one only needs to control one variable and where it is possible to develop an accurate, linear model of the plant. However the assumption of a relatively simple system is a critical matter, and even in this case the level of expertise needed to apply conventional control theory effectively is quite high. However, as we also mentioned for some applications, it is often possible in practice to buy an off-the-shelf conventionally designed PID controller, install it, and tune it based on experience and trained intuition. Nonetheless there is an overall sense of limitation to the types of control problems that can effectively be attacked by someone with only a moderate degree of knowledge in conventional control theory.

By contrast in fuzzy control there is much less of this sense of limitation even with only a moderate degree of knowledge. Very few assumptions need to be made about the nature of the system. It is possible to control several variables at once and even to control on the basis of intangible criteria, as was done with performance indices in the Sendai subway system. None of this requires one to reach the level of a true expert in the field.

Fuzzy control is so readily applied to multivariate control specifications that it is easily possible to incorporate, as feedback control, many aspects of a system that might otherwise have been controlled by sequential control only. It is also possible to view input variables in ways that make the distinction between control device and expert system seem insignificant.

All of this is very favorable to fuzzy control, but it is important to remember that there are a few problems with it as well. We can think of about four such problems.

First the matter of fuzzy inference for approximate reasoning is far from trivial. Its mastery requires some serious study even if one limits oneself to the classical operators, as we did here (although what we call the alternative interpretation is

much easier to study than the standard one). Furthermore, it is still debated which operators are best to use.

Second, typically a fuzzy control design involves some amount of knowledge-engineering work in order to determine the control rules and the linguistic variable definitions. As in most types of AI applications, it is a difficult problem to express knowledge that is wholly or partly intuitive. In some sense the structure the rule base should take is more obvious for a fuzzy control design than for an expert system design, but in other ways the process is still quite difficult. As mentioned there is often more difficulty still in communication in the case of industrial process control. In some cases this can be overcome partially by simply observing the actions of experienced human operators rather than asking them, but it may still be difficult to obtain complete information.

Third unlike conventional control, fuzzy control incorporates no obvious way of predicting how well a given control-device design will perform until it is actually built. One can simulate, but that requires some sort of accurate model, which we may have been assuming we did not need because after all, it was supposed to be one of the advantages of fuzzy control design that precise models are unnecessary. However, the model need not be linear, and overall there are some ways of overcoming this difficulty, as we discuss in later chapters. Furthermore there is at least a little comfort in the idea that the literature demonstrates the natural robustness of fuzzy control designs, so the results are likely to be good.

Fourth also unlike conventional control, fuzzy control theory in its classical form does not provide a rigorous approach to tuning a controller if it is found to be below standards in some aspect of its performance. However much of the research on improving classical method has been directed at this particular problem, and there are now several approaches for dealing with it, some of which will be related in Chapter 6. Furthermore we can again argue that there is a high likelihood that the original design will be quite good and most designs are not so difficult to diagnose and correct in practice.

Despite these several challenges and potential drawbacks, there is now ample evidence that fuzzy control design works well in practice. The challenge is not so much in how to make it work, as in how to make it work as well as possible. This is a good reason to investigate for example the question of which forms of dynamic compensation work well for a typical fuzzy control device, as we do in Part II of this book.

CHAPTER 6

Some Common Variations in Fuzzy Control Design Methods

When we refer to the classical approach to fuzzy control design, we mean any design that more or less follows the methods first developed by Mamdani and his colleagues. More specifically, this approach is characterized by:

- A fuzzy inference engine based on the min-max operators and other standard assumptions of approximate reasoning.
- A knowledge base (both control rules and linguistic variable definitions) constructed by knowledge engineering work rather than any form of machine learning.
- Linguistic variables that are of the normal sort for both inputs and outputs; that is, each term relates to a fuzzy number in the base variable space.
- Control rules based on concepts analogous to either P or PD control in the conventional theory. No other forms of dynamic compensation are considered.

Chapter 5 discussed the classical approach in great detail, and it included our own views on many related matters. It is the purpose of Chapter 6 to briefly discuss some of the most significant variations on the classical approach that have developed recently. The specific topics covered are not comprehensive and the treatment of each will be less than complete, but the purpose is to give a basic idea of what has developed outside of the classical approach. No discussion of fuzzy control theory would be complete without at least this much discussion of variations in methods.

Looking ahead the specific designs discussed as examples in Part II related to this chapter have the following characteristics:

- They use various forms of dynamic compensation, which is indeed the point of the comparisons.
- They are based on a neural plant model.
- They are tuned by genetic algorithms.
- On all other points they follow the classical approach to fuzzy control.

Thus, Part II will effectively offer detailed examples for some of the same topics presented in this chapter.

6.1. Fuzzy Singleton Method

Over the past few years, a concept known as the fuzzy singleton method has become quite popular for new applications, at least in Japan. The meaning is quite simple: The linguistic variable definitions associated with the input variables are the same as in the classical approach, but the linguistic variables for each output variable take a new, simplified form. That is each term in the term set of an output variable is related not to a fuzzy number, but rather to a crisp value in the base variable space.

The rationale for this is also quite easy to understand. The first point is that, as we discussed as early as Chapter 4, there is something of an *ad hoc* feeling about the whole defuzzification process, at least as it developed historically. As related in that chapter, relatively recent work by Filev and Yager has added a degree of rationality to the defuzzification concepts, but this still has had only minor influence on the way defuzzification is viewed in most practical applications to date. On the one hand, the mean-of-maxima and the center-of-gravity methods intuitively seem appropriate; but on the other hand, most designers of fuzzy control devices seem to have little concern for rigorous arguments as to why this may be so. The second point is that it is very difficult to explain precisely what we hoped to gain by associating output variable terms with fuzzy numbers in the first place.

Our own perspective has been to think of the ordinary type of linguistic variable definitions as being based on a fuzzy relation between the term set and the base variable space. For the simplified sort of linguistic variable, the fuzzy relation can be replaced by a crisp function, presumably a one-to-one function, from the term set to the base variable space. That is where n is the number of outputs, for all $j \in \{1, 2, \dots, n\}$, there exist functions such that:

$$g_j : T_{yj} \rightarrow Y_j$$

where T_{yj} is the term set for the j th output variable and Y_j is the base variable space for the same variable.

However we recall that the result of applying the fuzzy control rules is a fuzzy subset of T_{yj} , something we referred to as the y'_j value. Therefore we must again think in terms of the extension principle to apply the function in the following way:

$$g_j : \mathcal{F}(T_{yj}) \rightarrow \mathcal{F}(Y_j)$$

The result is a fuzzy subset of Y_j . Let us call this fuzzy subset y''_j because it differs only slightly from the y'_j values illustrated in Fig. 5.1 and discussed in Chapter 5.

Therefore, we must defuzzify in a certain sense. Specifically we select the value $y_b \in Y_j$ whose membership grade $y''_j(y_b)$ is greatest. If there is no unique maximum, then we must have some method for choosing, such as taking the mean, then rounding to the nearest base value. The final crisp output y_j can be thought of as:

$$y_j = \text{mean of } \{y_b \in Y_j \mid y_b = \underset{y_s \in Y_j}{\text{Max}} y''_j(y_s)\}$$

rounded to the nearest value in Y_j . This too may seem slightly *ad hoc*, but it is so simple that it is difficult to imagine doing things in any other way.

The overall result is that Steps 3 and 4 of the fuzzy inference algorithm (see Chapter 5) are considerably simplified and somewhat rationalized. (Steps 1 and 2 remain the same.)

An informal survey conducted by looking through issues of *Journal of Japan Society for Fuzzy Theory and Systems* leads us to conclude that perhaps half or more of the new fuzzy control applications discussed in recent years in Japan use the fuzzy singleton approach. A specific example is (Geng and Muta, 1993), which describes a control system design for trains on a certain railway line in China. Their design used four input variables,

Train speed error,
Train time error,
Wind, and
Slope,

to determine one output variable,

Control notch position.

A linguistic variable of the usual type was associated with each of the four input variables, each with term sets of five to seven terms. Associated with the control notch position was a simplified type of linguistic variable with seven terms, each corresponding to a specific control notch. An incomplete set of 49 control rules was used.

Other fairly recent papers, such as (Mizumoto, 1993a), demonstrate that under reasonable assumptions, the fuzzy singleton approach and the classical approach with COG defuzzification are essentially equivalent. This should lead to greater use of the fuzzy singleton method in the future.

6.2. Variable Reduction

Having just considered a fairly straightforward approach for simplifying the output variable space of the approximate reasoning process, we now discuss an approach for simplifying the input variable space in an even more straightforward way. This approach simply applies the general concept of variable reduction to the fuzzy inference procedure. In this section we present the basic concept of the approach, then consider its significance and its advantages and disadvantages.

Perhaps the best way of motivating and explaining the approach is to discuss a simple example. Let us consider an application very similar to the pioneering steam-engine-control experiments at Queen Mary College which uses four input variables,

PE (pressure error),
 PC (change in pressure error),
 SE (speed error), and
 SC (change in speed error),

are used to determine two output variables,

HC (change to boiler heater setting) and
 TC (change to throttle setting).

We have seen that Mamdani *et al.* dealt with this situation by using only 24 rules, but two qualifications must be considered for how this small number was attained. First by using disjunction, logical negation, and the special term ANY in the antecedents, most rules treat multiple combinations of the inputs. Second there is no guarantee made that the rules considered all combinations.

However if we consider instead a tabular representation of the rule base as discussed in Section 5.2, it would have a more imposing appearance of two arrays each of four dimensions. If we assume seven-term term sets for the linguistic variables of each input, there are a total of 4802 locations in the tables, each of which must either be filled with a linguistic term related to an output variable or left undefined.

Alternatively, we consider the following modification. Crisp inputs (before fuzzification or any other part of the fuzzy inference process is performed) are combined as follows:

$$\begin{aligned} PI &= PE + a(PC) \\ SI &= SE + b(SC) \end{aligned}$$

where PI, PE, PC, SI, SE, and SC are each considered as crisp (ordinary numeric) values and a and b are weights determined by heuristic knowledge about the system. Let PI stand for pressure error index and SI for speed error index. These PI and SI values are then fuzzified on the basis of appropriately defined fuzzy relations to their corresponding linguistic variable forms. The rule base still consists of two tables, but in this case each table has only two dimensions. If seven-term term sets are again assumed, this results in only 98 locations to be filled in the tables, and the entire rule base can very easily fit on one sheet of paper.

Some of the results from this approach are quite obvious. By taking linear combinations of the input variables in their crisp forms, the rule-base-specification phase of the design process becomes a problem of greatly reduced size. Furthermore the design becomes simpler to implement, and each cycle can be calculated more

quickly. These last points are particularly important if there is an incentive to use inexpensive electronic hardware. Clearly, these are significant advantages of variable reduction. Also, the concept is easy to understand, and has a certain intuitive appeal.

However if we consider carefully the nature of variable reduction as it applies to fuzzy control design, some potential disadvantages are apparent. As with many of the other topics presented in this chapter, the use of variable reduction in this way is a compromise: It moves one significant step away from specifying system design by the fuzzy logic-based approximate reasoning model, and one step closer to specifying system design as a conventional, or crisp, mathematical structure. It is still possible to assume that this system design is based on heuristic knowledge, but if that is the case, then part of that heuristic knowledge must be expressed as a crisp structure—not just in selecting the parameters but in deciding which inputs should be combined in which ways in the first place. Pursued to its extreme this approach presumably results in the expression of complete mathematical models of the control device, based on heuristic knowledge, but makes no use of fuzzy set concepts.

There is nothing necessarily wrong with this idea, at least not in its compromise form, but it is clear that there are potential limitations. First variable reduction sacrifices some degree of flexibility for the sake of simplicity in specifying the rule base. The relative influence of two combined inputs is fixed by a single parameter, and it is no longer possible to vary the relative strength of influence for specific conditions.

The second potential problem is that variable reduction raises the level of abstraction in design specification, which results in an approach that is at least slightly less compatible with design based on heuristic knowledge. The concept of a pressure error index for example is more abstract than either of its component terms. It is not clear that we can specify either a fuzzy relation for its linguistic variable or rules based on it as a natural expression of heuristic knowledge.

In this sense using of variable reduction in a fuzzy control design seems more natural when combined with an approach based on plant models and auto-tuning, discussed later in this chapter. Specifically, the parameters could be adjusted as well as the linguistic variable specifications and perhaps even the rules. This approach would be conceptually close to the midpoint between conventional control and classical fuzzy control.

6.3. Variations in Inference Algorithms

The question of which operators to use for implication, conjunction, disjunction, and logical negation can be debated for a decade or more with seemingly little progress toward a resolution. Early papers, such as (Baldwin and Guild, 1980),

seem to be no different in spirit from more recent papers, such as (Mizumoto, 1993b). As (Nguyen, 1993) points out, there are various reasons for suspecting that operators used in the classical approach are not the best, but there seems to be no definitive way of demonstrating this point. The question is which way of approximating the vagueness in human reasoning is most effective in capturing human intuitive thinking on how to design a control device. On the one hand, experimentation seems to be the only way to approach that question, but on the other hand experiments do not constitute a proof.

We do not go into much detail on this issue for two reasons. First as much as the issue is debated and as often as researchers raise questions about the classical operators, there still seem to be few actual applications where the choice of operators is considered a critical point. Second the context of the issue is actually approximate reasoning, a more general topic than fuzzy control. It is just that fuzzy control, as the most established area of application for fuzzy set theory, becomes a natural forum for debates of this type.

Those interested in the issue can refer to the literature. In addition to the references just listed (Nakanishi *et al.*, 1993), (Baldwin, 1981), (Baldwin and Pilsworth, 1980), (Mizumoto and Zimmermann, 1982), (Tsukamoto, 1979), (Cao *et al.*, 1990), (Yu *et al.*, 1990), (Nafarieh and Keller, 1991), (Piskunov, 1992), and the entire volume edited by Mamdani and Gaines (1981) are all quite informative.

6.4. Plant Models for Fuzzy Control Design

At first thought, it may seem that there is a conflict between the rationale for fuzzy control design and the idea of modeling the plant to be controlled. After all, did we not claim that the primary advantage that fuzzy control had over conventional methods was that fuzzy control made it unnecessary to develop a precise mathematical model for the plant? It perhaps seems almost underhanded to come back now and begin talking about why it is often beneficial to model the plant as part of the fuzzy control design process.

There are three points on which one can respond to this type of contention. First a plant model is still not a necessity or a central focus in a fuzzy control design, only a convenience that makes the testing and tuning steps more economical and that allows for the use of certain advanced design techniques. Second the modeling process is much less constrained with fuzzy control because there is no strong incentive to make the model linear and otherwise highly simplified as there is in conventional control theory. Third the modeling process is less constrained still because the model need not even be of a precise type. That is the model need not take the form of differential equations. It can be a fuzzy model, a model based on neural nets, or it can assume other reasonable form, precise or imprecise, for

simulating a system. The model can be based on heuristics, and these heuristics can be based on human intuition or learned by the system itself.

There are a number of reasons why a plant model is often beneficial as a part of fuzzy control design.

Because predicting the performance of a fuzzy control device before actually building it and tuning it once built are the weakest points of classical fuzzy control design, a reasonably accurate model, available for simulation on a computer, essentially eliminates the first problem and makes the second one at least more economical to address. A plant model also makes it possible to use various machine-learning or otherwise adaptive methods as part of the design process. When such an approach works well it can greatly reduce the burden of knowledge-engineering work in the design and eliminate the tuning problem entirely. In addition a plant model is an economical way of examining the feasibility of new variations in design approach and for comparing approaches, as demonstrated in the examples in Part II.

As early as the mid-1980s Sugeno considered fuzzy models of the plant to be an important direction for the further development of fuzzy control, as seen in (Takagi and Sugeno, 1985), (Sugeno, 1985, 1987); however (Procyk and Mamdani, 1979) used plant models in fuzzy control even earlier. Even after studying various examples in the literature, it is difficult to define fuzzy modeling precisely; it is even difficult to state what fuzziness is supposed to do for modeling.

On the one hand, if one carefully reads Takagi and Sugeno (1985), it becomes clear that their primary goal is to establish a new type of regression model, one that can be described as something close to piecewise linear, but in a more rounded off form, where especially the divisions (nondifferentiable points) are very much smoothed or rounded. Takagi and Sugeno use fuzziness (without explaining the rationale for this very carefully) to smooth out piecewise linear regression models. Other researchers extended these ideas; directions for example (Filev, 1991) applies the ideas to more general types of piecewise functions with similarly rounded off corners. Some aspects of Kosko's approach [see (1992)] for combining neural and fuzzy approaches are also related to this view.

These approaches presented by Takagi and Sugeno and others are sometimes considered formal approaches to fuzzy modeling. Unfortunately at least from their description in key papers presenting them, there are some difficulties. First the rationale for formal models is not obvious, and most papers make little effort to enlighten us on this point. Second if directly analytical methods are used to solve fuzzy regression problems, as many of the papers imply, some messy applied mathematics beyond the comprehension of many potential users is involved. Third these formal approaches as first described do not mention specifically how whatever heuristic knowledge may be possessed about the system could be used at all. The papers seem to imply models derived solely from data.

However another general approach to fuzzy modeling is more intuitive; it is sometimes referred to as the informal approach, and it is still viewed in the literature as viable in practical problem solving [see for example (Deboeck, 1994)].

Let us concentrate on this more basic concept of fuzzy models and particularly on its rationale. Recalling Fig. 4.5 there are several possible ways (only a few are represented Fig. 4.5) of expressing a system behavior. These include not only the artificial systems that we may wish to design but also the natural or otherwise preexisting systems that we may wish to analyze. We are more accustomed to thinking of using heuristics (in the form of rules, frames, fuzzy set definitions, etc.) for expressing an artificial system that we wish to design, but there is nothing that completely rules out the use of heuristics as a medium for expressing the behaviors of natural systems as well. For example in designing a system, we interpret a rule of the familiar form:

IF conditions THEN action

to mean under these conditions this is what should happen (or what I want to happen). But there is no reason that it could not just as easily be interpreted to mean under these conditions this is what does happen (according to my observations, experience, intuition, etc.).

In Chapter 5 the term naive physics was briefly mentioned. We note that the informal approach, which can also be called a naive approach to fuzzy modeling is a more direct application of naive physics than is the design of fuzzy control rules. The process of building such a model is very similar to the classical fuzzy control design process as presented in Chapter 5. That is we consider the following:

- The system inputs and outputs
- Rules to describe one's knowledge of what the outputs can be expected to be for various combinations of the inputs where both inputs and outputs are described linguistically
- Fuzzy sets (actually fuzzy relations) for relating the various terms for each variable to quantities
- A fuzzy inference engine or algorithm

This informal approach, provides a good rationale for fuzzy models. These models are based on human-learned heuristic knowledge of the system; fuzzy aspects of such models are there to provide the best way of capturing the vague aspects of that heuristic knowledge. The methods for building such models are also easily understood by anyone able to follow the basic points in Chapters 3–5 of this book for example, and these methods do not require difficult mathematics. They also allow us to use our heuristic knowledge, and in fact they are based on it.

As clear as the rationale and methods may be, there are some obvious limitations to building fuzzy models based solely of human-derived heuristics. In describing the behavior we wish to see in a system we are designing (such as the fuzzy

control device itself), it seems appropriate to base the design on our heuristic notions of how we want it to behave. But when trying to describe the behavior of a pre-existing system (such as the plant we wish to model), we want to incorporate precise empirical data about the system.

What can we say about fuzzy modeling? On the one hand most formal approaches, at least as originally presented, seem to have little or no clear rationale; these are difficult to solve and seem to allow no intuitive human input at all. On the other hand completely informal models, though practical in some circumstances, have so little rigor as to be based on supposition.

Perhaps the best solution is a sort of compromise, a semiformal approach. Models of this type would be clearly based partly on human intuition about the system and partly on empirical data. For example the human-learned heuristics can provide the overall structure of the model at a fairly detailed level, but the parameters of the model would be adjusted on the basis of the data. The method of calculation to be used might be based more on numerical and iterative search methods than strictly analytical methods, in that way substituting inexpensive computational power for high-level mathematical sophistication. The rationale for such an approach is also clean: A model based partly on human intuition and partly on numerical data may replicate humanlike thinking. After all, one obviously desirable attribute of any modeling technique is that it should easily allow humans to reach further insights based upon it.

Regardless of which approach is most rational, fuzzy modeling has been used in the fuzzy control design process. One of the earliest examples of basing the complete design of an actual application on the fuzzy model method is (Sugeno and Kang, 1986); the design relied specifically on detailed techniques developed by (Takagi and Sugeno, 1984), who also presented detailed examples related to control. We consider fuzzy models again in Chapter 8.

More recently the use of neural nets for modeling the plant has become a particularly common approach. Provided that one understands the nature of typical applications of multilayered neural nets, it is very easy to understand the basic concept involved in using this approach for capturing the system behavior of the plant, and thereby building a sort of plant model. As discussed in Chapter 3, one way of applying neural net methods is for building an artificial system that gradually learns by example what outputs it should produce in response to various inputs. It is then obvious that one way to build an artificial system that emulates a pre-existing phenomenon is to collect many examples of typical behavior from that phenomenon, and then to train a neural net based on those examples.

As is often the case with machine learning, the practice is not quite so simple as the theory makes it sound. Some care must be taken in setting up the network, especially in determining what the network inputs should be in the case that a dynamical system is to be represented. While it is true that neural nets have the natural ability of learning to ignore insignificant inputs, it is still important to deal

carefully with the overall network structure design because in the end we typically desire a network that can learn in a reasonable amount of time when implemented as software on general purpose hardware. This also implies that great care must be taken using hidden layers. (Chapter 8 discusses a plant model constructed by neural methods in detail.)

Fuzzy models and models based on neural nets are only the two most common approaches to plant models for use in fuzzy control design. There is virtually no restriction on the model format if it lends itself to fairly simple simulation on a computer. More orthodox mathematical formats (differential equations, etc.) can also be used if it is easy to develop the model in that form. In any case a plant model that can be simulated on a computer permits us to test control device designs before building them and to tune designs easily.

6.5. "Neuro & Fuzzy"

During the early 1990s Japanese researchers and Japanese industry became particularly interested in two major approaches for combining machine learning by neural nets with fuzzy control. They called these *neuro & fuzzy* and *neuro-fuzzy*, respectively. A large number of papers have appeared in Japan on these two approaches, but perhaps the best survey of the overall Japanese view on these approaches is (Hayashi, 1993). Both this section and the following one owe a considerable debt to that survey for details and examples. Before presenting those details, we note that these two approaches bear some similarity to Kosko's recent work in combining neural and fuzzy methods [see (Kosko, 1992)], but although Japanese papers frequently reference Kosko, there are differences in emphasis. Also machine learning as applied to the design of a device that interacts with a dynamical system seems to have major difficulty evaluating performance or determining error values. Berenji for example in (1992b), attempted to attack this problem quite directly using adaptive evaluation network (AEN), that is a separate neural net that learns to serve as a better and better evaluation function. By comparison the neuro & fuzzy and the neuro-fuzzy approaches attack this problem less directly; for example the neuro & fuzzy approach typically applies neural methods to only those aspects of a control problem for which a good static evaluation function is readily available, then uses classical fuzzy control methods alone on other aspects. The neuro-fuzzy approach is typically concerned more with reexpressing heuristic knowledge in a fuzzy set format than in directly deriving those heuristics independent of human knowledge. Specifically neuro-fuzzy methods are concerned with producing fuzzy set interpretations of knowledge or information about the inputs and outputs.

We must be careful to avoid misinterpreting the intention of machine learning. Conceivably neural nets can be used to make the control adaptive in an ongoing

sense: We can build a control device that continues learning from experience while being used. This would be very useful, especially if it adapted over time to the conditions and requirements of individual users. Unfortunately this is not what neuro & fuzzy or neurofuzzy usually means. These two approaches reached complete development when Japanese industry was most interested in fuzzy control for consumer products and making products with adaptive control in the true sense was too expensive, so machine learning is typically part of the design process only.

The rationale for these approaches is somewhat sketchy. Japanese technical literature in general tends to emphasize brevity, which is sometimes good but sometimes results in failure to elaborate on the rationale behind a method or to explain the problems it addresses. In fact many Japanese papers skate over the question of rationale simply by saying that these approaches address the problem of tuning, but this explanation is at best incomplete. Neuro & fuzzy and especially neuro-fuzzy do not directly tune fuzzy controllers in the usual sense of diagnosing and adjusting for inadequacies in performance.

Where neuro & fuzzy and neuro-fuzzy require information on plant responses (and surprisingly it is not always required) this information is typically supplied by a plant model, as discussed in the previous section, and not by the plant itself. This is of course true in most cases where machine learning is applied to fuzzy control. To implement machine learning directly on the responses of the actual physical plant could work in theory, but typically it would be uneconomical or otherwise impractical.

The neuro & fuzzy approach can most easily be described as any method whereby a control problem is attacked by using a neural net in parallel with a fuzzy control design of the ordinary sort. Hayashi (1993) further breaks down neuro & fuzzy approaches into two major categories, as illustrated in Fig. 6.1. In (a) the outputs of the neural net are distinct from those of the fuzzy control design; thus the two parts control two separable aspects of the plant, either different modes of operation or completely distinct outputs. In (b) the neural net outputs are a subset of the fuzzy control design outputs, and these are combined in a supplemental manner based on some appropriate weighting scheme. The designer may be confident enough to believe that the fuzzy control design is basically correct just on the basis of the KE (knowledge engineering)-based design process alone but at the same time may not be confident enough to believe it is a perfect design either. In this case some supplemental guidance based on simulated or actual results may improve the overall control.

In either case the neural net is typically intended to attack some aspect of the control problem for which a static evaluation method is relatively easy to develop. If the other aspects of the problem are easily handled by the heuristics directly from human knowledge in the normal KE-based method of classical fuzzy control design, then neuro & fuzzy can effectively combine the strengths of the two methods.

Hayashi, who incidentally is employed in a research laboratory at Matsushita Electric Industrial Co. Ltd., as an example first considers the control of the type of reversible heat pump now popular in Japan that can be used both for cooling (air-conditioning) a room in summer and for heating it in the winter. This he classifies in the category of neuro & fuzzy (see Fig. 6.1a). In other words the neural net and the classical fuzzy control design attack different aspects of the control problem, which are quite separable because they constitute different modes of operation.

Specifically, the neural net attacks the aspects relating to optimizing the comfort levels of the room occupants, and the fuzzy control design attacks all of the various other aspects of operation such as the need to de-ice the outdoor heat exchanger during winter operation, etc. The neural net is trained in the following way: The operation of the heat pump is simulated under the assumption of a particular setting and certain ambient conditions to determine various factors

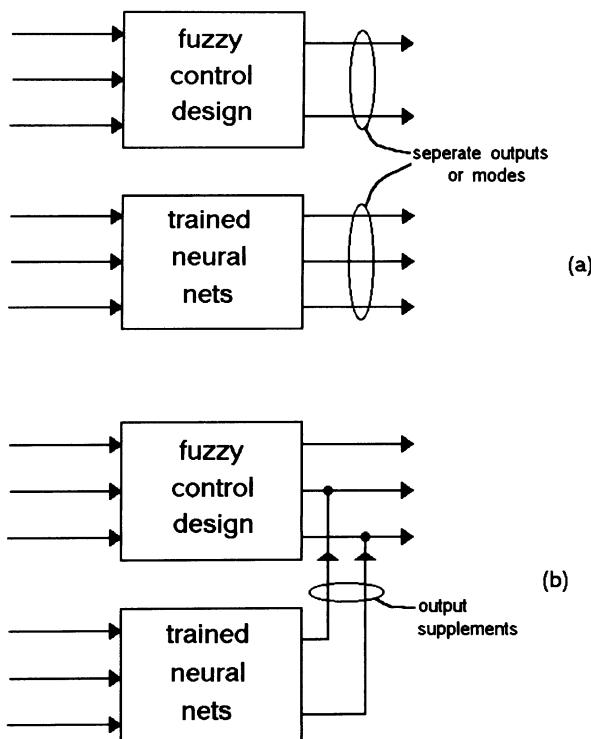


Figure 6.1. (a) First type of neuro & fuzzy control structure, (b) second type of neuro & fuzzy control structure.

relating to comfort, including room air temperature, humidity, velocity, exchange rate with fresh outside air, etc. These factors are fed into an evaluation function that is based on a textbook on comfort analysis for environmental engineering; this evaluation function result yields a comfort index. This index then becomes a basis for training the neural net in combination with the assumed ambient conditions for this case.

As an example of the second type of neuro & fuzzy control, in Fig. 6.1b, Hayashi considers the design of an automatic washing machine. To achieve a more advanced form of automatic control than previously existed in washing machines, we may wish to design a machine that automatically sets

- Water levels,
- Water flow rates,
- Length of wash cycle,
- Length of rinse cycles, and
- Length of spin cycles

all based on the following inputs:

- Size of load,
- Quality of load (how dirty),
- Water temperature, and
- Air temperature.

The fuzzy control designer be quite confident in designing this type of control by classical fuzzy control methods but may be slightly unsure of exactly what the rules should be relating to the lengths of the wash and spin cycles for various conditions.

Therefore the design was completed in the following way. A control device was designed by the totally KE-based methods of classical fuzzy control design using the designer's best knowledge for controlling all five outputs listed above on the basis of the inputs listed. That design was then set aside, and actual data were collected for how long of a wash cycle and how long of a spin cycle were necessary for various conditions. Specifically, 192 examples were considered for the length of wash cycle needed, and 120 examples for the length of spin cycle needed. Two neural nets were then designed, one for the wash cycle and one for the spin cycle. The structure of each of these two was quite simple as neural net structures go, with apparently four (the paper is a bit confusing on this point) input nodes, seven nodes in one hidden layer, one output node in each, and based on a typical back-propagation network (BPN) format. These networks were then trained by the usual BPN-learning algorithm on the basis of the 192 or 120 examples. In the final design the fuzzy control device was used for all outputs, but the outputs for wash cycle and spin cycle lengths were supplemented by the outputs of the trained neural nets. The nature of the weighting scheme used for this supplementation is not explained,

but it obviously must have reflected a subjective evaluation of relative confidence in the heuristic rules and the examples.

One point relating to the application examples given here and elsewhere for neuro & fuzzy control is that it would be difficult to describe them as true feedback control. In the heat pump application, the system is concerned with setting a setpoint, not in maintaining the system at that setpoint. The washing machine has even less to do with feedback control in the ordinary sense. Thus we understand why static evaluation functions can easily be stated for such examples.

If the system designer is familiar with both the classical approach to fuzzy control design and the basic methods of machine learning by neural nets, neuro & fuzzy approaches are easy to apply, particularly for those applications where some aspects of the control problem can be evaluated in statically.

6.6. "Neuro-Fuzzy"

These methods are more difficult to master than the neuro & fuzzy. A number of specific methods were developed in Japan for neuro-fuzzy control, but all are concerned with specifically identifying the fuzzy numbers (in our thinking, the fuzzy relations) relating to the linguistic variables for the various inputs and outputs. Although this is sometimes represented in the literature as tuning, it is actually more of process of identifying from other sources of knowledge how to express the linguistic variables as fuzzy sets. In fact, more specifically, it is primarily concerned with the process of deriving fuzzy sets from crisp data, and in this way it is closely related to the more formal approaches to fuzzy modeling.

It is important to note that neuro-fuzzy methods as presented by Hayashi are for identifying the linguistic variable definitions only. They do not identify the fuzzy control rules. In fact they assume that the collection of control rules (on linguistic terms) is already defined to some degree.

Hayashi discusses two specific, detailed methods for neuro-fuzzy control, both of which he was involved in developing at the Matsushita laboratory, and both of which are quite complex. These he calls "neural network-driven fuzzy inference" (*Neural-net-kudou-fuzzy-suiron*) and "Descent method fuzzy control" (*Koukahou-fuzzy-seigyo*), respectively.

Neural network-driven fuzzy inference, [see (Hayashi *et al.*, 1990)] is an approach for determining general fuzzy sets representing linguistic variables from sets of crisp examples of inputs and outputs. It assumes that these crisp data are already available. The source of the data is not described in detail, but presumably the intention is that they should come in part from specific examples of the cases for which the control-rule-base designer intended each rule to apply, and in part from other specific knowledge about the input-output relationships.

The discussions that follow assume a control device with several inputs and one output. If the number of outputs is more than one, this method can still be used simply by taking one output at a time and partitioning the rule base accordingly. Let m represent the number of inputs and r the number of rules.

The rules are considered in a way that may seem unusual at first, but we will see that this is somewhat similar to the Takagi and Sugeno concept of a fuzzy model, as discussed in a later chapter. Assuming that a given rule in the rule base is fired, then how can the crisp output value best be defined as a function of the crisp input vector? We are referring now to crisp values on the actual base variable spaces; therefore this function is altogether different from the function expression of the rule base considered in Chapter 5.

First r separate neural nets (call them NN-1, NN-2 . . . , NN- r) are constructed, each with m input nodes, one output node, and some suitable number of hidden nodes in between. These are trained by the BPN algorithm to represent the crisp functions just described by sets of examples, each example being a crisp input vector and a value for what the output should be in that case.

Additionally, one more neural net (call it NN-mem) is constructed with m input nodes, r output nodes, and again some suitable number of hidden nodes in between. This is trained to identify effectively the fuzzy sets for all input variables by training it on examples of typical crisp input vectors for which a given rule would definitely apply. One should note that directly speaking, what NN-mem actually learns is something that approximates the concept of m -dimensional fuzzy numbers.

Finally, all of the neural nets so far described (NN-mem, NN-1, NN-2, . . . , NN- r) are combined in one global structure with m inputs and one output. This is trained to identify effectively the fuzzy sets for the output variable by a modification of the usual BPN learning algorithm.

Obviously, this is a method with a fairly high degree of complexity. We have no intention of considering every detail of the method here, but the above description should give a basic idea.

The descent method fuzzy control approach came later [see (Nomura *et al.*, 1992)] and it is slightly simpler and more rationalized than the preceding method. We again assume m inputs, one output, and r rules. In this case, one constructs only one neural net with m inputs, r hidden nodes, and one output node; thus each node in the hidden layer corresponds to one rule. This network is trained using data similar to the crisp input-output data used with the earlier method, and by a modification of the BPN-learning algorithm.

Descent method fuzzy control is quite different from their earlier method in the sense that Nomura *et al.* provide a simple way of explicitly stating the fuzzy sets learned by the network. These are assumed to be symmetric triangular fuzzy numbers.

As an example of neuro-fuzzy design, Hayashi considers a vacuum cleaner with a photo-sensor device installed in the intake. Inputs from this device can be

used in combination with knowledge concerning the characteristics of various types of debris to determine indirectly the amount and type of debris that the vacuum cleaner is currently picking up. However, the knowledge format is quite technical in nature and represents a rather complex, and certainly nonlinear, relationship. It would be quite difficult to define linguistic variables from this knowledge in the usual KE-based manner. Neuro-fuzzy methods provide a mechanism for attacking this particular type of circumstance in an application.

Anyone familiar with the history of neural methods is probably aware that some of the supposedly basic research has become highly technical; that is some researchers believe the only effective way to attack certain specific, technical, complex application problems has been to develop technically complex and highly specialized (using specialized structures and algorithms) types of neural nets, sometimes called hierarchical systems. The neocognitron is an example of this. It is probably fair to summarize the Japanese approaches to neural-fuzzy methods as something very similar to this. They use technically complex, highly specialized methods to attack those specific control problems where it is somehow easier to obtain crisp data than it is to express directly fuzzy sets as heuristics.

6.7. Genetic Algorithms in Fuzzy Control Design

One of the most recent shifts in interest in the Japanese research community is to combining machine learning or optimization by genetic algorithms with fuzzy control. Many papers have been published in this area starting in 1992 or 1993.

Unlike the neuro & fuzzy and neurofuzzy approaches just discussed, it is quite literally true that much of the research on combining GA with fuzzy control directly does attack the problem of tuning fuzzy controllers. It is perfectly reasonable to claim that these are true auto-tuning mechanisms for fuzzy control designs. Alternatively, one could present the use of GA with fuzzy control design as a sort of optimal fuzzy control. In any case, typically auto-tuning takes the form of adjusting membership functions (linguistic variable definitions), not in adjusting linguistic control rules, though there are exceptions to this rule. In the discussions that follow, we assume that control rules are not only defined but fixed and only linguistic variable specifications are subject to change.

Assuming the reader is familiar with GA methods discussed in Chapter 3 and classical fuzzy control design discussed in Chapter 5, it quite easy to understand how GA could be used for auto-tuning of fuzzy control. GA can form the basis of a widely applicable method of optimization, and tuning of a control device can be thought of as an attempt to optimize performance so there is a fairly natural match except for two problems.

The GA methods, as discussed in Chapter 3, work well for many types of optimization problems, but the solution spaces typically must be finite. How then

can GA be used to identify membership functions? Well, this is not really a very serious problem in that it can be overcome simply by making a few assumptions. First, in fuzzy control design one frequently assumes TFNs or some other specific type of fuzzy numbers for which each membership function can be specified by just a few parameters. Second, it is not unreasonable to discretize and place lower and upper bounds on solution spaces for each of these parameters. For example it may be sufficient to know the parameters identifying a given TFN only to the nearest tenth of a unit, and to assume that they must fall in specific ranges. These two assumptions are all that is necessary to make the overall solution space finite. In fact depending on how many assumptions we wish to make about the overall nature of each fuzzy relation, it is possible to make the solution space quite small.

The second, and more major problem is that GA methods also require a well-defined evaluation or objective function, but it is not obvious how to define such a function for tuning a fuzzy controller. Reviewing various papers we note many ways of solving this problem, some quite clear, others not so clear.

One of the best starting points in our opinion, is (Yamamoto *et al.*, 1993). We present here our own explanation of their approach. Assume for now a control device design with multiple inputs and outputs but whose overall design effectiveness can be measured by just one output of the overall system. Also, referring to Chapter 2, assume that one given test input function (the word input here means setpoints) is sufficient for testing transient response of the control device design on that one-system output. Finally assume that all membership functions (relating to all input and output linguistic variables) can be expressed by some list of m parameters, $(p_1, p_2, \dots, p_m) = p$, where each parameter takes only a finite set of values. We can then specify function:

$$J : p \rightarrow \mathcal{R}$$

as

$$J(p) = \int_0^k |E(t)| dt \quad (6.1)$$

where $E(t)$ is the actual (crisp) error in the system output t time units after the start of application of the given test input function, given the design identified by the parameters, and k is some reasonable number of time units by which time the system can be assumed to have settled quite close to a steady state. Obviously the objective is to minimize J subject to p .

It is necessary to elaborate on a few points here. First even if all of the preceding assumptions are accepted as they stand, a considerable amount of calculation is necessary to evaluate J . This involves simulation of the plant (using some type of model as discussed in Section 6.4), simulation of the control device (which can be

easily done because the control rules are assumed fixed and we are calculating for a given set of parameters at this point), and numerical integration. The computer can be programmed to make these calculations fairly easily if the designer has good programming skills and knowledge of systems science, but the calculation times will obviously be rather long. What is more difficult is to make general statements about the nature of this function. J. Yamamoto *et al.* state that it is a highly irregular function with many peaks and valleys, but they provide no true analysis, only empirical evidence. If what they say is generally true, then this is a good additional justification for using GA methods, because properly applied these are good for avoiding being trapped by local minima.

The second point is that it is quite easy to understand how this method can reasonably be generalized to deal with slightly more complex objectives. For example if multiple system outputs (which from the point of view of the control device may be inputs) are necessary to characterize overall system performance, then a value can be calculated based on Eq. 6.1 for each of these, and the results can be combined as a weighted sum, with the weighting reflecting the relative importance of the variables. Similarly if it seemed desirable to use more than just one test input function to characterize fully transient performance relative to some variable, then this can be handled in the same way. It is even possible to add a weighted term for steady state error if it is desirable to consider that as a criterion in addition to transient response.

There is evidence that using GA methods as in the approach just discussed is the best approach known for auto-tuning of fuzzy control devices. This method can also be called optimal fuzzy control. As promising as this may be, we must still carefully consider its rationale. What is the basis of autotuning or optimal fuzzy control? To be honest it is a sort of compromise. Typically a major part of the design (usually the control rules) is still based on intuitive knowledge, so auto-tuned fuzzy control is still based much more on heuristic, naive physics knowledge than is conventional control. On the other hand another major part of the design (usually the identification of the linguistic variables) is accomplished by applying an optimization technique to a model of the plant, and thus is model-based. In terms of rationale this places auto-tuned fuzzy control near the midpoint, between conventional control and classical fuzzy control. However in terms of structure and range of applicability, it is much closer to classical fuzzy control.

6.8. Dynamic Compensation in Fuzzy Control

One focus of the sample fuzzy control designs in Part II compare the effectiveness of various forms of dynamic compensation in fuzzy control. As much as we would like to claim that this is a totally unique idea, it is not, but the results are quite

unique because there have been surprisingly few attempts at a careful comparison of the relative effectiveness of using various forms in the context of fuzzy control.

This short section points out a few other references that mention concepts relating to dynamic compensation. As noted earlier much of fuzzy control has used one specific form of dynamic compensation, namely PD from the time of the first laboratory control devices by Mamdani *et al.*, but this point is rarely stated explicitly. Perhaps some researchers are not anxious to point out similarities between fuzzy and conventional control in a way that shows the debt owed to the conventional theory.

A paper that specifically discusses a concept of fuzzy proportional-plus-integral (PI) control is (Siler and Ying, 1989). They consider the very interesting question of what fuzzy logic operators would make a linear fuzzy PI controller behave identically to a linear conventional PI controller. The answer to the question was something that they called “mixed fuzzy logic,” and they felt that this should have some impact on the overall question of operators in fuzzy logic (the question discussed for example in Section 6.3). We cannot find further reference on this in the literature.

A paper specifically discussing fuzzy PID control is (Chen *et al.*, 1993), which considers specific techniques for analyzing in conjunction with a fuzzy plant model the stability of any fuzzy PID controller. They also consider aspects of their design approach that can be used with other types of plant models.

We are not saying that fuzzy PID or fuzzy PI control is rarely used. Approaches qualifying as fuzzy PID control have become quite numerous, but the subject of dynamic compensation is rarely considered in a comparative or general way in the world of fuzzy control; often it is not even explicitly discussed. Most of the literature on fuzzy control ignores or avoids these topics; fortunately the few exceptions to this trend, such as those just discussed, report exceptionally good research.

PART II

Case Study on the Utility of Dynamic Compensation in Fuzzy Control

CHAPTER 7

Miniature Steam Engine as a Test Plant

The design of control devices is, or at least should be, a practical science, regardless of whether conventional or fuzzy methods are used. For this reason, any comparison of methods must in the end be based on a realistic sample application, which means that at least to some degree it should be a design to control an actual physical plant, which we call a test plant.

One must consider one's objectives carefully in choosing a test plant. When we say that it should be a practical example, we do not necessarily mean that it be part of a commercially profitable operation. In fact, to do basic experimentation on commercial applications is often quite impractical because these are typically either large scale industrial processes or products designed for mass production. It is often much more economical to experiment with plants intended to be used only in the laboratory. However, whatever test plant is chosen, it should share enough characteristics with real application plants so that others can be convinced that the control design principles one demonstrates for the test plant reasonably can be expected to apply to real applications as well.

Ideally, the control design should be tested on an actual physical plant, and certainly this makes the demonstration most dramatic, but it is quite common first to demonstrate a control method by simulating control over a model of a plant in the computer. The main reason that one may choose to simulate is that actually physically to control a plant involves purchasing, setting up, and debugging interfaces, digital/analog and analog/digital converters, and so on. Even for a small, laboratory-type test plant, this involves more expense and trouble than one may at first imagine.

If one does decide to demonstrate a control principle by means of simulation over a model of a plant, there are two basic sources for such models. One possibility is to base the model plant on a system that can be easily mathematically modeled just on the basis of physical first principles. One reason that the inverted pendulum has been very popular with some researchers discussing fuzzy control is that it is an excellent example of a system that is very nonlinear, but still can be very easily modeled in this way. However, there is a disadvantage to model test plants of this type. They are in some sense too artificial. They contain complexity only where the complexity is designed in, as with the nonlinearity of the inverted pendulum. They

still seem too much simpler than the types of artificial mechanisms we use in everyday life to be completely convincing as realistic examples.

The second possibility is to base the model on a realistically complex physical device. This alleviates the problem of realism just mentioned, but it leaves a question as to how the model can be constructed. Any system defined on (and thus any model of) a real world phenomenon is necessarily a simplification to some degree. Therefore, what modeling technique would allow us to capture as much realistic complexity as possible? There is perhaps no general answer to this question, but two promising possibilities are fuzzy models and modeling by neural nets, as discussed in Chapter 6.

The physical test plant we discuss here is a miniature stationary steam engine, and the modeling method is based on neural nets. Chapter 7 examines the characteristics of the physical test plant and why we chose it; Chapter 8 considers modeling that plant.

7.1. Brief Introduction to Steam Engine Principles

Steam engine technology is primarily a product of the eighteenth century. The small, inefficient steam engines that existed before that time were only novelties, and any improvements made later to reciprocating steam engines were fairly minor by comparison to the progress made in that century. Though steam engines are obviously examples of complex mechanisms, they represent a technology with a relatively long history of engineering analysis and design, skillful operation, and even influence over economic and social changes.

Though most people associate the name James Watt with all early developments relating to steam engines, Thomas Savery and later Thomas Newcomen were actually the first to make progress toward reasonably practical steam engines, starting in about 1698. The greatest significance of James Watt's work, in the second half of the eighteenth century, was that he first studied the problems of steam engine design in a truly scientific way, and this led to very dramatic improvements in efficiency. Watt rationalized steam engine technology. Nearly the only thing that Watt was unwilling to do was to use high pressure, apparently because he was very concerned about possible explosions. Thus one of the few innovations left until the early nineteenth century was the invention of more efficient high pressure engines.

Steam engines are the only significant examples of external combustion engines. (Stirling engines having had little practical significance to date.) These are driven by the heat of combustion, but the combustion occurs in a boiler outside of the mechanical apparatus of the engine proper. To amateur mechanics of the late twentieth century, accustomed mostly to internal combustion engines, some aspects of steam engine design will be familiar, but others will be foreign.

Reciprocating steam engines, to which our discussion is limited, drive pistons by the expansion of steam in cylinders just as internal combustion engines drive pistons by the explosion of combustible gases within cylinders. However there are many differences in practical details.

First nearly all practical steam engines are double acting (or double action). Each piston is driven first one way in the cylinder, and then driven back also by steam. Incidentally, several types of valve mechanisms are used for porting steam into and out of the steam engine cylinders. The choice of a valve mechanism depends mainly on how much simplicity one is willing to sacrifice for the sake of higher efficiency.

Second most steam engines above a certain size have multiple cylinders as do most internal combustion engines above a certain size, but there are differences. While in both types of engines the use of multiple cylinders has a smoothing effect on the crank shaft rotation, this result is the primary reason for the practice only in the case of the internal combustion engine. After all, in a double acting steam engine every stroke is a power stroke as opposed to only one stroke in four for a four-stroke gasoline engine.

In the steam engine the main reason for multiple cylinders is to improve efficiency: Unlike the designs of internal combustion engines, in a multicylinder steam engine, the steam is fed through the cylinders in series, not in parallel. That is, the exhaust steam from the first cylinder is ported into the second cylinder in a "double expansion" (two-cylinder) steam engine. Triple expansion and quadruple expansion engines carry this even further so that, for example, in a quadruple expansion engine the fourth cylinder gets only steam that has already been through three other cylinders. This arrangement is found to improve overall efficiency by decreasing the effects of condensation.

A third difference is that, generally speaking, it is easier to design a steam engine for a high torque/low cycle speed application than it is an internal combustion engine, thus a steam engine must typically needs to be geared down less than an internal combustion engine. The most obvious reason for this difference is that in one case the piston is driven by an explosion, while in the other it is driven by an expansion of a pressurized gas.

Naturally, for an efficient external combustion engine, the design of the boiler is also important. A main consideration in this regard is designing the boiler with a large area of heat exchange surface between the water/steam chamber and the combustion gases. This may entail, for example, running a manifold of pipes through the water/steam chamber in such a way that the combustion gases on their way to the exhaust stack will flow through the insides of the pipes, while the outsides of the pipes are submerged in the water. This is called a fire-tube boiler, and boilers of this type were once much more common than water-tube boilers, for which the relative positions of the water and combustion gasses were reversed.

Clearly, the burner or furnace must also be designed for as complete a combustion as is possible of the coal or whatever fuel is used.

Another aspect of boiler design for external combustion engines is the concept of superheating. As we recall 212° F or 100° C is the boiling point of water under the assumption of 1 atmosphere of pressure. Because the objective of the boiler is to produce steam of a considerably higher pressure than 1 atmosphere, the boiling temperature is also significantly higher; the exact temperature depending of course on the pressure requirements of the engine, more or less in accordance with the perfect gas law.

Whatever the boiling point, if the steam is simply heated to that temperature and left at that, then it would begin to condense as soon as it left the boiler. It is undesirable for large amounts of condensation to occur in the cylinders or before the steam even reaches them because this effectively decreases pressure. A way to avoid this is to heat the steam considerably above its boiling point for the given pressure. This is done by passing the steam just generated into a second chamber or through pipes where it is heated to a higher temperature by further indirect contact with the combustion gases without exposure to water in the liquid state.

Many steam engines, such as those in early twentieth century steam-powered automobiles, were designed for recirculation of the spent steam to the boiler, usually first condensing it back into liquid form. They were not perfectly closed systems in this sense, but the idea was to keep the water replenishment needs to a minimum. In other cases, the spent steam was exhausted out of the same stack as the smoke and combustion gases in such a way as to increase the draw of the stack.

7.2. Steam Engines as Control Problems

Much of the fascination of steam engines relates to their operation. Certainly in practice, the efficient operation of a 19th century steam locomotive, for example, or any other large scale steam engine was a highly technical process, and required more skill than the operation of many types of equipment used today. But even in its basic elements, steam engine operation provides an interesting control challenge, whether manually or automatically controlled. Specifically, steam engine operation requires feedback control on at least two variables with two quite different time scales.

Engine output speed or power can be controlled in the short term by a throttle. Unlike an internal combustion engine where the throttle adjusts the amount of air/vaporized fuel mixture that is made and drawn into the cylinders for combustion, the steam engine throttle is simply a valve that can be used for partially constricting in varying degrees the flow of steam from the boiler to the engine.

But in order for the throttle to continue to be effective in controlling the engine speed, the boiler pressure must be maintained in an acceptable range. Because the

boiler must be viewed primarily as a thermal device, and because thermal devices typically respond quite slowly to changes in their inputs it is normally very difficult to maintain the pressure at or even near a precise setpoint under varying conditions of engine load and throttle setting. It is possible to heat the boiler continuously in such a way that there would always be an excess of pressure and allow the excess to be vented off by the safety valve. This would make it much simpler to maintain a nearly constant pressure, but it would also obviously be too inefficient to be practical in most cases. How then, does one adjust heat input to the boiler (by adjusting fuel inputs, air inputs, or both) so as most effectively to maintain the proper boiler pressure? The answer probably depends primarily on the circumstances.

Where possible anticipation may play an effective role; for example if a steam locomotive engineer knows that the train is approaching a hill and current boiler pressure is only about average, then (with the help of the fireman) he/she would start building up a "head" of steam now. Similarly, minutes before approaching a downgrade, provided that the pressure was not already low, the engineer may allow the boiler to cool down slightly. In cases where accurate anticipation is possible, this approach (combining feedforward and feedback elements) may be the most effective way to operate a device with properties similar to those of a steam engine, but it is not always possible.

Where one cannot use anticipation, it is necessary to rely more or less on feedback alone. However, in this case, true multivariate control is likely to be quite beneficial. If the pressure is currently right at the setpoint, but the speed has just started to fall, then it is clearly appropriate to make a small upward adjustment to the boiler heat input. In this case, information, not just about the pressure, but also about the speed was useful in making the decision on how to adjust the heat.

It is also conceivable, for somewhat different reasons, that throttle adjustment might benefit from a multivariate view. If the pressure is currently high, then smaller adjustments are necessary for controlling speed than would be the case if the pressure was low.

To summarize the control of a steam engine is as a challenging, multivariate, feedback control problem with very significant time delays in one of the controlled variables. This is particularly true where we assume that power needs cannot be anticipated in advance.

But there are other important characteristics of steam engine control as well. Some aspects make it ideal as an example of heuristic control design. First the time scale for feedback is within a range that humans can deal with manually. It does not require reaction times in the low millisecond range, and in fact there is usually enough time to think somewhat carefully about what one is doing. On the other hand, the time scale is not so slow as to make the operation boring. Secondly, steam engines are not new devices, and there are traditions about how to operate them.

Finally, provided that one concentrates on the basic elements as described above, especially as applied for a small steam engine in the laboratory, then the system is not so technically complex as to confuse the intent of the investigation. Steam engine operation is complex enough to be very interesting as an example, but need not be so complex that one becomes lost in the trivial technical details.

In all these ways, steam engines are well suited for use as test plants in demonstrating fundamental control design principles. Therefore, it is not surprising that steam engines have had a significant role in the history of automatic control.

Watt developed the flyball governor for his steam engines. This mechanical device was designed to control output speed by automatically adjusting the throttle. It was a promising start even though it was in no sense multivariate, and it was very difficult to make it function in a stable manner. However, in later centuries, attempts to analyze governors could be described as the beginnings of conventional control theory; for example see (Maxwell, 1868).

Just as significantly, a small steam engine was also the test plant used for the first complete demonstration of fuzzy control, as discussed in Section 5.8. Mamdani and Assilian did not explain the reasons for choosing a steam engine in any of the major references on their work, but it seems likely that their thinking corresponds to the preceding discussions.

As a final practical point, if one intends to use a steam engine as a test plant, demonstrating control principles as they relate to the operation of the engine itself, then it is probably best to use a stationary one. Vehicles powered by steam engines may be more interesting to watch, but unless the goal is to demonstrate control over the overall operation of such a vehicle, it adds unnecessarily to the complexity of problem.

7.3. Our Miniature Steam Engine and Its Modifications

Until recently a small British firm, Mamod Ltd. of Slough, Berks, manufactured small working model steam engines as educational toys. The recommended fuel for these was tablets of solidified alcohol. All of the popular models were steam-powered miniature vehicles, such as steam roadsters or steam tractors, but they did make a few models of stationary steam engines. We obtained one of these for our test plan.

While Mamod's products are true working steam engines in miniature, they are very simple types of steam engines: single acting, single expansion, (and therefore relying heavily on a flywheel) and using a simple slide valve. On the stationary model we use, the water/steam chamber of the boiler is a brass cylinder about 1.8 in. in diameter and 4.5 in. long. The burner pan is located under this horizontal cylinder and vented so that there is a reasonably good flow of the combustion gases over the brass of the lower half of the cylinder. The cylinder has

a water level view port on one end, and it comes with a safety valve and a steam whistle screwed into the upper side. Water can be replenished between runs by temporarily removing the safety valve, then funneling through that hole; there is no provision for superheating.

A fine copper tube carries the steam from the boiler to the valve and throttle mechanism. The throttle varies the restriction in the flow of steam by adjusting the alignment of the slide valve. It allows the steam engine to be reversed, but in practice it does not work quite as well in one direction as it does in the other. There is a simple cam effect that allows for a complete stop plus three other control notches in each direction for the throttle. The cylinder and piston are brass. The cylinder has only a 0.35-in. external diameter, and it is 1.4 in. long. By our measurements there seems to be a 0.25-in. bore and a 0.75-in. stroke. There tends to be almost no steam leaking from the piston, but there is some degree of leakage around the slide valve and also considerable condensation. The steam that does not escape or condense is fed back through two other fine copper tubes to a stack. This is probably mainly for decoration, though perhaps it helps a little in preventing steam burns when children operate the engine. The stack has nothing to do with the combustion exhaust gases, which go directly through vents at the side of the boiler.

Though the Mamod stationary engine is a very simple type of steam engine, it is nonetheless, by any technical definition, a true steam engine. There is no reason to doubt that at least the basic elements of its operation are just the same as for any other steam engine.

However, there were still problems in using it as a test plant for our purposes. These mostly related to measurement and control:

- There was no provision at all for adjusting the heat input to the boiler
- There was no way of measuring the boiler pressure
- The throttle adjustments seemed a little too rough for our purposes. There were just too few control notches, and the short handle made it difficult to adjust smoothly even for those
- There was no way to measure engine output speed accurately
- There was too little load on the engine. The flywheel just spun freely, driving nothing

Fortunately it was possible to overcome all of these problems with various modifications to the apparatus. Before going into the details of these matters, one general point should be emphasized. Compared to conventional control system design, we are less constrained in our methods of measurement and adjustment. We have no need here for a linear plant model; therefore it is not strictly necessary to make the methods of measurement and adjustment as directly proportional as possible. A reasonable degree of care is still required in the control apparatus design, but the design is not strictly constrained.

For adjusting the boiler heat inputs, there are three basic approaches possible:

- Adjusting the input of fuel
- Adjusting the input of air (oxygen)
- Manipulating the heat transfer

The first possibility is extremely difficult to control in the short term in the case of solid fuel, so it would probably require the change to a different form of fuel; since this would be a major modification, it seemed impractical. The third possibility could be accomplished for example by sliding a plate in between the burner pan and the water/steam chamber of the boiler. This is fairly easy to do, but any method that intentionally makes the boiler inefficient seemed too artificial.

That left adjusting the air input, which has two subpossibilities, adjusting air still leaving the burner as a natural convection-type system or adjusting air with a burner converted into a forced air system. In the original design there were actually several different vents through which air flowed into the burner, so it would be very difficult to seal all of them in a manner that was easily adjustable. For that reason conversion to a forced air system seemed most practical, though it was necessary to take care to avoid a blast furnace effect.

The conversion involved a small fan and a little sheet metal work, as illustrated in Fig. 7.1. First the lower part of the burner was sealed off on all sides with sheet metal except for one vent. We were able to purchase a very small fan designed for cooling relatively small integrated circuit devices. This was installed by means of a small duct (made from an empty beer can) added to the remaining vent. The fan was designed to run on 12 V DC. We found that by connecting it in series with a $500\ \Omega$ variable resistor (potentiometer), it was possible to control the speed of the

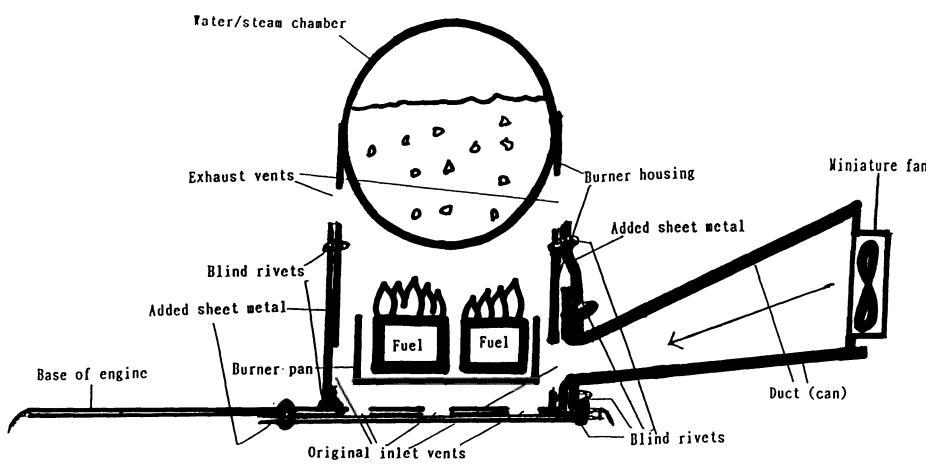


Figure 7.1. Cross-sectional sketch of modifications for heat adjustment.

fan very easily. For the sake of a constant power supply, we used an AC adapter rather than batteries. The potentiometer was mounted conveniently on the control panel with a graduated knob. It works very well, and allowing for a delay of a few seconds, we can even control the engine speed to some degree by the heat control knob alone without touching the throttle.

The pressure measurement required considerable trial and error. We refer here to gauge or relative pressure (the pressure above the atmospheric pressure) rather than absolute pressure because this is the custom in boiler operation. The Mamod stationary steam engine functions in a range from approximately 8–15 PSI, though we did not know this at first. This is apparently about the pressure range that drove James Watt's steam engines. Most available pressure gauges do not start registering accurately until at least about 15 PSI, so these seemed useless. On the other hand even 8 PSI is much too high a pressure for the U-tube method to work, and even if it were not, that is not a particularly convenient method. Gauges designed with reset buttons also do not work well for continuous readings even if we tape down the button. Furthermore the construction of the gauge must permit it to function well under high temperatures and dampness; this is not true of many types of gauges.

We found a heavy brass tire gauge without a reset button that registered quite accurately starting at about 3 PSI. With more experimentation, we removed the top of the steam whistle and connect it securely to the gauge by means of pneumatic hose. The gauge was therefore also connected to the control panel. However this created another problem because heat loss from the hose was great enough to cause a significant change in the overall system dynamics: The engine lost a relatively

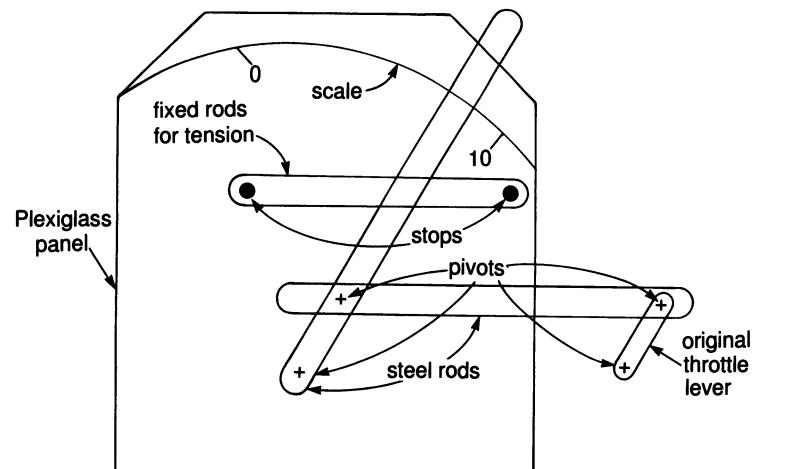


Figure 7.2. Sketch of modifications to the throttle mechanism.

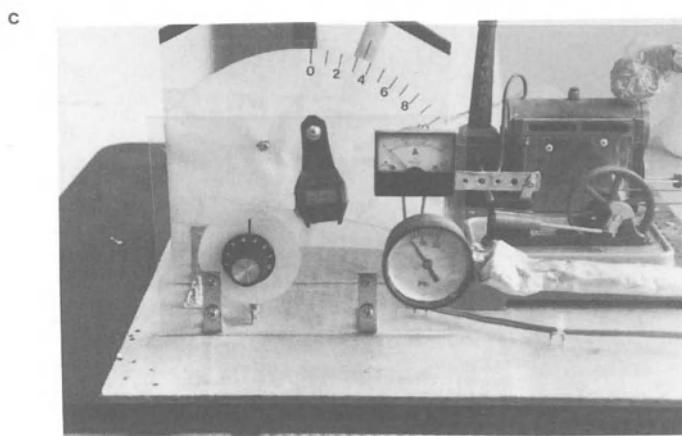
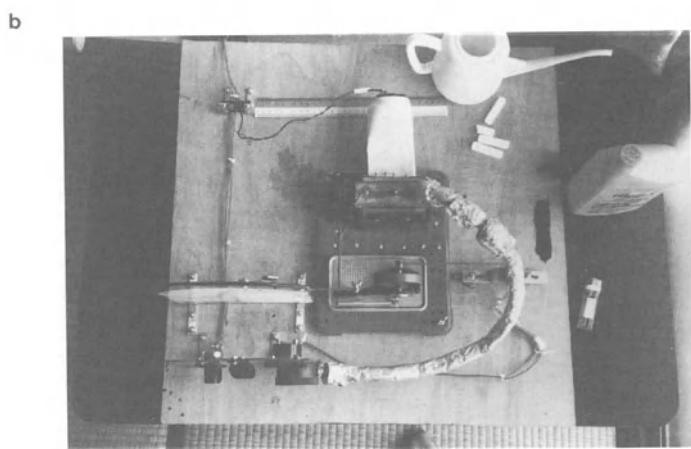


Figure 7.3. Modified steam engine; (a) front view, (b) top view, (c) close-up of control panel area.

significant amount of power from heat loss through the length of uninsulated hose. This was overcome by insulating the hose with a foam insulation product and with aluminum tape.

For more precise throttle adjustment, we designed a lever mechanism as illustrated in Fig. 7.2, constructed of metal rods, fasteners, and a clear plastic panel. This was designed to allow for a finer adjustment but at the same time to protect the somewhat delicate throttle from being forced too far in any direction. All connections were made to have as little give as possible.

The problems of speed measurement and insufficient load were corrected by adding a miniature DC motor, then connecting that to the flywheel by a pulley system. The motor acts as a generator producing a small direct current that varies with speed. It works fairly well to measure the voltage produced, but measuring the current by an ammeter connected in series with a small resistor works better still; the ammeter is mounted on the control panel. The pulley band is a very fine steel spring designed for the purpose, so therefore there is almost no aging effect even in the presence of heat.

Better approaches to speed measurement are possible: We considered basing our approach on one of the microprocessor-based bicycle speedometers now available, but this required additional expense and more difficult modifications, and that type of speedometer is usually based on discrete time intervals. The motor and ammeter system seems to work well enough for our purposes.

The entire assembly is mounted on a lacquered plywood panel; Fig. 7.3 shows various views. The control panel previously mentioned is a small clear plastic panel mounted in front of the panel for the throttle control mechanism. Thus all controls and measurements could be clustered in one area, on the left side in the front view. This made it easier to videotape the operation for purposes of constructing the model. Also for this reason, a stop watch is mounted on the control panel. It is now interesting to operate the steam engine, though some disassembly is required each time it must be serviced in preparation for another run. The method used in constructing a model of this modified steam engine is the subject of Chapter 8.

CHAPTER 8

Simulation Model of the Test Plant

Chapter 8 discusses the process of developing a model of the physical test plant. We use the word “model” loosely to describe any effective way of accurately approximating the behavior of the test plant in the form of software, such as a computer program.

To repeat some points from the two previous chapters, there are two reasons why such a model is important. First although we preferred to demonstrate and compare fuzzy control design concepts we consider by implementing them directly to control an actual complex physical device, various limitations force us to simulate control within the computer. However, we added the requirement that the model should capture as much of the real complexity of the actual device as is reasonably possible. Second constructing a model of the test plant makes it more practical to add an auto-tuning element to the design process of the various fuzzy controllers, as discussed in Chapter 9.

After considering various approaches to modeling the steam engine, we settled on a type of approximation neural net structure. Using models in the fuzzy control design process is discussed in Section 6.4, but we consider more carefully the question of modeling approaches in the first section of this chapter and the reasons for our particular approach in the present research. Later sections consider in greater detail the process of developing the specific model and evaluate its performance.

8.1. Concerning Approaches to the Modeling of Plants in Fuzzy Control Applications

If one does find it useful to develop a plant model for a fuzzy control application, then one is almost completely unrestricted as to the type of model. First, there is no pressure to make the model linear or to limit the number of variables. Furthermore, there are few limitations on the basic approaches that can be used in developing or in expressing the model. It is very possible to use a conventional type of mathematical model, but it is equally possible, and often more appropriate, to use approximation neural nets or a fuzzy modeling approach.

For the application presented here, we briefly considered the possibility of using conventional mathematical models of the steam engine. As noted previously

some researchers discuss demonstrations of fuzzy control principles based on test-plant models of a conventional, crisp nature, but these are typically such systems as inverted pendulums or trailers that need to be backed up, systems that are nonlinear, but are simple to analyze by conventional means. Clearly, there is a value in sometimes demonstrating fuzzy control in this way, but it is attacking a type of problem that is somewhat artificial, and it is not what we wanted to do here.

It is possible to attempt to develop a fairly complex model of a steam engine by conventional means, but that would require careful analysis of many physical principles and careful examination of many minute details of the actual steam engine to create a precise model with sufficient accuracy to predict reasonably well the actual performance. This would be an overwhelming task, and there is no incentive to attempt it here. There are also disincentives to using conventional models even if feasible. We prefer to emphasize that the various approaches we consider to fuzzy control are all applicable without the need for conventional modeling.

In principle, supervised machine learning by neural nets is based on the concept of a structure that generalizes effectively by learning to approximate correct results from a collection of sample data. Faced with such an explanation, one might reasonably assume that this is very much a data- driven approach. The rationale of fuzzy modeling is best understood in the context of the informal approach as described in Chapter 6. Thus, if this approach is used to understand the principles, then fuzzy modeling is to be viewed as based on the concept of making heuristic, naive physics sorts of statements about the system to be modeled, and then interpreting those statements (much as in fuzzy control) by means of approximate reasoning using linguistic variables and a fuzzy inference algorithm. Faced with this description, one would be very reasonable in assuming that it is entirely a knowledge-driven or heuristic-driven approach. Thus in terms of simple statements of principle or rationale, the two approaches are fundamentally different.

But when actually applying these approaches to the task of modeling a complex system, one quickly discovers that the neural net approach is driven by heuristics as well as data, and the fuzzy modeling approach is usually driven by data as well as heuristics. To generalize most model-building processes begin with hypotheses, then verify or refine those hypotheses on the basis of experimental data; typically this is the only generally efficient process for building accurate models. If we think of model building and theory building as somewhat synonymous, then it makes sense that most successful model development processes are something quite analogous to the scientific method. This is one way of explaining the fundamental similarity in rationale of nearly all modeling approaches regardless of apparent differences.

The concept of models based entirely on heuristics as related earlier seems appealing, and in some circumstances an informal model of this type can be useful. On closer reflection a fuzzy model based on heuristics alone falls short when it is necessary to model a complex system, as discussed in Chapter 6. The process of

designing a device we intend to build and the process of modeling a natural or otherwise previously existing object are quite different in spirit. When designing a device, we may sometimes be comfortable working almost entirely from heuristics; when attempting to model an existing phenomenon, the model builder generally cannot rely on heuristics alone, without data to verify or to modify those heuristics.

Only a few sources on applications of fuzzy modeling discuss totally heuristic-based (informal) models as a practical alternative. One example is (Deboeck and Cader, 1994), who discuss fuzzy models of the stock market based on the knowledge of experienced traders. Most of the literature emphasizes methods for using data in developing the (formal) fuzzy model. In fact the most formal approaches of some long-term advocates of fuzzy modeling, such as Takagi and Sugeno or Filev, emphasize the use of data almost exclusively.

Although several general structures have been proposed for fuzzy models, the structure suggested in the early Takagi and Sugeno approach [see (1985)] is still representative and a good place to start. It is sufficient to consider a system with only one output because in a multiple-output system, each output can be estimated by a separate structure. Therefore let us say that the actual relation between multiple inputs $\{x_1, x_2, \dots, x_m\}$ and the one output y can be represented by the function:

$$y = f(x_1, x_2, \dots, x_m)$$

Further suppose that a collection of experimental data exists, but also that these data are somewhat noisy due to imprecision in measurement or other factors.

Takagi and Sugeno claim that generally this input-output relation can be approximated very effectively by a collection of linear functions and a collection of fuzzy sets with piecewise linear membership functions. Specifically, they consider a collection of rules of the following format.

Rule #k: IF x_1 is t_{1k} AND x_2 is t_{2k} ... AND x_m is t_{mk}
THEN $y = p_{k0} + p_{k1}x_1 + p_{k2}x_2 + \dots + p_{km}x_m$,

where $t_{1k}, t_{2k}, \dots, t_{mk}$ are various linguistic terms related to the input base variable space by fuzzy sets (relations) with piecewise linear membership functions. (Obviously the same term may appear in more than one rule.) An example follows:

Rule #3: IF x_1 is medium_big₁ AND x_2 is small₂
THEN $y = 2.20x + 0.26x + 1.3$

where:

$$\text{medium_big}_1(x) = \begin{cases} (x - 4.1)/4.7 & \text{where } 4.1 \leq x \leq 8.8 \\ 0 & \text{elsewhere} \end{cases}$$

and

$$\text{small}_2(x) = \begin{cases} (7.1 - x)/7.1 & \text{where } 0.0 \leq x \leq 7.1 \\ 0 & \text{elsewhere} \end{cases}$$

The Takagi and Sugeno model is a similar concept to the one used in the neural network-driven fuzzy inference approach to neurofuzzy, which is no doubt inspired by it (see Chapter 6). However whereas Hayashi *et al.* use neural nets, Takagi and Sugeno use something closer to conventional statistical techniques to obtain the various parameters from the data; these include both the coefficients defining the linear functions and the parameters identifying the piecewise linear membership functions. Most advocates of fuzzy models claim that while neural methods often capture system behavior effectively, the distributed representation is almost useless in leading to new (human) insights on the nature of the system. Fuzzy models on the other hand both capture system behavior and lead to new insights.

In most applications fuzzy models are based on data as well as heuristics, and most neural net applications are based on heuristics as well as data. Any process that attempts to develop a model on the basis of raw data without using previously existing knowledge about the system is hopelessly inefficient in the case of most complex systems. In Chapter 3 we mention that the distinction between machine learning and knowledge engineering is not a crisp one: Even if a given application is based primarily on a machine-learning technique, some human knowledge is still necessary, at the very least, to identify the basic structure of the system to be considered.

If one carefully examines the process of developing a typical neural net application, it soon becomes obvious that the use of the developer's knowledge about the system is very significant, usually going far beyond the identification of the basic system structure (knowledge of the application area, not of neural net methodology, although obviously that too is important).

A discussion of all the ways in which application knowledge is used in developing neural nets would be quite tedious, but we can briefly consider two of the most obvious ways. First, nearly every recent practical guide to neural methods [see especially (Kung, 1993), (Müller and Reinhardt, 1990), and (Deboeck and Cader, 1994)] discusses the need for preprocessing in most types of supervised learning applications. That is, usually the raw inputs are inadequate as a basis for training the network, and therefore these inputs need to be manipulated in some way so that they provide more significant information for the decision or classification task of the network. This concept of preprocessing clearly implies that the application developer has a great deal of knowledge about exactly how data should be manipulated to create variables with useful information content, and that in turn implies a very significant level of previous knowledge about the application area.

Application knowledge is also used in most supervised learning applications: selective set of training examples. Unlike most statistical analyses, where there is an effort to avoid introducing biases into the data used, most guides to neural methods suggest that one should carefully (though apparently subjectively) consider which data would form the most typical examples of system behavior and therefore belong in the training set.

In reality, most neural net applications are quite tedious to develop because they usually require considerable experimentation before satisfactory results are obtained. This is the case even for the relatively small application we consider. But as with any other process of experimentation, the amount of effort needed would be very many times greater if one did not begin with some previous knowledge, or at least with some reasonable hypotheses.

Because we consider the process of developing a fuzzy model and the process of developing an approximation neural net to be fundamentally quite similar, the choice between the two approaches is based on practical points. Furthermore the claim that fuzzy models are true models because they lead to greater human understanding of the system they model is not important to us, partly because we use an auto-tuning mechanism. Finally, the literature is not helpful in determining which approach results in more accurate estimates of system behavior or which is easier to develop. Some papers report results of experiments comparing the performance of fuzzy control devices with neural based machine learning of control, but such a comparison is slightly different from comparing the ability to approximate system behavior. Papers specifically concerned with fuzzy modeling typically compare performance to conventional statistical models.

Therefore, in the end, the decision to use a neural approach was somewhat arbitrary. Our justification is the hunch that if the basic structure of the input-output relation is quite clear, then fuzzy modeling is quite easy to apply in practice and may be superior in performance; if the overall nature of the relation is unclear, then a neural approach is more practical even though it involves considerable experimentation. In steam engines the nature of input-output relations is unclear.

Having decided on a neural approach, the next step is to decide more specifically on which neural approach. It is reasonable to divide applications-oriented research with artificial neural nets into two broad categories; the first is research into ways of extending the neural techniques and architectures themselves; the second is research that applies neural methods strictly as a tool in research with some other aim. The work of the present chapter is clearly of the second category.

One implication of this is that it is appropriate for us to draw most of our information on neural methods from general, textbook-like sources. However even this is not an altogether easy task. Neural net methodology has developed so rapidly since its revival in popularity during the 1980s that it now takes considerable time just to glance through all of the structures and techniques considered fairly standard; some recent books called introductory are nearly 500 pages long. Furthermore many issues are not yet settled in this field, so there is a surprising degree of disagreement from book to book on what are the standard approaches; Part of the reason is that authors on neural nets come from diverse backgrounds; in addition there are major differences in the types of applications emphasized.

Those working in the first research category sometimes criticize those in the second for overusing some neural approaches while neglecting others that are

potentially useful. In particular BPNs are often cited as overused; however in the long run people usually know what is working well. If the BPN approach were not very useful, then application developers would have a strong incentive to use other approaches. Although Kosko and others are investigating more generalized ways of joining neural methods with fuzzy set concepts, it continues to be true that most uses of neural methods as part of an actual fuzzy control application are based on the BPN approach or minor variations thereon. Therefore with only slight misgivings we decided to experiment almost exclusively with BPN-type methods; details are discussed in the following section.

8.2. Details of the Neural-Based Model Design

A more specific way of stating the goal of the work discussed in this chapter is to say that it is to find an effective way of predicting the boiler pressure and the engine speed (as measured by the electrical current produced) of our small test-plant steam engine under various conditions. For given initial conditions and a given schedule of second-by-second settings of the heater control and throttle, what will be the boiler pressure and the engine speed at each second? This section considers the actual process of developing an approximation neural net to predict those outputs in some detail. We explain several aspects of the neural structures and algorithms used and the source of the training and testing data.

The approach we use is a software implementation of what Kung [see (1993)] would call a “nonrecurrent deterministic temporal dynamic model” of a rather simple type in that it is quite close to a standard BPN approach. Effectively time-delay elements exist only in the input layer; actually, the time delays are handled by a separate structure prior to the preprocessing, as explained later.

We first tried what Kung calls an all class in one network (ACON) structure, but we found that a one class in one network (OCON) structure was a much more sensible way to proceed in this application. In our case this means that we dealt with pressure approximation and speed approximation separately throughout the training and initial testing phases of the model-development process. The sets of hidden nodes connected to the two outputs were distinct, and effectively the input nodes were distinct as well because it was obvious that the preprocessed inputs should differ for the two outputs. Even different training examples could be used for the two networks as appropriate.

We limited ourselves to linear basis functions (LBFs), but for activation functions, we tried both the sigmoid and the hyperbolic tangent functions. Most sources suggest that the LBF–sigmoid combination is quite natural. However (Klimasauskas, 1994) indicates that recently the hyperbolic tangent is considered generally superior to the sigmoid by some researchers, but that this may vary with the application. Since experimentation to determine which produces better results

for our particular application added only slightly to the effort, we proceeded with it. One can easily confirm that the gradients in the case of the hyperbolic tangent function (used with the LBF) can also be calculated easily from the activations and the weights (in a similar way to the sigmoid). The variables should be scaled differently for hyperbolic tangent as opposed to sigmoid, and therefore one should take care in interpreting the results of the comparison between the two. In the end, we found that in this application the sigmoid was clearly superior. The sigmoid case both converged to a lower error value (even when adjusting for different scalings) on the training data and generalized better on the test data.

We experimented with networks without hidden layers and with those containing various numbers of nodes in one hidden layer. We found that a hidden layer was preferable in this application but approximately five nodes were sufficient in each case for the pressure network and the speed network.

One of the most critical elements of our neural based model was determining the proper input preprocessing to use. This required both heuristic knowledge about the steam-engine behavior and considerable experimentation.

One reason that preprocessing is desirable for many types of approximation neural nets is that ironically neural nets (even multilayered, real valued ones as with BPNs) are said to be rather poor at learning arithmetic except for the sort of arithmetic implicit in the basis function used. Thus if LBFs are used, then a BPN can be expected to be capable of learning linear combinations of the inputs, but it will probably be inept if the application requires it to combine inputs in a nonlinear way. This implies that the inputs to the network itself may need to be quite different from the inputs to the model as a whole, if the neural based model is to work well at all. Realizing this, just what should the preprocessing be?

It usually is infeasible to attempt to solve this problem by providing the network with every imaginable nonlinear combination of variables, and to expect the network to sort out for itself what is important and to zero out weights of the rest. Even if the number of basic model inputs is small, how would one know where to stop in this? For example, even supposing only two basic inputs, x_1 and x_2 , would it be sufficient to go only up to third order terms such as x_1^3 and $x_1x_2^2$, or would it be necessary to go higher? Furthermore what about x_2/x_1 or $x_1(x_2)^{1/2}$? The problem obviously becomes much worse as the number of basic model inputs increases. In dynamical systems, it is worse still because one needs to consider the values of the variables at various times in the past.

The only solution is to be selective, and this typically means selecting on the basis of both heuristics about the system and experimentation. In this way we finally selected six network inputs for the pressure network and six for the speed network.

Before continuing our description of the preprocessing, we must explain the approach for dealing with time. For these temporal aspects, there was a compromise for each of the outputs.

It becomes quite obvious to anyone who actually operates our small, simple steam engine with its homemade heat control apparatus that the pressure reading actually depends on both a long term and a relatively short term history. Relatively short term history means that heat adjustments made approximately 5 seconds earlier have a major influence on pressure, and as mentioned in Chapter 7, it is even possible to adjust speed (without using the throttle) by the heat control in this way. However this is not the complete picture: The actual range of pressures attainable in the short term depends on the total amount of heat inputs over the past 2 or 3 minutes; this is what we mean by long term history. Ideally our model would fully account for both factors, but a model requiring several minutes worth of past data seemed impractical to us; we therefore partially ignored the long term aspects. The only exception is that we require the model to keep track of how much time has elapsed since the fuel in the boiler was ignited, and this is provided as an input to the pressure network. For the short term aspects, the network uses data over the past 8 seconds, with 1 second as the elemental time unit.

The compromise for the speed network was of a different nature. The speed, as measured by electrical current, depends on conditions over the past 2 seconds. In this case 1 second is too coarse as an elemental time unit. For example there is a significant difference in the speed reading for a given throttle adjustment in the first half of a second as compared with the same adjustment in the second half of the second. However because all training and testing data were collected manually, even the second-by-second data were very tedious to collect; thus collecting data would be more arduous if we went to fractions of seconds.

We used a simple notation for describing the nature of each network input: Uppercase indicates the basic variable—H: heat input, P: pressure, T: throttle, and S: speed; a subscript indicates the time unit associated with the value, so S_{-2} indicates the speed reading 2 seconds ago. Sums over time are indicated by sigma, with the range in seconds, and a change in reading is indicated by a delta, with time units as the variable subscript; thus ΔT_{-2--1} indicates the change in the throttle setting from 2–1 second ago.

The following table summarizes throttle network inputs:

Notation	Explanation	Assumed Value Range
-6		
\sum_{-8}^P	Where the pressure was about 7 sec ago	30, 60
$\sum_{-8}^{-6} H - \sum_{-3}^{-1} H$	How much the input was changed from about 7 sec ago to about 2 sec ago	-30, 30
P_{-1}	The most recent pressure reading	10, 20

Notation	Explanation	Assumed Value Range
-2		
$\sum H$	Magnitude of recent heat input	0, 60
-7		
-2	The effect of pressure lost due to steam blown off recently. (At least that was the intention. Whether it is a useful input for that reason or for some other is uncertain.)	0, 24000
$\sum P^2 T$		
-7		
Time since ignition	Explained above	2, 10 (min)

The following table summarizes speed network inputs:

Notation	Explanation	Assumed Value Range
S_{-2}	Speed before recent throttle changes	0.035, 0.265
S_{-1}	Most recent value	0.035, 0.265
ΔT_{-2--1}	Recent change in throttle	-10, 10
ΔT_{-1-0}	Current change in throttle	-10, 10
0	Recent energy inputs	0, 12000
$\sum P^2 T$		
-2		
T_{-1}	Recent throttle magnitude	0, 10

A simple matter that can be dealt with as part of the preprocessing is normalization or scaling. The network inputs should be rescaled from their actual range of values to whatever range of activations is appropriate for the particular neural structure used. This is usually just a linear transformation, but there is no set answer to the question of exactly what range of activations is appropriate for a given structure. It is often suggested that network inputs and network outputs should be treated slightly differently in this regard. Also, Klimasauskas goes so far as to suggest that truly to optimize the model one should experiment with different activation ranges for each variable. However, for our application we felt it appropriate to treat all variables the same, and thus all activation ranges for the input and output layers are taken to be [0.1, 0.9] in the case of sigmoid activation functions, or [-0.9, 0.9] when we experimented with hyperbolic tangent functions. Naturally, in the final model, the network outputs must be rescaled to their actual values.

Putting it all together, Figure 8.1 shows the structure for training the pressure part of the model and Figure 8.2 for the speed part. The pressure part requires one or more sets of training examples that include second-by-second readings for heat, pressure, and throttle and the initial time elapsed since ignition. Each set of readings must be contiguous in time, and if more than one set is used, then this must be indicated in the data file with an initial time reading for each. In each set the first 8 seconds of data are used for priming the history tubes, so training begins only at

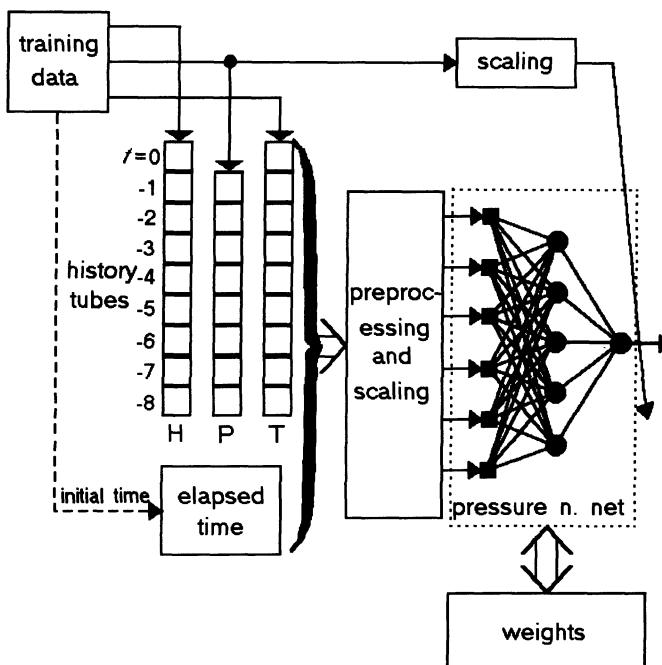


Figure 8.1. Schematic of pressure neural net training.

the ninth second. Similarly the speed part is trained by sets of readings of pressure, throttle, and speed. However this does not require keeping track of elapsed time, and only 2 seconds are needed to prime the history tubes. In each case test data had the same formats as training data, though not the same data.

These data were collected in the following way. A video camera was mounted on a tripod in such a way as to provide a close-up view of the control panel of the steam engine as described in Chapter 7. The boiler was filled up to a set mark on the view glass with water of approximately 180° F, and three-and-a-half tablets of fuel were placed in the burner. The stop watch was started at the time of ignition, and the heat input was left at the lowest setting for 2–3 minutes; at that point the boiler pressure reaches about 9.5 or 10.0 PSI, so the engine could begin to function fairly smoothly. The camera was then turned on, and the heat input control and throttle were manipulated at various intervals. The camera was turned off after about 8 or 9 minutes when the fuel was consumed, so that the amount of heat produced was decreasing regardless of the heat control setting. Many such runs were made; most were used for informal examination purposes only. About five runs were sufficient for collecting all necessary training and test data.

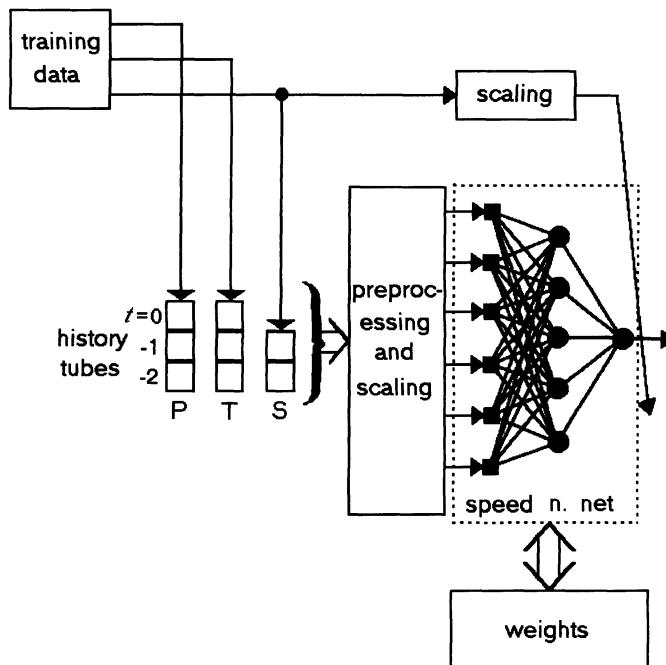


Figure 8.2. Schematic of speed neural net training.

The video was rerecorded on a standard-sized (VHS) videotape, which was analyzed using an ordinary video cassette recorder (VCR) and television set. By using the slow play and pause buttons on the remote control, it was possible to stop the tape just after the stop watch ticked off each second, then to record the readings on a handwritten data sheet.

We select parts of the various recorded runs as training data that well illustrate how pressure and speed react to the controls under various conditions. For the pressure network three parts of two runs were used for training with lengths of 26, 79, and 12 seconds, respectively. Because the first 8 seconds of each set were used for priming, this resulted in $18 + 71 + 4 = 93$ examples for the pressure training. For speed training five parts of runs were used, with lengths of 60, 15, 7, 97, and 16 seconds, respectively. Because only the first 2 seconds were necessary for priming here, this resulted in 185 examples. In each case a smaller number of examples was used as test data, and these were distinct from the training data. The question of evaluation is considered in the following section.

The initial weights were generated randomly. This corresponds to a state of complete ignorance before training begins. For each node in the hidden and output

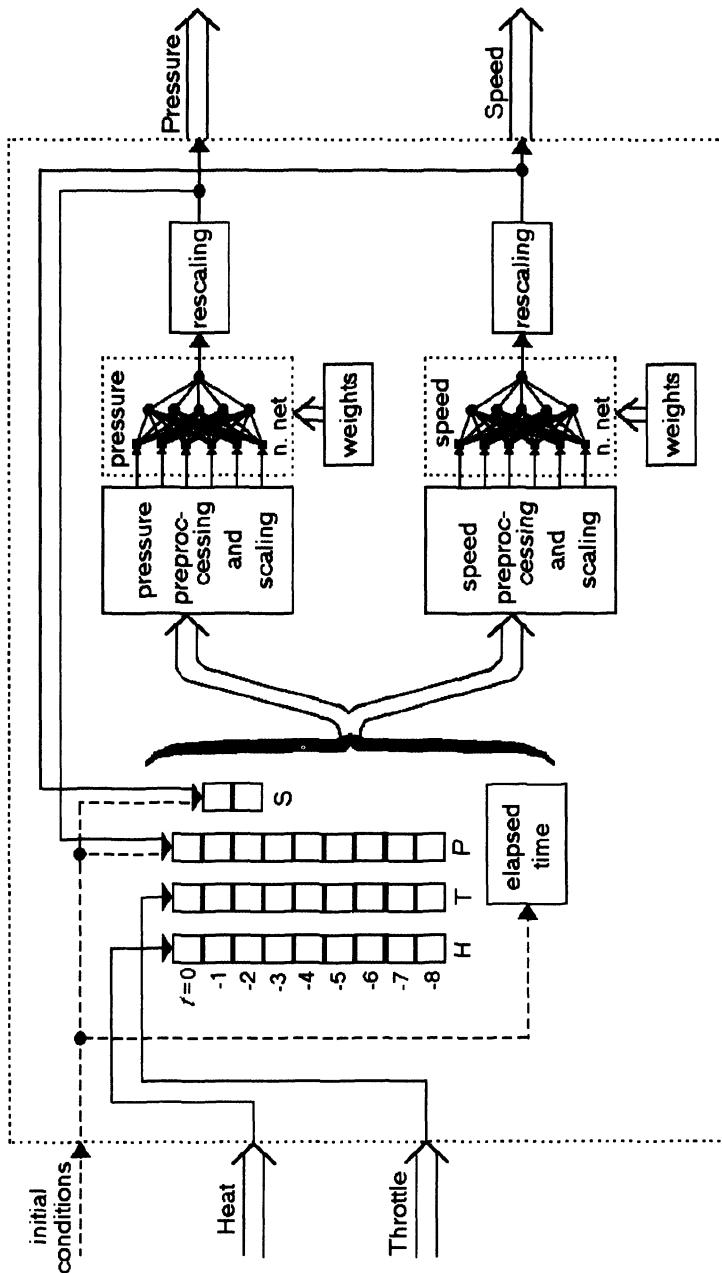


Figure 8.3: Schematic of completed neural steam-engine model.

layers, each of the initial weights associated with the inputs to the node were drawn uniformly from $[-2/k, 2/k]$ where k is the number of node inputs; this is standard procedure for BPNs. After training began, the weights were saved to a file at various stages to be ready for use in other training sessions, testing, and eventually the completed model.

Throughout most of the experimentation, we used a simple version of a batch-training strategy rather than an individual strategy. By this we mean that the weights were adjusted only after each complete pass through the entire collection of training examples rather than after each example. A batch strategy apparently nearly always offers more stable convergence than an individual one.

Some authors describe the danger of overtraining an approximation network, which results in lower errors on the training examples but leaves the network less capable of generalizing accurately. We were therefore careful to avoid this problem by saving the weights at various points during the training, then checking performance against independent collections of test examples. However, we found that for this application, overtraining was not a problem unless the number of iterations were allowed to exceed by far what would normally be considered convergence in the error value.

Once training reached a satisfactory stage, the completed model was ready for validation and use. The structure of the completed model is shown in Fig. 8.3. The evaluation of this model is the subject of the following section.

8.3. Evaluation of Model Accuracy

The model developed as we have described and illustrated in final form in Fig. 8.3 is perfectly acceptable for approximating the behavior of the actual steam engine discussed in Chapter 7. This does not suggest that it would have been impossible to optimize the model further than we did even within the context of the model structure we assumed; however we feel that the model's behavior is nearly indistinguishable from the actual device.

We note that in any case the precision of measurement is not extremely high in the actual device. Though we carefully selected the pressure gauge and later modified it to be as precise as possible for our purposes, it is still possible to read it only with an error of plus or minus 0.5 PSI. Because we normally operated the steam engine from 10–20 PSI, the error tolerance is a full 5% of the operating range. The speed readings as measured by electrical current are somewhat more precise, but as already mentioned, 1 second is coarse as an elemental time unit in the case of speed; this has an effect of as much as a 0.025 A error in extreme cases of throttle adjustment at the beginning or end of a second, though usually the effect is much less. Nonetheless in an operating range of from about 0.035–0.265 A, this is very significant. Model performance should be evaluated in the context of error toler-

ances of the actual device, and there may come a point when it no longer makes sense to attempt to optimize the model further unless one is willing to invest more in hardware to make the actual device more precise.

It is sometimes useful to make a distinction between testing and validation in the process of developing a model based on supervised learning, and Klimasauskas is careful to emphasize this point. The purpose of test data and the testing process should be to evaluate the progress of the model at various stages in training or when attempting to modify the model in some other way. Validation on the other hand is intended as a final check on the overall quality of the model once assumed complete.

With this in mind we decided to make a completely separate run of the steam engine, to collect data from a part in the middle of the run when moderate adjustments were being made to the controls, and use those as validation data. When we evaluated the model in this way, we used the complete structure shown in Fig. 8.3. That is once the initial conditions are set, the model must rely completely on its own outputs for historical data on pressure and speed. This is quite a strict test for a dynamical model if the test is continued for many time periods. The part of the test run during which controls were being adjusted with moderate frequency was about 1.5 minutes, so we based the validation on an even 90 seconds of actual simulation, not including priming.

The results are shown in Table 8.1. The percentage errors are calculated as a fraction of the overall operating range, which is taken to be 10.0–20.0 PSI in the

Table 8.1. Sample Performance of Steam Engine Model

Time	Heat	Throttle	Actual	Pressure			Speed		
				Predicted	%Error	Actual	Predicted	%Error	
Priming Data									
5:32	0.0	3.3	14.5				0.145		
5:33	0.0	3.3	14.5				0.145		
5:34	0.0	3.3	14.5				0.145		
5:35	0.0	3.3	14.5				0.145		
5:36	0.0	3.3	14.5				0.145		
5:37	0.0	3.3	14.5				0.145		
5:38	0.0	3.3	14.5				0.145		
5:39	0.0	3.4	14.5				0.145		
Actual Test Data									
5:40	0.0	3.0	14.5	14.3	-1.51%	0.140	0.139	-0.33%	
5:41	0.0	2.5	14.0	14.3	3.06%	0.135	0.130	-2.33%	
5:42	0.0	2.2	14.0	14.3	2.98%	0.105	0.119	6.01%	
5:43	0.0	2.2	14.5	14.3	-2.01%	0.095	0.110	6.44%	
5:44	0.0	2.2	14.5	14.3	-1.98%	0.095	0.103	3.68%	
5:45	0.0	2.2	14.0	14.3	3.05%	0.090	0.098	3.60%	

Table 8.1. (Continued)

Time	Heat	Throttle	Actual	Pressure			Speed		
				Predicted	%Error	Actual	Predicted	%Error	
5:46	0.4	2.2	14.0	14.3	2.99%	0.085	0.094	3.97%	
5:47	2.4	2.2	14.0	14.3	2.91%	0.095	0.091	-1.83%	
5:48	3.8	2.2	14.5	14.3	-1.89%	0.095	0.088	-2.97%	
5:49	4.0	2.2	14.5	14.4	-1.10%	0.095	0.086	-3.82%	
5:50	4.0	2.2	15.0	14.5	-4.95%	0.090	0.085	-2.24%	
5:51	4.0	2.2	15.5	14.6	-8.86%	0.085	0.084	-0.43%	
5:52	4.0	2.2	15.5	14.7	-8.00%	0.090	0.084	-2.78%	
5:53	4.0	2.2	15.5	14.8	-7.46%	0.090	0.083	-2.84%	
5:54	4.0	2.2	15.5	14.8	-7.20%	0.090	0.083	-2.83%	
5:55	4.0	2.2	15.5	14.8	-7.19%	0.095	0.084	-4.97%	
5:56	4.0	2.2	15.5	14.8	-7.13%	0.095	0.084	-4.93%	
5:57	4.0	2.2	15.5	14.8	-6.87%	0.095	0.084	-4.90%	
5:58	4.0	2.2	15.5	14.8	-6.52%	0.095	0.084	-4.84%	
5:59	4.0	2.2	15.5	14.9	-6.19%	0.095	0.084	-4.78%	
6:00	4.0	2.2	16.0	14.9	-10.90%	0.100	0.084	-6.87%	
6:01	4.0	2.2	16.0	14.9	-10.69%	0.095	0.084	-4.60%	
6:02	4.0	2.2	15.5	14.9	-5.52%	0.100	0.085	-6.68%	
6:03	4.0	2.2	16.0	15.0	-10.38%	0.095	0.085	-4.42%	
6:04	4.0	2.2	16.0	15.0	-10.21%	0.095	0.085	-4.34%	
6:05	4.0	2.2	16.0	15.0	-10.03%	0.100	0.085	-6.43%	
6:06	4.0	2.2	16.0	15.0	-9.84%	0.100	0.085	-6.35%	
6:07	4.0	2.2	16.0	15.0	-9.65%	0.095	0.086	-4.10%	
6:08	4.0	2.2	15.5	15.1	-4.48%	0.100	0.086	-6.20%	
6:09	4.0	2.2	16.0	15.1	-9.32%	0.095	0.086	-3.94%	
6:10	4.0	2.2	16.0	15.1	-9.17%	0.100	0.086	-6.04%	
6:11	5.0	2.2	16.0	15.1	-9.02%	0.100	0.086	-5.97%	
6:12	5.0	2.2	16.0	15.1	-8.76%	0.100	0.086	-5.89%	
6:13	5.0	2.2	16.0	15.2	-8.36%	0.105	0.087	-7.98%	
6:14	5.0	2.2	16.0	15.2	-7.92%	0.100	0.087	-5.70%	
6:15	5.0	2.2	16.0	15.2	-7.57%	0.105	0.087	-7.75%	
6:16	5.0	2.2	16.0	15.3	-7.25%	0.105	0.087	-7.62%	
6:17	5.0	2.2	16.0	15.3	-7.06%	0.105	0.088	-7.49%	
6:18	5.0	2.2	15.5	15.3	-1.89%	0.105	0.088	-7.37%	
6:19	5.0	2.2	16.0	15.3	-6.82%	0.105	0.088	-7.25%	
6:20	5.0	2.2	16.0	15.3	-6.66%	0.100	0.089	-4.98%	
6:21	5.0	2.2	16.0	15.4	-6.48%	0.100	0.089	-4.88%	
6:22	5.0	2.2	15.5	15.4	-1.29%	0.100	0.089	-4.79%	
6:23	5.0	2.2	16.0	15.4	-6.11%	0.105	0.089	-6.87%	
6:24	5.0	2.2	16.0	15.4	-5.94%	0.105	0.089	-6.78%	
6:25	5.0	2.2	16.0	15.4	-5.80%	0.100	0.090	-4.52%	
6:26	5.0	2.2	16.0	15.4	-5.66%	0.100	0.090	-4.44%	

(continued)

Table 8.1. (Continued)

Time	Heat	Throttle	Actual	Pressure			Speed		
				Predicted	%Error	Actual	Predicted	Predicted	%Error
6:27	5.0	2.2	16.0	15.4	-5.52%	0.100	0.090	0.090	-4.36%
6:28	5.0	2.2	15.5	15.5	-0.38%	0.100	0.090	0.090	-4.28%
6:29	5.0	2.2	15.5	15.5	-0.24%	0.100	0.090	0.090	-4.20%
6:30	5.0	2.2	15.0	15.5	4.91%	0.100	0.091	0.091	-4.13%
6:31	4.0	2.2	15.0	15.5	5.05%	0.100	0.091	0.091	-4.05%
6:32	4.0	2.2	15.5	15.5	0.08%	0.100	0.091	0.091	-3.98%
6:33	4.0	2.2	15.5	15.5	-0.02%	0.095	0.091	0.091	-1.75%
6:34	4.0	2.2	15.5	15.5	-0.16%	0.100	0.091	0.091	-3.88%
6:35	4.0	2.2	15.5	15.5	-0.21%	0.095	0.091	0.091	-1.69%
6:36	4.0	2.2	15.5	15.5	-0.23%	0.100	0.091	0.091	-3.85%
6:37	4.0	2.2	15.5	15.5	-0.15%	0.095	0.091	0.091	-1.67%
6:38	4.0	2.2	15.0	15.5	4.97%	0.095	0.091	0.091	-1.66%
6:39	4.0	2.2	15.5	15.5	0.17%	0.100	0.091	0.091	-3.81%
6:40	4.0	2.2	15.0	15.5	5.30%	0.090	0.091	0.091	0.57%
6:41	5.0	2.2	15.0	15.5	5.38%	0.095	0.091	0.091	-1.56%
6:42	5.0	3.8	15.0	15.6	5.57%	0.095	0.098	0.098	1.36%
6:43	5.0	4.2	14.5	15.6	10.90%	0.115	0.115	0.115	0.10%
6:44	5.0	4.2	15.0	15.6	6.36%	0.145	0.127	0.127	-7.64%
6:45	5.0	4.2	15.0	15.7	6.78%	0.145	0.136	0.136	-3.72%
6:46	5.0	4.2	15.0	15.7	7.20%	0.150	0.144	0.144	-2.63%
6:47	5.0	4.2	15.0	15.8	7.51%	0.145	0.150	0.150	2.21%
6:48	5.0	4.2	15.0	15.8	7.79%	0.145	0.155	0.155	4.35%
6:49	5.0	4.2	15.0	15.8	7.98%	0.145	0.159	0.159	6.02%
6:50	5.0	4.2	15.0	15.8	8.17%	0.145	0.162	0.162	7.30%
6:51	5.0	4.2	15.0	15.8	8.36%	0.145	0.164	0.164	8.27%
6:52	5.0	4.2	14.5	15.9	13.55%	0.145	0.166	0.166	9.00%
6:53	2.8	4.2	14.5	15.9	13.73%	0.145	0.167	0.167	9.56%
6:54	1.9	4.2	14.5	15.9	13.69%	0.140	0.168	0.168	12.15%
6:55	1.0	4.2	14.5	15.8	13.27%	0.140	0.169	0.169	12.42%
6:56	1.0	4.2	14.5	15.8	12.55%	0.145	0.169	0.169	10.36%
6:57	1.0	4.2	14.5	15.7	11.86%	0.145	0.169	0.169	10.33%
6:58	1.0	4.2	14.5	15.6	11.27%	0.145	0.168	0.168	7.80%
6:59	1.0	4.2	14.5	15.6	11.01%	0.150	0.168	0.168	7.80%
7:00	1.0	4.2	14.5	15.6	10.93%	0.145	0.167	0.167	9.77%
7:01	1.0	4.2	14.5	15.6	11.17%	0.145	0.167	0.167	9.62%
7:02	1.0	4.2	14.0	15.6	16.33%	0.145	0.167	0.167	9.53%
7:03	1.0	4.2	14.0	15.6	16.43%	0.145	0.167	0.167	9.50%
7:04	1.0	4.2	14.0	15.6	16.42%	0.145	0.167	0.167	9.50%
7:05	1.0	4.2	14.0	15.6	16.40%	0.145	0.167	0.167	9.50%
7:06	1.0	4.2	14.0	15.6	16.42%	0.145	0.167	0.167	9.50%
7:07	1.0	4.2	14.0	15.6	16.49%	0.145	0.167	0.167	9.51%
7:08	1.0	4.2	14.0	15.7	16.59%	0.145	0.167	0.167	9.52%
7:09	1.0	4.2	14.0	15.7	16.72%	0.145	0.167	0.167	9.55%

case of pressure and 0.035–0.265 A in the case of speed. The averages over the 90 seconds of the absolute values of the percentage errors were 7.34% in the case of pressure and 5.52% in the case of speed. It is interesting that the speed part did somewhat better than the pressure part here because the speed part depends somewhat on the pressure part in the final model and also because the speed part seemed to be the most troublesome during the development process.

If an appropriate interface could be developed, it might be quite interesting to use a test vaguely reminiscent of the Turing test. A person could be allowed to experiment with the actual steam engine for some time until becoming quite familiar with its behavior. Then the same person could be asked to operate the steam engine through an interface that made it impossible to see directly whether the interface was connected to the actual steam engine or to its neural model. We could then ask the person to distinguish between the actual device and the model, based only on behavior; if it proved impossible for the person to distinguish between the two, then the model would be judged very good.

We did set up a program for operating the completed model in manual form (adjusting the heat control and throttle settings manually as we went along) and experimented with it. This is not the same as the test just described because we were fully conscious that we were operating the model, and we may have been biased in our judgments; however we were convinced that the model would probably pass the test if the interface were available.

CHAPTER 9

Fuzzy Control Designs with Auto-Tuning and Various Forms of Dynamic Compensation

We are now ready to discuss fully the complete design process for our fuzzy controllers. Chapter 9 presents examples and further details of several of the topics considered in Chapters 5 and 6. In addition to demonstrating the basic aspects of fuzzy control design, we also discuss in considerable detail one practical and easily comprehensible way of using a neural model and genetic auto-tuning to improve the designs.

Chapter 9 is critical with respect to another goal of this book—comparing the utility of various forms of dynamic compensation in the context of fuzzy control. To make such comparisons in terms of actual demonstrations rather than conceptual or theoretical justifications, we must present specific designs on which to base those comparisons. Having previously discussed the steam-engine test plant, we now consider the process of designing fuzzy P (proportional), fuzzy PD (proportional-plus-derivative), fuzzy PI (proportional-plus-integral), and fuzzy PID (proportional-plus-integral-plus-derivative) control designs for that plant.

The design process presented here can be viewed as typical of fuzzy control development in general, except we must emphasize designing the various control devices as objectively as possible, so that our comparisons in Chapter 10 can be meaningful. We recognized from the start that it is nearly impossible to compare design techniques in an absolutely conclusive way, and this is even truer for techniques emphasizing heuristic knowledge. However there are now several variations in fuzzy control design methods, so it is possible to lean toward the more objective choice at various points in the process, thereby avoiding arbitrary comparisons. Therefore we decided to emphasize auto-tuning methods.

In this chapter we continue to emphasize similarities among fuzzy control, conventional control theory, and AI rather than the differences. We present some specific examples here of the benefits of this perspective.

9.1. High-Level System Designs for Fuzzy P, PD, PI, and PID Steam-Engine Controllers

In any control design process, we must decide on at least a tentative structure for the overall control system before going into design details. Even in conventional control, this step is nearly always based on heuristic knowledge about both general control processes and the specific test plant. Structures based on negative feedback are generally most effective; beyond that the structures should be as simple as possible, though simplicity is less essential for fuzzy control than conventional control.

An effective way of presenting the high level designs is to consider the proportional, derivative, and integral elements in turn. It then becomes obvious how these are combined for PD, PI, and PID control. The proportional element is the foundation for everything else. Neither in conventional control nor fuzzy control do we attempt to implement feedback without it; furthermore if derivative or integral elements are also used, then these are usually appropriate extensions of the proportional element structure.

Thus, the basic structure of all control designs used here is illustrated in Fig. 9.1. The actual pressure and speed readings are fed to the control device where they are compared to setpoints (desired values for pressure and speed at that time), resulting in pressure-error and speed-error values. These values are then used to determine what adjustments to make to the heat-control and throttle settings. This is generally simple, but one complexity is that this control is assumed to be truly multivariate. The pressure-error value is likely to be more important than speed error in determining appropriate heat-control setting; the opposite is true when determining throttle setting, but for reasons explained in Section 7.2, it is assumed that both error values are relevant to both control settings. One advantage reason-

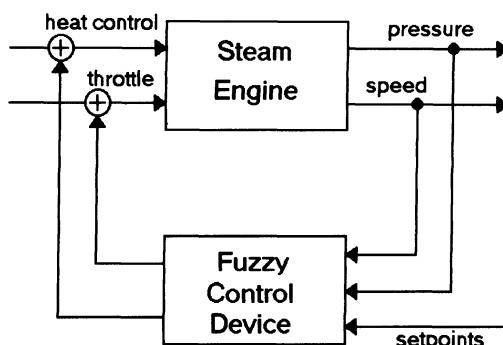


Figure 9.1. Basic structure of multivariate steam-engine-control design.

ably claimed for fuzzy control is that truly multivariate control is much easier to implement by fuzzy control methods than conventional methods. Note: One of the main reasons for choosing this particular test plant was to demonstrate the ease of multivariate fuzzy control design.

We must consider now the temporal aspects of the system. Fuzzy control nearly always treats time as a discrete variable; thus we should state the fixed time interval we intend to use. The steam engine model discussed in Chapter 8 was based on a 1-second time interval, so this is the most convenient interval to assume in our control designs. Although a somewhat shorter interval, such as 0.5 second, may be preferable, the 1-second interval was acceptable for approximating plant dynamics. It is therefore reasonable to assume that the same assessment applies to the cycle time controlling the plant.

Another consideration is that although we do not physically implement control here, it is still desirable for the sake of realistic demonstrations to consider the physical limitations that would exist if we did. For example if there were an interface board in our computer connected to a step-motor driver that in turn was connected to a step motor for adjusting the throttle setting on the steam engine, how long would it take to calculate the new throttle adjustment, then move the step motor by that amount? Although we did not conduct studies aimed at a precise answer to that question, it seems that an overall 1-second cycle time is about right from this perspective as well.

In describing the variables used in proportional control, we frequently abbreviate pressure error (P), speed error (S), heat change (HC), and throttle change (TC); this applies to both base variable and linguistic variable forms in each case. From the point of view of the fuzzy inference process within the fuzzy control device, P and S are the two input variables, and HC and TC are the two output variables.

The derivative element, or at least the conceptual equivalent of a derivative element, is a simple extension of the concepts in the proportional element. As previously stated, many fuzzy control designs use a derivative element, which could be called fuzzy PD control, but they are only rarely explicitly described in this way.

A derivative element in the context of fuzzy control design means if the current value of some variable (such as pressure) is useful in determining an appropriate control action (such as an adjustment in heat control setting), then it is probably also true that the most recent change in the value of that variable (such as the pressure change over the past second) is an additional piece of useful information. Chapter 2 describes at length a conceptual basis for the utility of a derivative element. Now if time is considered in terms of discrete, fixed intervals, then the most practical way of estimating the current derivative of a variable in real time is by calculating the difference between the current reading of that variable and the previous reading.

Once we understand the rationale for this derivative element, we must still consider what exactly we wish to take the derivative of. Is it the variable itself, or

is it the error function for that variable? The fuzzy control literature is often ambiguous on this point, and it seems likely that some fuzzy control applications use one interpretation of change, and other applications use the other interpretation; in many cases it is an unimportant distinction anyway. The distinction is significant only if there is an ongoing change in the setpoint, as for example when we consider the response to the type of ramp test function often used in conventional control design. Conventional control literature by contrast is very clear that the derivative to consider is error function's. If we reflect on the desired outcome in the face of changing setpoints, then this is the appropriate choice.

Therefore our derivative element adds two input variables to the fuzzy inference process—pressure error change (PC) and speed error change (SC). If the variables are treated as functions of time, with $t = 0$ indicating a present value and $t = -1$ indicating an immediately previous value, then base variable values are calculated as:

$$PC(0) = P(0) - P(-1)$$

$$SC(0) = S(0) - S(-1)$$

However the fuzzification of these variables is considered separately from the raw error values. Thus for fuzzy PD control of the steam engine, the fuzzy inference process has four inputs—P, S, PC, and SC—and two outputs—HC and TC.

An integral element, which historically has been much less common in fuzzy control, can be implemented in a way similarly appropriate for a discrete time system by taking the sum of the past several error values calculated. (Chapter 2 considers the conceptual justification for an integral term in detail.)

In the present research, our first assumption was to add two more input variables, pressure-error sum (PS) and speed-error sum (SS), to the design structure. However we realized that in this formulation of the control problem, PS is very useful, but SS is not: For our steam engine fuzzy controllers designed with 1-second cycle times, a variable that sums past speed errors has no value, and it does more harm than good if we try to use it in the design. We consider the general significance of this point in Chapter 11.

The only remaining difficulty is to decide how many time units to go back in calculating this sum. Even literature on conventional control theory tends to be somewhat vague on this point, and many texts use an indefinite integral type of notation when representing the integral term, though obviously it is not to be taken literally. The reason for this vagueness may be that traditionally in conventional control, the integral term is approximated by analog circuitry, and the integral limits cannot easily be stated precisely anyway.

Experience with the system suggested using 10 seconds to indicate recent tendencies in the pressure error, and experimentation demonstrated that this works well. It was also convenient to work with an average rather than a sum so that the

base value can be easily compared with the current error value. Thus the PS base value is calculated as:

$$PS(0) = 0.1 \sum_{t=-9}^0 P(t)$$

Figure 9.1 shows the overall structure for all forms of control considered here. However the pressure and speed data are processed differently in each design, so that from the perspective of the approximate reasoning processes within each fuzzy controller, the inputs and outputs are as follows:

Fuzzy P control:	Two inputs—P and S two outputs—HC and TC
Fuzzy PD control:	Four inputs—P, S, PC, and SC two outputs—HC and TC
Fuzzy PI control:	Three inputs—P, S, and PS two outputs—HC and TC
Fuzzy PID control:	Five inputs—P, S, PC, SC, and PS two outputs—HC and TC

9.2. Fuzzy Inference Algorithm Used

In principle decisions about inference algorithm implementations can be left to a late stage in the design process and to some degree this is also true in practice. However at least some tentative method of implementing the inference algorithm must be available before one can experiment with possible designs to analyze their effectiveness or to tune them. It may be most practical therefore in many circumstances to develop a complete system for implementing fuzzy control designs before proceeding very far with other aspects of a detailed design, for example when an auto-tuning method is to play a major role in the overall design process, which is the case here. For this reason we now explain some aspects of the inference algorithm used and its software implementation.

The work here is based on min-max fuzzy operators and the Mamdani interpretation of implication (or what is sometimes called implication) according to the alternative perspective presented in Section 5.4. Because this work will be simulated on a computer, we can describe this task the type of software implementation illustrated in Appendix B. For our purposes here we start with the software necessary for implementing the type of structure shown in Fig. 9.2.

We first make a few comments on the general nature of the software. There is nothing special about the actual programming for these examples, and there is no reason to go into detailed discussions of it in the body of this work. If anything the

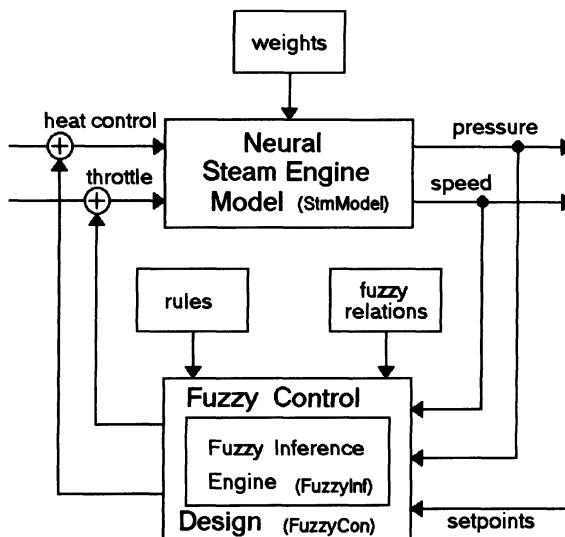


Figure 9.2. Schematic of the system model.

programming techniques used were somewhat behind the times, because we did not use object-oriented features of recent versions of Turbo Pascal, and we probably even failed to make as much use of dynamic variables as we should have. However we mention a few aspects of the highest levels of organization of the software developed. For software development efforts of even a moderate scale, it is convenient to be able to use separately compiled (external) collections of user-developed subprograms and data structure definitions. In the Pascal language, such collections are referred to as units, and some features of the language make it simple to develop and use these.

Referring to Fig. 9.2, as our system model, most of the software for implementing this model is contained in three units:

- **StmModel** : The software for the completed neural model of the steam engine described in Chapter 8 and illustrated specifically by Fig. 8.3
- **FuzzyInf** : Approximate reasoning routines and structures based on the alternative interpretation. Thus, this is more or less a generalized and expanded version Appendix B.
- **FuzzyCon** : All aspects of fuzzy P, PD, PI, and PID control of the steam engine other than the approximate reasoning inference engine. Thus, this contains routines and structures that pre- and postprocess the variables and invoke FuzzyInf routines when necessary

Using these units it is easy to develop various ways of experimenting with the system model. For example we can quickly set up a program for a manual version where one simply inputs the initial conditions and the second-by-second setpoints, then observe results under various designs. Alternatively we can set up a program for running various fixed test functions from a separate file, but this is already moving toward additional software, which is described later in this chapter.

We note here a few points about the design that may seem minor but are actually significant. When using the alternative interpretation of approximate reasoning by fuzzy inference, it is unnecessary to define explicitly the base value space of input variables, but it is still preferable to state a finite set of base values for output variables for the defuzzification step. It is convenient that in this application, the heat control setting and the throttle settings can be viewed in almost precisely the same way. In both cases these are adjustable in a continuous range from 0–10 arbitrary units, but it is unnecessary to consider adjustments finer than 0.1 unit. Also in both cases it is unnecessary to adjust the control more than half of its range in either direction in any 1 second. Therefore we decided to define the base variable spaces as –5.0–5.0 units at intervals of 0.1 for both the heat control and the throttle.

Also, as a minor part of the postprocessing in each control design, there is logic to ensure that the control device never attempts to adjust either of these beyond their limits. That is, the control devices must never attempt to move heat control or throttle below 0.0 or above 10.0.

9.3. Control Rules for Fuzzy P, PD, PI, and PID Steam-Engine Controllers

The first decision to make in developing a rule base for fuzzy control is selecting the sets of terms to be used with each of the linguistic variables. Term sets are not required to be the same for all variables even within the same application, and there are few precise guidelines for choosing various term sets. In the overall history of fuzzy control methods, there has been a tendency to vary the term set from one variable to the next according to the nature of the heuristic knowledge about the various variables, but the set that came closest to being a standard one was the seven-term set discussed in Chapters 4 and 5—{NB, NM, NS, ZO, PS, PM, PB}.

Glancing through more recent literature, we note two new trends—a tendency toward using the same term set for all variables in a given application (except possibly when the fuzzy singleton method is used) and toward smaller term sets, five being a fairly common size now. Some papers, such as (Hayashi *et al.*, 1993), indicate that it may be practical to use only three terms for each variable provided that the other aspects of the design are carefully considered.

We used a set of five terms for each linguistic variable—{NB, NS, ZO, PS, PB} (for negative_big, negative_small, near_zero, positive_small, and positive_big). The following factors contributed to this decision:

- We preferred to avoid using variable reduction because the choice of parameters fixes the influence of the dynamic compensation terms too inflexibly.
- Lacking an obvious reason for viewing some variables more coarsely or finely than others, we wished to keep the overall design structures as uniform as possible.
- Our interpretation of fuzzy PID control in this steam engine control application involves an approximate reasoning process with five input and two output variables. Regardless of the format used or the interpretation of fuzzy inference algorithms, this many variables implies that the rule base must be very large even with term sets of five terms. If seven terms were used for each linguistic variable, then the rule base would have an overwhelming size.
- Term sets of five terms are not too radical a departure from either the traditions of classical fuzzy control or the recent trends, but three terms seem too extreme.

We saw in Chapter 6 that a popular fuzzy control design strategy is to develop the rule base on the basis of heuristic knowledge, then to determine fuzzy relations for the various linguistic variables on the basis of an auto-tuning method; this is what we intend to do here. However we do not interpret the idea of heuristic-based rules so strictly that it is impossible to tune or modify the rule base later if its performance is unacceptable in some way. Therefore the rule bases used in this research and presented in this section are in some cases based wholly on our original heuristic concepts, but in other cases these are based on a combination of knowledge that existed before we began experimenting and knowledge that we learned from experimenting.

Table 9.1. Control Rules for Fuzzy P Control Design for the Steam Engine Example

(a) HC Algorithm						(b) TC Algorithm							
S	P	NB	NS	ZO	PS	PB	S	P	NB	NS	ZO	PS	PB
NB	PB	PB	PB	PB	PS	PS	NB	PB	PB	PS	ZO	NS	NS
NS	PB	PS	PS	PS	ZO		NS	PB	PS	ZO	NS	NB	
ZO	PS	ZO	ZO	ZO	NS		ZO	PB	PS	ZO	NS	NB	
PS	ZO	NS	NS	NS	NB		PS	PB	PS	ZO	NS	NB	
PB	NS	NB	NB	NB	NB		ZO	PS	ZO	NS	NB	NB	

Table 9.2. Control Rules for Fuzzy PD Control Design for the Steam Engine Example

(a) HC Algorithm

(continued)

Table 9.2. (Continued)

		(b) TC Algorithm																		
SC	S	NB			NS			ZO			PS			PB						
		NB	NB	NS	ZO	PS	PB	NB	NS	ZO	PS	PB	NB	NS	ZO	PS	PB			
PC	PB	PB	PB	PS	ZO	PB	PB	PS	PS	ZO	PB	PS	ZO	ZO	NS	PS	PS	ZO	NS	NS
	NS	PB	PB	PS	ZO	PB	PB	PS	ZO	NS	PS	PS	ZO	ZO	NS	PS	ZO	ZO	NS	NS
	NB	PB	PB	PS	ZO	PB	PB	PS	ZO	NS	PS	PS	ZO	NS	NS	PS	ZO	ZO	NS	NB
	NS	PB	PB	PS	ZO	PB	PB	PS	ZO	NS	PS	PS	ZO	ZO	NS	PS	ZO	ZO	NS	NB
	ZO	PB	PB	PS	ZO	PB	PB	PS	ZO	NS	PS	PS	ZO	ZO	NS	PS	ZO	ZO	NS	NB
	PS	PB	PB	PS	ZO	PB	PB	PS	ZO	NS	PS	PS	ZO	ZO	NS	PS	ZO	ZO	NS	NB
SC	PB	PB	PS	ZO	ZO	PB	PB	PS	ZO	ZO	PB	PS	ZO	ZO	NS	PS	ZO	ZO	NS	NS
	NS	PB	PB	PS	ZO	PB	PB	PS	ZO	ZO	PB	PS	ZO	ZO	NS	PS	ZO	ZO	NS	NB
	NB	PB	PB	PS	ZO	PB	PB	PS	ZO	ZO	PB	PS	ZO	ZO	NS	PS	ZO	ZO	NS	NB
	NS	PB	PB	PS	ZO	PB	PB	PS	ZO	ZO	PB	PS	ZO	ZO	NS	PS	ZO	ZO	NS	NB
	ZO	PB	PB	PS	ZO	PB	PB	PS	ZO	ZO	PB	PS	ZO	ZO	NS	PS	ZO	ZO	NS	NB
	PS	PB	PB	PS	ZO	PB	PB	PS	ZO	ZO	PB	PS	ZO	ZO	NS	PS	ZO	ZO	NS	NB
PC	PB	PB	PS	ZO	NS	PB	PB	PS	ZO	NS	PB	PB	ZO	ZO	NS	PS	ZO	ZO	NS	NB
	NS	PB	PB	PS	ZO	PB	PB	PS	ZO	NS	PB	PB	ZO	ZO	NS	PS	ZO	ZO	NS	NB
	NB	PB	PB	PS	ZO	PB	PB	PS	ZO	NS	PB	PB	ZO	ZO	NS	PS	ZO	ZO	NS	NB
	NS	PB	PB	PS	ZO	PB	PB	PS	ZO	NS	PB	PB	ZO	ZO	NS	PS	ZO	ZO	NS	NB
	ZO	PB	PB	PS	ZO	PB	PB	PS	ZO	NS	PB	PB	ZO	ZO	NS	PS	ZO	ZO	NS	NB
	PS	PB	PB	PS	ZO	PB	PB	PS	ZO	NS	PB	PB	ZO	ZO	NS	PS	ZO	ZO	NS	NB
SC	PB	PB	PS	ZO	NS	PB	PB	PS	ZO	NS	PB	PB	ZO	ZO	NS	PS	ZO	ZO	NS	NB
	NS	PB	PB	PS	ZO	PB	PB	PS	ZO	NS	PB	PB	ZO	ZO	NS	PS	ZO	ZO	NS	NB
	NB	PB	PB	PS	ZO	PB	PB	PS	ZO	NS	PB	PB	ZO	ZO	NS	PS	ZO	ZO	NS	NB
	NS	PB	PB	PS	ZO	PB	PB	PS	ZO	NS	PB	PB	ZO	ZO	NS	PS	ZO	ZO	NS	NB
	ZO	PB	PB	PS	ZO	PB	PB	PS	ZO	NS	PB	PB	ZO	ZO	NS	PS	ZO	ZO	NS	NB
	PS	PB	PB	PS	ZO	PB	PB	PS	ZO	NS	PB	PB	ZO	ZO	NS	PS	ZO	ZO	NS	NB
SC	PB	PB	PS	ZO	NS	PB	PB	PS	ZO	NS	PB	PB	ZO	ZO	NS	PS	ZO	ZO	NS	NB
	NS	PB	PB	PS	ZO	PB	PB	PS	ZO	NS	PB	PB	ZO	ZO	NS	PS	ZO	ZO	NS	NB
	NB	PB	PB	PS	ZO	PB	PB	PS	ZO	NS	PB	PB	ZO	ZO	NS	PS	ZO	ZO	NS	NB
	NS	PB	PB	PS	ZO	PB	PB	PS	ZO	NS	PB	PB	ZO	ZO	NS	PS	ZO	ZO	NS	NB
	ZO	PB	PB	PS	ZO	PB	PB	PS	ZO	NS	PB	PB	ZO	ZO	NS	PS	ZO	ZO	NS	NB
	PS	PB	PB	PS	ZO	PB	PB	PS	ZO	NS	PB	PB	ZO	ZO	NS	PS	ZO	ZO	NS	NB
SC	PB	PB	PS	ZO	NS	PB	PB	PS	ZO	NS	PB	PB	ZO	ZO	NS	PS	ZO	ZO	NS	NB
	NS	PB	PB	PS	ZO	PB	PB	PS	ZO	NS	PB	PB	ZO	ZO	NS	PS	ZO	ZO	NS	NB
	NB	PB	PB	PS	ZO	PB	PB	PS	ZO	NS	PB	PB	ZO	ZO	NS	PS	ZO	ZO	NS	NB
	NS	PB	PB	PS	ZO	PB	PB	PS	ZO	NS	PB	PB	ZO	ZO	NS	PS	ZO	ZO	NS	NB
	ZO	PB	PB	PS	ZO	PB	PB	PS	ZO	NS	PB	PB	ZO	ZO	NS	PS	ZO	ZO	NS	NB
	PS	PB	PB	PS	ZO	PB	PB	PS	ZO	NS	PB	PB	ZO	ZO	NS	PS	ZO	ZO	NS	NB

For fuzzy P control even in this case of multivariate control, the rule base is quite straightforward, especially because we base each linguistic variable definition on only five terms. We know that speed error should influence heat control adjustment, but not nearly so much influence as pressure error. We also know that the opposite is true for throttle adjustments. The rule base shown in tabular form in Table 9.1 is one expression of this knowledge, and in fact there are not many other ways it could be expressed.

A rule base for fuzzy PD control is slightly more challenging, but it is reasonable to assume that the rule base chosen for fuzzy PD control should have as its projection where PC = ZO and SC = ZO, the fuzzy P control rule base. Beyond that it is not completely obvious what the relative influences of the error-change

Table 9.3. Control Rules for Fuzzy PI Control Design for the Steam Engine Example

		(a) HC Algorithm						(b) TC Algorithm					
PS	S	NB	NS	ZO	PS	PB	PS	NB	NS	ZO	PS	PB	
NB	P						NB	P					
	NB	PB	PB	PB	PB	PB		NB	PB	PB	PS	ZO	NS
	NS	PB	PB	PB	PS	PS		NS	PB	PS	ZO	NS	NB
	ZO	PS	PS	PS	PS	ZO		ZO	PB	PS	ZO	NS	NB
	PS	PS	ZO	ZO	ZO	ZO		PS	PB	PS	ZO	NS	NB
NS	PB	ZO	ZO	ZO	NS	NS		PB	PS	ZO	NS	NB	NB
	NB	PB	PB	PB	PB	PS		NB	PB	PB	PS	ZO	NS
	NS	PB	PS	PS	PS	PS		NS	PB	PS	ZO	NS	NB
	ZO	PS	PS	ZO	ZO	ZO		ZO	PB	PS	ZO	NS	NB
	PS	ZO	ZO	ZO	ZO	NS		PS	PB	PS	ZO	NS	NB
ZO	PB	ZO	NS	NS	NS	NS		PB	PS	ZO	NS	NB	NB
	NB	PB	PB	PS	PS	PS		NB	PB	PB	PS	ZO	NS
	NS	PS	PS	PS	ZO	ZO		NS	PB	PS	ZO	NS	NB
	ZO	ZO	ZO	ZO	ZO	ZO		ZO	PB	PS	ZO	NS	NB
	PS	ZO	ZO	NS	NS	NS		PS	PB	PS	ZO	NS	NB
PS	PB	NS	NS	NS	NB	NB		PB	PS	ZO	NS	NB	NB
	NB	PS	PS	PS	PS	ZO		NB	PB	PB	PS	ZO	NS
	NS	PS	ZO	ZO	ZO	ZO		NS	PB	PS	ZO	NS	NB
	ZO	ZO	ZO	ZO	NS	NS		ZO	PB	PS	ZO	NS	NB
	PS	NS	NS	NS	NS	NB		PS	PB	PS	ZO	NS	NB
PB	PB	NB	NB	NB	NB	NB		PB	PS	ZO	NS	NB	NB
	NB	PS	PS	ZO	ZO	ZO		NB	PB	PB	PS	ZO	NS
	NS	ZO	ZO	ZO	ZO	NS		NS	PB	PS	ZO	NS	NB
	ZO	ZO	NS	NS	NS	NS		ZO	PB	PS	ZO	NS	NB
	PS	NS	NS	NB	NB	NB		PS	PB	PS	ZO	NS	NB

Table 9.4. Control Rules for Fuzzy PID Control Design for the Steam Engine Example (HC Algorithm)^a

(continued)

Table 9.4. (*Continued*)

(continued)

Table 9.4. (Continued)

		SC		NB		NS		ZO		PB		PS		PB		
PS	PC	S	NB	NS	ZO	PS	PB	NB	NS	ZO	PS	NB	NS	ZO	PS	PB
NB	PB	PS	PB	PB	PB	PS	PB	PS	PB	PS						
	PS	PB	PB	PB	PB	PB	PB	PS	PB	PB	PB	PS	PB	PS	PB	PS
	PS	PS	PB	PB	PB	PB	PB	PS	PB	PB	PB	PS	PB	PS	PB	PS
	ZO	PS	PS	PS	ZO	PS	PS	ZO	ZO	PS	ZO	ZO	ZO	ZO	ZO	ZO
	PS	ZO														
	ZO															
	ZO															
	ZO															
	ZO															
	ZO															
NS	PS															
	PS	ZO														
	ZO															
	ZO															
	ZO															
	ZO															
	ZO															
	ZO															
	ZO															
	ZO															
PS	PB	PS														
	PS	ZO														
	ZO															
	ZO															
	ZO															
	ZO															
	ZO															
	ZO															
	ZO															
	ZO															
PB	PS															
	PS	ZO														
	ZO															
	ZO															
	ZO															
	ZO															
	ZO															
	ZO															
	ZO															
	ZO															

^aTC algorithm for all values of PS is the same as the TC algorithm in Table 9.2.

values should be. We decided to make PC slightly less important than P, but more important than S or SC in determining HC. Similarly for throttle control, we made SC slightly less influential than S but more important than P or PC. Because it is important to emphasize stability in the case of the speed control, we experimented at one point with making SC more influential than S but found that it did not work quite so well; therefore we returned to the original design. The complete rule base for the fuzzy PD steam engine controller is shown in Table 9.2.

For the PI and PID control, we found that the integral term is best when its influence is not too strong. We therefore made PS only moderately influential in the rules related to HC and totally without influence in the rules related to TC. This means that the TC rules for PI control are the cylindrical extension of the rules for P control, and the same is true of the relation between the PID and the PD rules. Table 9.3 shows the rule base for PI control; and Table 9.4 shows the one for PID control.

9.4. Linguistic Variable Definition and Auto-Tuning of Fuzzy Steam Engine Controllers by Genetic Methods

Auto-tuning of fuzzy controllers is now a quite popular approach, and though it is certainly possible to interpret this as an automatic generation of fuzzy control rules, the most common interpretation is as a technique for identifying the various (input and output) linguistic variable relations. One way of viewing auto-tuning is by first formulating linguistic variable identification as an optimization problem, then solving that problem. In this section we explain the process from this perspective in detail. Note: Near optimal solutions are generally considered sufficient, and this is probably sensible because it is difficult to formulate the problem in an absolutely precise manner anyway.

To formulate the optimization problem, the first task is how to state an objective function. The point is that we must find some way of expressing a reasonable evaluation function as illustrated simply by Fig. 9.3—a function that provides an index to indicate for example which of two possible collections of linguistic variable relations is the better. Note: This assumes that the rule base and all other aspects of the design other than the linguistic variable relations are fixed. There are very many ways one could approach this question, and there is not yet any general consensus as to which approach is most reasonable. As noted in Section 6.7, we feel that the thinking of (Yamamoto *et al.*, 1993) is appealing for its clarity, but we also believe that their approach should be further refined.

Let us begin by assuming it is possible to state either a single test function or a small set of test functions that typify circumstances to which we wish our control design to respond. We use *test function* to mean a data structure containing both a set of initial conditions and a collection of setpoints defined as functions of time

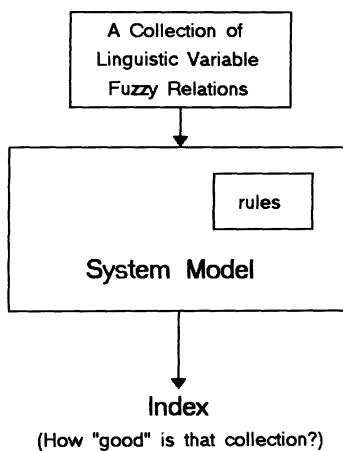


Figure 9.3. Basic concept of control-design-evaluation function as applied to linguistic variable relations.

over some interval; for example see our single test function used for auto-tuning in this application:

```

Starting at 7 minutes and 30 seconds after
ignition:
Assume Heat Control setting = 5.0
Pressure = 16.50
Throttle setting = 7.0
Speed = 0.212

```

and that these same conditions had been maintained for the previous 8 seconds as well.

Leave the setpoints at the current values (16.5 and 0.212) for 5 more seconds to stabilize. Do not yet begin to measure performance during this time.

At 7 minutes 35 seconds change the setpoints to 16.30 for pressure and 0.150 for speed. Maintain those new setpoints and measure performance of the control device for the next 45 seconds.

This is obviously a step-type function on both pressure and speed. Whether it perfectly typifies the circumstances to which we would like the control devices to be able to react is a rather subjective matter, but let us assume for now that this is a reasonable test function.

Even after deciding on such a test function or collection of test functions, we still do not have our evaluation or objective function. It is insufficient to say that we want the control device to be able to respond well to this situation; we must define an index for just how well a given design is responding. This is a good example of why it is useful to have an open mind about the conventional control theory literature.

Dorf (1974), defines four types of performance indices and discusses their relative significance and merits in detail. Given a test function and a control design, let $e(t)$ be the error function with $t = 0$ representing the time we wish to begin measuring performance and $t = k$ indicating the end of the performance-measurement period. Four indices are then defined as follows:

$$\text{IAE} = \int_0^k |e(t)| dt$$

$$\text{ISE} = \int_0^k e^2(t) dt$$

$$\text{ITAE} = \int_0^k t|e(t)| dt$$

$$\text{ITSE} = \int_0^k te^2(t) dt$$

This too is a somewhat subjective matter, but Dorf seems to indicate that integral of time times absolute error (ITAE) is superior to the others. It is generally preferred because it is the most selective index, tending to have the most distinct minimum as a given parameter is varied in a typical conventional control design.

Selectivity need not be the only criterion for judging an index. We may also wish to consider how meaningful the index is for our purposes. One very appealing point about ITAE is that, with an appropriate test function and limits of integration, it can very elegantly combine measurements of responsiveness, stability, and accuracy in a reasonably balanced way. By saying that a moderate error occurring long after the setpoint was changed is to be penalized at least as much as a large error occurring immediately after the change, we have a good method for emphasizing relative stability and accuracy as well as responsiveness. An IAE or ISE index clearly places too much emphasis on responsiveness.

If fuzzy control designs are similar to conventional control designs for selectivity (a question we have not considered carefully), then ITAE is most likely to provide clear minima, but regardless of whether that is true, whatever minima it

provides will be most meaningful. Referring to Section 6.6, Yamamoto *et al.* used an IAE-type index; we however use an ITAE-type index.

One further complication is that we are concerned with a multivariate control problem. We can calculate an ITAE index for pressure and one for speed, but that leaves us with two indices rather than the one we desire. Due to differences in scales and so on, the ITAE value calculated for pressure tends to be about ten times that for speed in a typical tentative design; we therefore decided to deal with this problem simply by calculating the following ITAE score:

$$\text{Overall ITAE Score} = \text{ITAE}_{\text{Pressure}} + 10 \text{ ITAE}_{\text{Speed}}$$

We have now dealt with the first question, and we are on our way to developing the type of structure we saw in Fig. 9.3.

The second question in formulating the fuzzy control design as an optimization problem is how to state the solution space in a reasonably concise way. We assume here that all that remains is to design the fuzzy relations for the linguistic variables; even so it is not immediately obvious how to express the problem. We must express several fuzzy relations, one for each input and output variable of the fuzzy inference process. In our case this is four, six, five, and seven fuzzy relations for fuzzy P, PD, PI, and PID control, respectively.

The major decision to be made here is how many assumptions to make about each fuzzy relation. If we begin by stating that each term must correspond to a TFN, then we are already well on our way. If each term is expressed in (a_1, a_2, a_3) form and five terms are assumed for each fuzzy relation, then this leaves us with the number of linguistic variables times 15 real parameters to state in order to express a given solution. However a few more assumptions may be reasonable as well, and these will reduce the dimension of the solution space much further. The following are reasonable for each fuzzy relation R , and this too is quite similar to the approach used by Yamamoto *et al.*, though they did not express their ideas in exactly this way.

- $a_{1,\text{NB}} = -\infty$ and $a_{3,\text{PB}} = \infty$
- Using the same notation as in previous chapters, for term set T :

$$\sum_{t \in T} R(t, x) = 1 \quad \text{for all } x \in \mathcal{R}$$

and furthermore

$$|\{t \in T \mid R(t, x) > 0\}| \leq 2 \quad \text{for all } x \in \mathcal{R}$$

These two assumptions imply the following:

$$a_{2,\text{NB}} = a_{1,\text{NS}} \quad a_{3,\text{NB}} = a_{2,\text{NS}} = a_{1,\text{ZO}}$$

$$a_{3,\text{NS}} = a_{2,\text{ZO}} = a_{1,\text{PS}} \quad a_{3,\text{ZO}} = a_{2,\text{PS}} = a_{1,\text{PB}}$$

$$\alpha_{3,PS} = \alpha_{2,PB}$$

- $\alpha_{2,ZO} = 0$. Some researchers do not make this assumption, but we feel it is necessary if the concept of near zero is to have a reasonable meaning.
- The TFNs need not be symmetric, but the overall fuzzy relation is symmetric around the y-axis, by this we mean

$$\alpha_{2,NB} = -\alpha_{2,PB} \quad \text{and} \quad \alpha_{2,NS} = -\alpha_{2,PS}$$

With these assumptions each fuzzy relation can be expressed by just two real values d_1 and d_2 , as shown in Fig. 9.4.

This leaves us with the number of input and output variables times 2 parameters to identify in each case. This means only 8, 12, 10, and 14 values for the fuzzy P, PD, PI, and PID steam-engine controllers.

We have now formulated the optimization problem clearly as minimizing the ITAE score in Fig. 9.5; thus Fig. 9.5 illustrates how to implement the idea of Fig. 9.3. Most of the additional software for accomplishing this is contained in the unit EvalRun, which draws the test-function definition(s) from a small file of a particular format; it works in combination with the system model, as expressed in the units StmModel, FuzzyInf, and FuzzyCon.

Now that the optimization problem has been clearly formulated, how should it be solved? Over the past few years there has been much interest in genetic algorithms for this auto-tuning task in fuzzy control. We also came into the design process already planning to use a genetic approach to auto-tuning.

However, a very reasonable question to ask is whether a genetic approach is actually the appropriate optimization method to use for solving this particular type of problem. Is GA really better than conventional optimization techniques for the problem as just formulated, or is the interest in genetic based auto-tuning of fuzzy control designs only part of a more general recent fascination with GA?

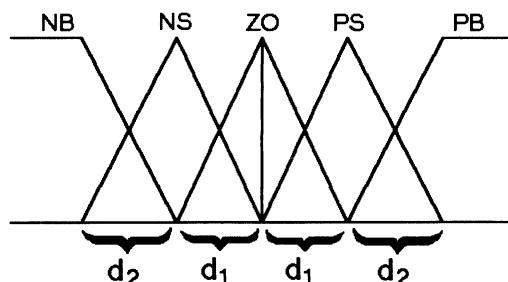


Figure 9.4. The expression of a 5-term linguistic variable relation by two parameters under given assumptions.

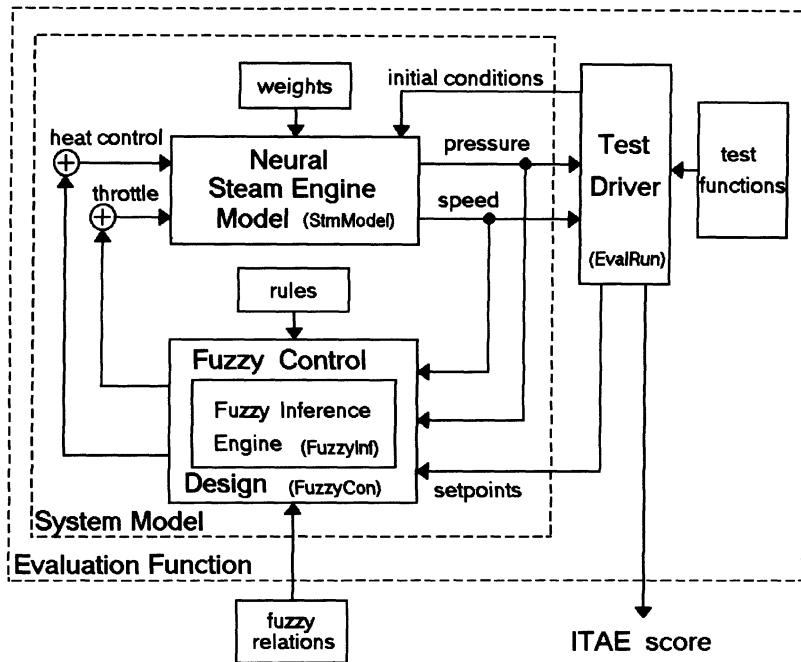


Figure 9.5. Implementation scheme for a control-design-evaluation function.

To answer this question fully would require a very detailed study both of the nature of the problem and of potentially effective conventional optimization techniques. We devoted only a few days to an informal and incomplete investigation; our tentative hypotheses were the following:

- All of the well known, relatively straightforward, generally applicable optimization techniques are either inapplicable to the optimization problem as we formulated it; or are applicable with performance levels that are much worse than GA.
- However, it may be possible with considerable effort to develop a specialized, probably highly technical, optimization technique that does work more efficiently than GA. Such a technique could possibly be an interesting research topic.

To summarize briefly the difficulties, we are dealing with a nondifferentiable function in several variables and rife with weak local minima. We conclude that for now it is reasonable to use a genetic approach.

There are challenges in applying a GA approach to this problem. These involve some difficult compromises in deciding exactly how to apply GA most effectively

to this difficult problem, as well as the general limitations of the genetic approach itself. To some degree these problems can be dealt with in a computationally intensive manner, by buying time on a supercomputer for example. However even in that case, it is still necessary to consider somewhat carefully the manner in which genetic techniques should be applied to this auto-tuning problem. In some cases as in this research, we can reasonably formulate the problem so that it is still possible to use a desktop computer.

The first difficulty is that each evaluation requires very considerable calculation time. The system model (plant and controller) must run through the chosen test function(s), and the error functions must be taken and integrated into an index each time the evaluation function must be calculated. In a genetic approach, this generally means for each individual organism in the population at each generation. Thirty organisms is generally taken as a minimum population because most genetic approaches seem to be much less effective below that number, and the number of generations assumed necessary can vary from several dozen to several thousand.

This is not an insurmountable problem in itself, but it leads to a difficult compromise. Ideally the evaluation function should make possible a rather comprehensive evaluation of the design. This means that it would be preferable to base it not on just one or a few, but on several test functions, and each of those test functions should be allowed to run for long periods of simulated (virtual) time.

For example in this application, the evaluation of each potential fuzzy controller design should be based on perhaps a dozen or more test functions; some where the pressure and speed setpoints are changed separately and others where they are changed simultaneously; some where the test function is of a step type, some where it is an impulse type, and others where it is of a ramp type; some where the setpoints are moved up, some where they are moved down, and some where the pressure and speed setpoints are moved in opposite directions; some where the changes are small and some where they are large; and so on. Each of these test functions should be allowed to run for several simulated minutes. We could easily set up a test function file according to our format to implement such a collection of test functions, but then each call to the evaluation function would have taken perhaps 100 times longer than it did. The only solution is to choose just one or a few test functions as carefully as possible based on heuristic knowledge about the system and its desired behavior, then to hope that we have chosen well.

The second compromise concerns discretization. Fortunately this is not so serious a problem as the first, but it still requires some degree of care. Genetic methods assume that the solution space of the problem is large but finite. For some problems, including our auto-tuning optimization problem, this means the solution space must be discretized in some way. More specifically for each variable (in this case the d_1 and d_2 parameters for each of linguistic inputs and outputs) lower and upper limits must be assumed, and the space between must be divided into some finite number of points. Obviously, this means that there is no guarantee of finding

the true optimum of the real valued problem; in fact for the typical real valued function, it is almost certain that we will not. Therefore we must assume at this point that near optimal solutions are acceptable.

In addition there are two obvious potential errors we may make in choosing the discretization scheme: We may err in our choice of upper or lower limits for one or more of the many variables, and we may not have left enough points in between so that we can even come close to the actual optimal point. In some sense, the first of these potential errors may not be a problem because we choose the upper and lower limits of the various d_1 and d_2 variables by ranges that are meaningful in defining the respective linguistic variable relation; therefore these constraints are not arbitrary anyway. Furthermore if we did find after experimenting that there is a potential benefit in changing the range, we could reconsider the scheme at that point.

The second potential error in the choice of a discretization scheme is slightly more serious, but our informal investigation of the nature of the optimization problem suggests that it is not a critical problem either. Though we found that the evaluation function was nondifferentiable and highly irregular overall, this irregularity did not seem to be such that even a moderately coarse discretization scheme would fail to find a near optimal solution. Figure 9.6 indicates the typical nature of the function when viewed in one dimension. This is taken from the actual fuzzy P control design, allowing only the d_1 parameter for S to vary from 0.0200–0.0300. Note: This plot of 1001 points fails to find a solution better than the one found by GA even though the GA discretization used here touched on only six points in this range on this dimension. Because this is only one example (though it is typical of

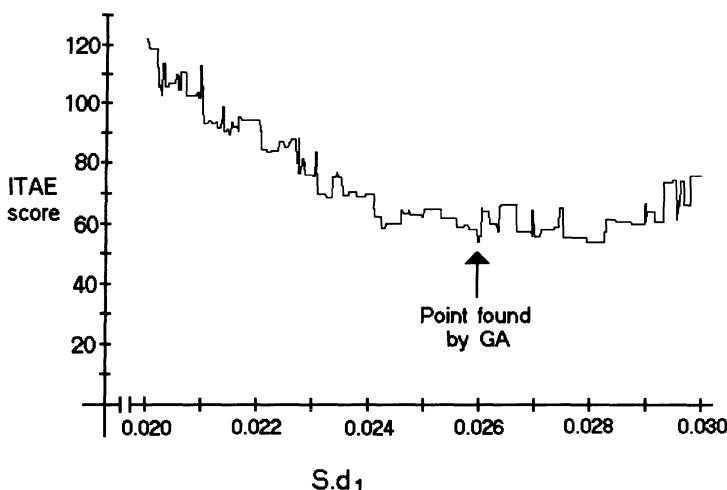


Figure 9.6. Change in ITAE score as a function of d_1 parameter for speed error in fuzzy P design.

several other examples we examined), it considers only a one-dimensional projection of the evaluation function, and the entire investigation was of an informal nature this is by no means conclusive. However it indicates why we can be fairly confident that it is appropriate to seek near optimal solutions to our formulation of the problem by using genetic methods and even moderately fine discretization schemes may be acceptable. Incidentally this function is not nearly so irregular as the one illustrated by Yamamoto *et al.* Whether this difference is due to the use of ITAE rather than IAE indices is a matter that we did not investigate but could be an interesting topic for future consideration. Another point is that the prevalence of weak minima should be viewed as an advantage rather than a disadvantage because it means that there is a much higher probability of finding a solution in the discretized solution space that actually does attain the global minimum of the real valued problem.

Concerning the general limitations of genetic algorithms, the most troubling is the lack of a clear termination criterion. Most conventional optimization methods give fairly obvious clues that they are converging on an optimal solution, so it is quite easy to decide when to terminate the calculations; this is not so true with genetic methods. That is not so much a problem in a design method whose goal is to find a better solution, but it does not provide a provably best solution, which still leaves us with a less conclusive basis for comparison than we would like.

Next we consider the matter of coding the solution space as a chromosome structure. It is very much a custom in genetic methods to emphasize the use of bit-string-type chromosomes wherever possible. Even after reading various explanations, such as Goldberg's (1989), of the rationale for this emphasis, we are still unconvinced that this is necessary; however, we bow to this convention. Based on manual experiments with the software developed to this point we chose upper and lower limits for each of the d_1 and d_2 variables for each of the fuzzy relations in each of the control designs. These ranges were then divided into 15 intervals, meaning 16 points inclusively. Although this may seem a rather coarse discretization, our experiments indicated that it did not create a problem; therefore each d_1 or d_2 parameter is indicated by four bits or 1 hex digit.

Some of the literature suggests that the chromosome coding should keep related information as close together as possible in the chromosome structure. The most obvious implication of this principle here is that each d_1 parameter should be adjacent to its corresponding d_2 parameter, though we probably would have done that anyway. We also decided to shift the codings around so that the information about the most closely related linguistic variables is as close together as possible; therefore the following orders were used

For fuzzy P control: P · HC · S · TC

For fuzzy PD control: P · HC · PC · S · TC · SC

For fuzzy PI control: P · HC · PS · S · TC

For fuzzy PID control: P · HC · PC · PS · S · TC · SC

Another software unit, called GAdecode, was constructed just for the data structures and routines required to convert chromosomes into collections of fuzzy relations.

Although we find it interesting to experiment with more complex approaches to genetic algorithms, as we discussed in (Lewis and Lewis, 1994), it seemed desirable in this case to choose a specific method that is both fairly simple and as efficient as possible in producing good results within several dozen generations. Such a method is described in Chapter 3 of (Goldberg, 1989) as simple genetic algorithm (SGA). Goldberg even presents examples of Pascal routines for implementing various aspects of the approach. Perhaps the best thing to look at in this sort of presentation is not the coding style, but the types of data structures used and the elegance of the implementations in the algorithms. Thus even if one desires to recode everything according to one's own style, it is still quite helpful to see this type of presentation.

Goldberg uses separate structures to contain an old population (the population as of the previous generation) and a new population (the one we are attempting to create at this generation). Pairs of parents are chosen from the old population by the "roulette wheel" method on fitness measure. Each pairing always produces two offspring in the new population. In many cases this is by single crossover and mutation, but in other cases neither crossover nor mutation occurs at all, which has the effect of cloning or allowing the two parents to survive into the next generation.

We made only a few modifications to SGA for our purposes. First since each evaluation takes so much calculation time, we installed mechanisms for saving and reusing the ITAE scores of organisms that do not change from one generation to the next—that is chromosomes where no crossover nor mutation occurred.

The second minor modification involved using the simple "racheting" mechanism (a method for ensuring that the best-of-generation never gets worse from one generation to the next) of allowing the best-of-generation always to survive (or be cloned) into the next generation. Often referred to as elitism, admittedly this racheting mechanism is artificial but we felt that it was beneficial, so we chose the simple way of doing it. In fact because Goldberg's SGA always works with even-numbered population sizes, we always allow the two best organisms in a generation to survive.

The third modification is a dynamic rescaling technique, based on a suggestion Goldberg makes later in the same chapter. Before proceeding we explain how the raw fitness scores are calculated. When selection is based on the roulette wheel method, it is most convenient to view the problem as a maximization problem rather than a minimization problem. Therefore we calculate a raw fitness value as:

$$\text{Raw fitness} = \frac{20}{\text{ITAE score}}$$

Then apply the dynamic rescaling technique to these values to produce the fitness values that we finally use. There is no special significance in the choice of 20. It simply places the fitness scores in a range that is comfortable to work with.

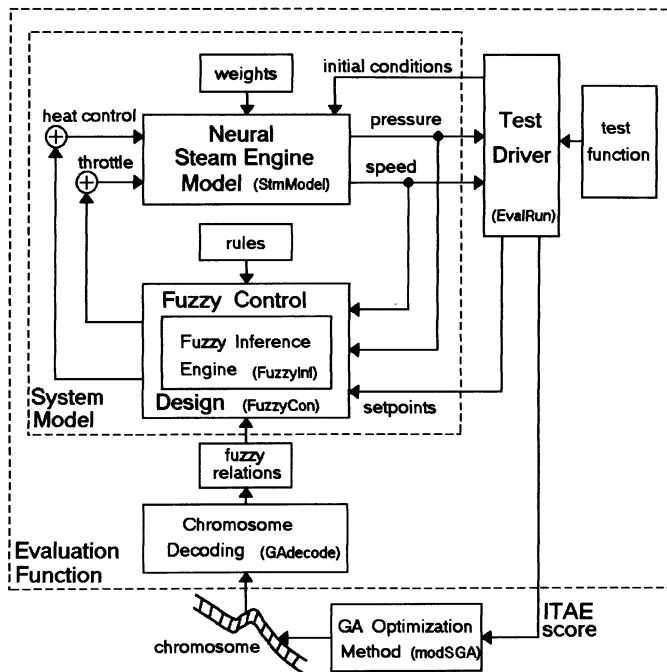


Figure 9.7. Complete genetic based fuzzy control auto-tuning scheme.

Dynamic rescaling applies a linear transformation to the fitness values based on a certain algorithm and on the average, maximum, and minimum raw fitness values for the population. This algorithm is designed in such a way as to optimize the effectiveness of the roulette wheel selection both in the early generations and at later stages in the evolution.

The data-structure definitions and routines necessary for implementing this slightly modified version of SGA were collected into another software unit called modSGA. Therefore, the complete auto-tuning method is as illustrated in Fig. 9.7.

Before concluding this section, we consider the auto-tuning process in a slightly different way. In conventional control tuning usually implies adjusting the "gains" on the proportional, derivative, and integral terms so that the control device performs to some specification. Why then is selecting the collection of linguistic variable relations referred to as tuning in the case of fuzzy control?

There is actually more of a connection between the two definitions of tuning than is at first apparent. Obviously, the control rules play a major role in determining how much influence a given input variable has over a given output variable. It is slightly less obvious, but the linguistic variable relations also play such a role.

Consider for example an input linguistic variable. If the d_1 and d_2 values are pushed out to their maximum allowed values, this has the effect of weakening the influence of that input because it becomes more difficult to attain high membership grades for the more extreme terms during the fuzzification process.

In fact, we noticed in the process of developing these designs that even though our interpretation of auto-tuning centers on the fuzzy relations, this is still helpful in tuning the control rules as well. For example if the auto-tuning system persists in trying to push out the d_1 and d_2 values for a given input linguistic variable as previously described, then this suggests that the rules are placing too much emphasis on that input, and perhaps they should be adjusted.

9.5. Final Designs Resulting from Auto-Tuning

We can now summarize the various completed and tuned fuzzy control designs as follows.

9.5.1. Fuzzy P Control

Base input variables :

$$P = \text{actual pressure reading} - \text{pressure setpoint}$$

$$S = \text{actual speed reading} - \text{speed setpoint}$$

Inference method: By alternative interpretation of approximate reasoning with fuzzy sets assuming max-min operators and other standard assumptions.

Linguistic input variable relations (in TFN a_1, a_2, a_3 form, see Fig. 9.8):

P:	NB: $a_1 = -\infty$	$a_2 = -3.0$	$a_3 = -1.2$
	NS: $a_1 = -3.0$	$a_2 = -1.2$	$a_3 = 0.0$
	ZO: $a_1 = -1.2$	$a_2 = 0.0$	$a_3 = 1.2$
	PS: $a_1 = 0.0$	$a_2 = 1.2$	$a_3 = 3.0$
	PB: $a_1 = 1.2$	$a_2 = 3.0$	$a_3 = \infty$
S:	NB: $a_1 = -\infty$	$a_2 = -0.062$	$a_3 = -0.026$
	NS: $a_1 = -0.062$	$a_2 = -0.026$	$a_3 = 0.000$
	ZO: $a_1 = -0.026$	$a_2 = 0.000$	$a_3 = 0.026$
	PS: $a_1 = 0.000$	$a_2 = 0.026$	$a_3 = 0.062$
	PB: $a_1 = 0.026$	$a_2 = 0.062$	$a_3 = \infty$

Rules: As found in Table 9.1.

Linguistic output variable relations (see Fig. 9.8):

HC:	NB: $a_1 = -\infty$	$a_2 = -1.8$	$a_3 = -0.2$
	NS: $a_1 = -1.8$	$a_2 = -0.2$	$a_3 = 0.0$

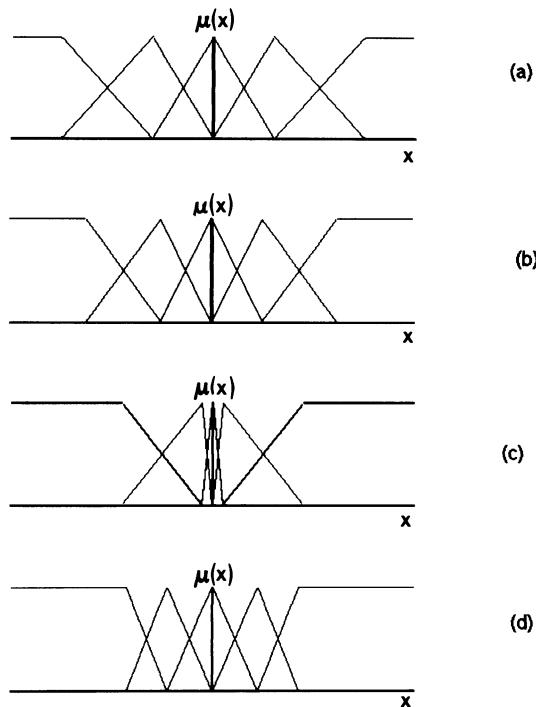


Figure 9.8. Fuzzy relations for linguistic variables in fuzzy P control design, (a) P (pressure error), (b) S (speed error), (c) HC (heat change), and (d) TC (throttle change).

ZO:	$a_1 = -0.2$	$a_2 = 0.0$	$a_3 = 0.2$
PS:	$a_1 = 0.0$	$a_2 = 0.2$	$a_3 = 1.8$
PB:	$a_1 = 0.2$	$a_2 = 1.8$	$a_3 = \infty$
TC:	NB: $a_1 = -\infty$	$a_2 = -1.7$	$a_3 = -0.9$
	NS: $a_1 = -1.7$	$a_2 = -0.9$	$a_3 = 0.0$
	ZO: $a_1 = -0.9$	$a_2 = 0.0$	$a_3 = 0.9$
	PS: $a_1 = 0.0$	$a_2 = 0.9$	$a_3 = 1.7$
	PB: $a_1 = 0.9$	$a_2 = 1.7$	$a_3 = \infty$

Defuzzification method: By center-of-gravity method.

Base output variable definitions for use in defuzzification:

$$\text{HC: } \{-5.0, -4.9, -4.8, \dots, 4.9, 5.0\}$$

$$\text{TC: } \{-5.0, -4.9, -4.8, \dots, 4.9, 5.0\}$$

Application of fuzzy control outputs:

New heat control setting = $\max[0.0, \min(10.0, \text{old setting} + \text{HC base value})]$

New throttle setting = $\max[0.0, \min(10.0, \text{old setting} + \text{TC base value})]$

9.5.2. Fuzzy PD Control

Base input variables:

$$P(0) = \text{actual pressure reading} - \text{pressure setpoint}$$

$$S(0) = \text{actual speed reading} - \text{speed setpoint}$$

$$PC(0) = P(0) - P(-1)$$

$$SC(0) = S(0) - P(-1)$$

Inference method: Same as for fuzzy P control.

Linguistic input variable relations (see Fig. 9.9):

P:	NB: $a_1 = -\infty$	$a_2 = -2.5$	$a_3 = -0.9$
	NS: $a_1 = -2.5$	$a_2 = -0.9$	$a_3 = 0.0$
	ZO: $a_1 = -0.9$	$a_2 = 0.0$	$a_3 = 0.9$
	PS: $a_1 = 0.0$	$a_2 = 0.9$	$a_3 = 2.5$
	PB: $a_1 = 0.9$	$a_2 = 2.5$	$a_3 = \infty$
S:	NB: $a_1 = -\infty$	$a_2 = -0.038$	$a_3 = -0.026$
	NS: $a_1 = -0.038$	$a_2 = -0.026$	$a_3 = 0.000$
	ZO: $a_1 = -0.026$	$a_2 = 0.000$	$a_3 = 0.026$
	PS: $a_1 = 0.000$	$a_2 = 0.026$	$a_3 = 0.038$
	PB: $a_1 = 0.026$	$a_2 = 0.038$	$a_3 = \infty$
PC:	NB: $a_1 = -\infty$	$a_2 = -0.575$	$a_3 = -0.225$
	NS: $a_1 = -0.575$	$a_2 = -0.225$	$a_3 = 0.000$
	ZO: $a_1 = -0.225$	$a_2 = 0.000$	$a_3 = 0.225$
	PS: $a_1 = 0.000$	$a_2 = 0.225$	$a_3 = 0.575$
	PB: $a_1 = 0.225$	$a_2 = 0.575$	$a_3 = \infty$
SC:	NB: $a_1 = -\infty$	$a_2 = -0.025$	$a_3 = -0.013$
	NS: $a_1 = -0.025$	$a_2 = -0.013$	$a_3 = 0.000$
	ZO: $a_1 = -0.013$	$a_2 = 0.000$	$a_3 = 0.013$
	PS: $a_1 = 0.000$	$a_2 = 0.013$	$a_3 = 0.025$
	PB: $a_1 = 0.013$	$a_2 = 0.025$	$a_3 = \infty$

Rules: As found in Table 9.2.

Linguistic output variable relations (see Fig. 9.9):

HC:	NB: $a_1 = -\infty$	$a_2 = -2.3$	$a_3 = -0.3$
	NS: $a_1 = -2.3$	$a_2 = -0.3$	$a_3 = 0.0$

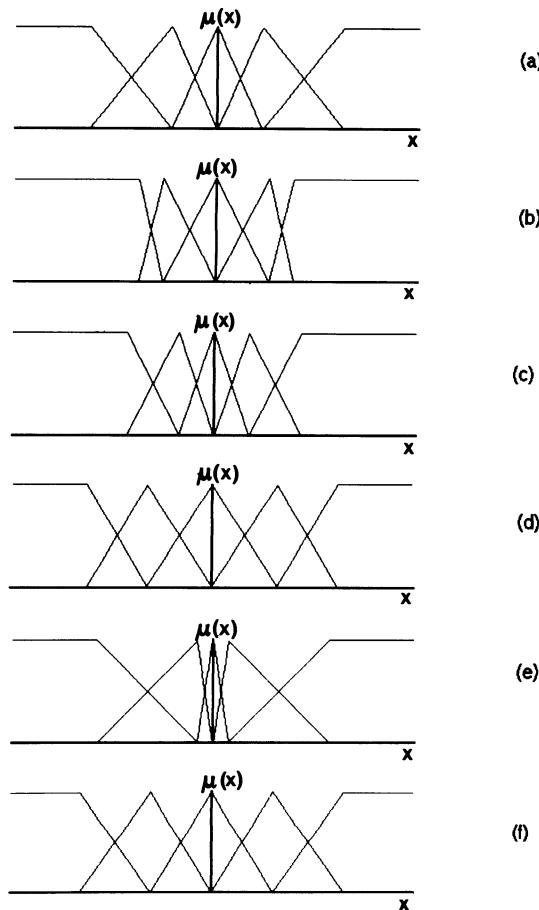


Figure 9.9. Fuzzy relations for linguistic variables in fuzzy PD control design; (a) P (pressure error), (b) S (speed error), (c) PC (pressure error change), (d) SC (speed error change), (e) HC (heat change), and (f) TC (throttle change).

$$\begin{array}{lll}
 \text{ZO: } a_1 = -0.3 & a_2 = 0.0 & a_3 = 0.3 \\
 \text{PS: } a_1 = 0.0 & a_2 = 0.3 & a_3 = 2.3 \\
 \text{PB: } a_1 = 0.3 & a_2 = 2.3 & a_3 = \infty \\
 \text{TC: } \text{NB: } a_1 = -\infty & a_2 = -2.6 & a_3 = -1.2 \\
 \text{NS: } a_1 = -2.6 & a_2 = -1.2 & a_3 = 0.0 \\
 \text{ZO: } a_1 = -1.2 & a_2 = 0.0 & a_3 = 1.2 \\
 \text{PS: } a_1 = 0.0 & a_2 = 1.2 & a_3 = 2.6 \\
 \text{PB: } a_1 = 1.2 & a_2 = 2.6 & a_3 = \infty
 \end{array}$$

Defuzzification method: Same as for fuzzy P control.

Base output variable definitions for use in defuzzification: Same as for fuzzy P control.

Application of fuzzy control outputs: Same as for fuzzy P control.

9.5.3. Fuzzy PI Control

Base input variables :

$$P(0) = \text{actual pressure reading} - \text{pressure setpoint}$$

$$S(0) = \text{actual speed reading} - \text{speed setpoint}$$

$$PS(0) = 0.1 \sum_{t=-9}^0 P(t)$$

Inference method: Same as for fuzzy P control.

Linguistic input variable relations (see Fig. 9.10):

P:	NB: $a_1 = -\infty$	$a_2 = -0.8$	$a_3 = -0.4$
	NS: $a_1 = -0.8$	$a_2 = -0.4$	$a_3 = 0.0$
	ZO: $a_1 = -0.4$	$a_2 = 0.0$	$a_3 = 0.4$
	PS: $a_1 = 0.0$	$a_2 = 0.4$	$a_3 = 0.8$
	PB: $a_1 = 0.4$	$a_2 = 0.8$	$a_3 = \infty$
S:	NB: $a_1 = -\infty$	$a_2 = -0.038$	$a_3 = -0.030$
	NS: $a_1 = -0.038$	$a_2 = -0.030$	$a_3 = 0.000$
	ZO: $a_1 = -0.030$	$a_2 = 0.000$	$a_3 = 0.030$
	PS: $a_1 = 0.000$	$a_2 = 0.030$	$a_3 = 0.038$
	PB: $a_1 = 0.030$	$a_2 = 0.038$	$a_3 = \infty$
PS:	NB: $a_1 = -\infty$	$a_2 = -1.95$	$a_3 = -0.85$
	NS: $a_1 = -1.95$	$a_2 = -0.85$	$a_3 = 0.00$
	ZO: $a_1 = -0.85$	$a_2 = 0.00$	$a_3 = 0.85$
	PS: $a_1 = 0.00$	$a_2 = 0.85$	$a_3 = 1.95$
	PB: $a_1 = 0.85$	$a_2 = 1.95$	$a_3 = \infty$

Rules: As found in Table 9.3.

Linguistic output variable relations (see Fig. 9.10):

HC:	NB: $a_1 = -\infty$	$a_2 = -1.2$	$a_3 = -0.2$
	NS: $a_1 = -1.2$	$a_2 = -0.2$	$a_3 = 0.0$
	ZO: $a_1 = -0.2$	$a_2 = 0.0$	$a_3 = 0.2$
	PS: $a_1 = 0.0$	$a_2 = 0.2$	$a_3 = 1.2$
	PB: $a_1 = 0.2$	$a_2 = 1.2$	$a_3 = \infty$
TC:	NB: $a_1 = -\infty$	$a_2 = -1.05$	$a_3 = -0.35$

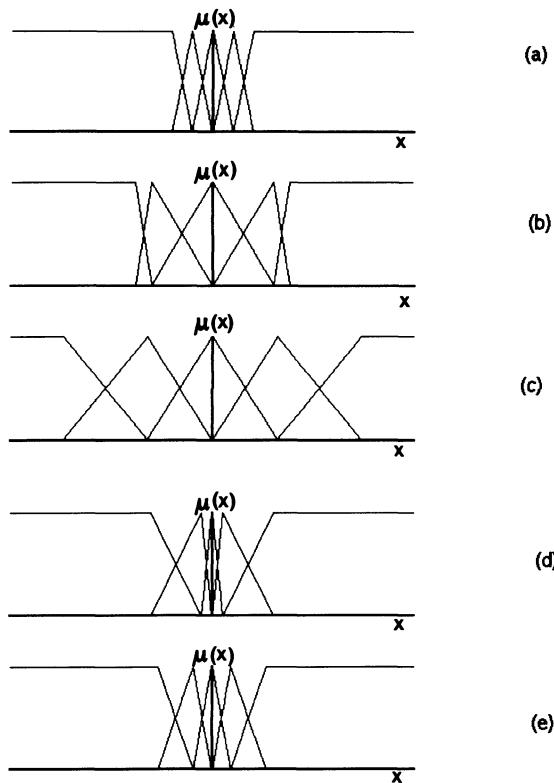


Figure 9.10. Fuzzy relations for linguistic variables in fuzzy PI control design; (a) P (pressure error), (b) S (speed error), (c) PS (pressure error average), (d) HC (heat change), and (e) TC (throttle change).

$$\text{NS: } a_1 = -1.05 \quad a_2 = -0.35 \quad a_3 = 0.00$$

$$\text{ZO: } a_1 = -0.35 \quad a_2 = 0.00 \quad a_3 = 0.35$$

$$\text{PS: } a_1 = 0.00 \quad a_2 = 0.35 \quad a_3 = 1.05$$

$$\text{PB: } a_1 = 0.35 \quad a_2 = 1.05 \quad a_3 = \infty$$

Defuzzification method: Same as for fuzzy P control.

Base output variable definitions for use in defuzzification: Same as for fuzzy P control.

Application of fuzzy control outputs: Same as for fuzzy P control.

9.5.4. Fuzzy PID Control

Base input variables

$P(0)$ = actual pressure reading – pressure setpoint

$S(0)$ = actual speed reading – speed setpoint

$$PC(0) = P(0) - P(-1)$$

$$SC(0) = S(0) - P(-1)$$

$$PS(0) = 0.1 \sum_{t=-9}^0 P(t)$$

Inference method: Same as for fuzzy P control.

Linguistic input variable relations (see Fig. 9.11):

P:	NB: $a_1 = -\infty$	$a_2 = -1.9$	$a_3 = -1.1$
	NS: $a_1 = -1.9$	$a_2 = -1.1$	$a_3 = 0.0$
	ZO: $a_1 = -1.1$	$a_2 = 0.0$	$a_3 = 1.1$
	PS: $a_1 = 0.0$	$a_2 = 1.1$	$a_3 = 1.9$
	PB: $a_1 = 1.1$	$a_2 = 1.9$	$a_3 = \infty$
S:	NB: $a_1 = -\infty$	$a_2 = -0.060$	$a_3 = -0.032$
	NS: $a_1 = -0.060$	$a_2 = -0.032$	$a_3 = 0.000$
	ZO: $a_1 = -0.032$	$a_2 = 0.000$	$a_3 = 0.032$
	PS: $a_1 = 0.000$	$a_2 = 0.032$	$a_3 = 0.060$
	PB: $a_1 = 0.032$	$a_2 = 0.060$	$a_3 = \infty$
PC:	NB: $a_1 = -\infty$	$a_2 = -1.10$	$a_3 = -0.25$
	NS: $a_1 = -1.10$	$a_2 = -0.25$	$a_3 = 0.00$
	ZO: $a_1 = -0.25$	$a_2 = 0.00$	$a_3 = 0.25$
	PS: $a_1 = 0.00$	$a_2 = 0.25$	$a_3 = 1.10$
	PB: $a_1 = 0.25$	$a_2 = 1.10$	$a_3 = \infty$
SC:	NB: $a_1 = -\infty$	$a_2 = -0.027$	$a_3 = -0.017$
	NS: $a_1 = -0.027$	$a_2 = -0.017$	$a_3 = 0.000$
	ZO: $a_1 = -0.017$	$a_2 = 0.000$	$a_3 = 0.017$
	PS: $a_1 = 0.000$	$a_2 = 0.017$	$a_3 = 0.027$
	PB: $a_1 = 0.017$	$a_2 = 0.027$	$a_3 = \infty$
PS:	NB: $a_1 = -\infty$	$a_2 = -1.6$	$a_3 = -0.8$
	NS: $a_1 = -1.6$	$a_2 = -0.8$	$a_3 = 0.0$
	ZO: $a_1 = -0.8$	$a_2 = 0.0$	$a_3 = 0.8$
	PS: $a_1 = 0.0$	$a_2 = 0.8$	$a_3 = 1.6$
	PB: $a_1 = 0.8$	$a_2 = 1.6$	$a_3 = \infty$

Rules: As found in Table 9.4.

Linguistic output variable relations (see Fig. 9.11):

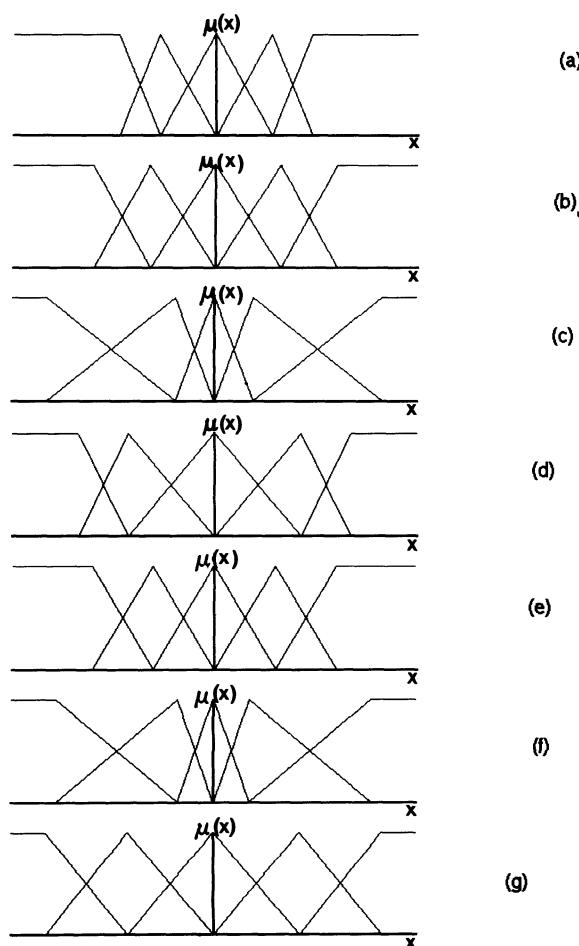


Figure 9.11. Fuzzy relations for linguistic variables in fuzzy PID control design; (a) P (pressure error), (b) S (speed error), (c) PC (pressure error change), (d) SC (speed error change), (e) PS (pressure error average), (f) HC (heat change), and (g) TC (throttle change).

HC :	NB : $a_1 = -\infty$	$a_2 = -3.1$	$a_3 = -0.7$
	NS : $a_1 = -3.1$	$a_2 = -0.7$	$a_3 = 0.0$
	ZO : $a_1 = -0.7$	$a_2 = 0.0$	$a_3 = 0.7$
	PS : $a_1 = 0.0$	$a_2 = 0.7$	$a_3 = 3.1$
	PB : $a_1 = 0.7$	$a_2 = 3.1$	$a_3 = \infty$
TC :	NB : $a_1 = -\infty$	$a_2 = -3.3$	$a_3 = -1.7$
	NS : $a_1 = -3.3$	$a_2 = -1.7$	$a_3 = 0.0$
	ZO : $a_1 = -1.7$	$a_2 = 0.0$	$a_3 = 1.7$

$$\begin{array}{lll} \text{PS: } a_1 = 0.0 & a_2 = 1.7 & a_3 = 3.3 \\ \text{PB: } a_1 = 1.7 & a_2 = 3.3 & a_3 = \infty \end{array}$$

Defuzzification method: Same as for fuzzy P control.

Base output variable definitions for use in defuzzification: Same as for fuzzy P control.

Application of fuzzy control outputs: Same as for fuzzy P control.

CHAPTER 10

Final Test and Results

Several important issues related to final testing and evaluation remain to be discussed. Fuzzy control literature continues to consider the topic of final testing in a somewhat *ad hoc* manner. Perhaps by considering types of specifications and tests mentioned in Section 2.6, it may eventually be possible to make final testing of fuzzy control devices a more standardized and simple process. However even conventional control designs typically pertain to complex systems and involve difficult tradeoffs, so that specifications and testing methods often must be viewed with some degree of flexibility. We have seen that fuzzy control designs often relate to systems that are more complex still, and typically involve even more tradeoffs, especially because they usually are multivariate in nature. It is therefore hardly surprising that the testing of fuzzy controllers is a topic that is difficult to treat in a generally standardized way.

The approaches used in this chapter will indeed be *ad hoc* in many respects, probably unavoidably so; but because we wish to make meaningful comparisons with respect to various forms of dynamic compensation we will emphasize objectivity as much as possible.

The objectives of the tests presented here are somewhat different from the objectives of final testing in a typical application because we compare the effectiveness of dynamic compensation as a general issue, however overall methods are still the same. Sections 10.1 and 10.2 present general examples of what final testing of a fuzzy controller involves, regardless of the application.

Section 10.3 offers some further insights into what happened during the autotuning process. These insights are based on only an informal investigation here, but they do suggest answers to our questions about dynamic compensation in fuzzy control.

10.1. Test Input Functions Used for Comparisons

One of the most critical aspects of the comparison is selecting test functions. This is naturally a serious difficulty when comparing complex designs for anything. Given two designs, A and B, even if Design A is superior to Design B in most circumstances, it may be possible to find a special case for which Design B gives the superior performance. In some contexts we can refer to standard or benchmark

tests, but this is quite difficult in the case of control-device designs partly because it would require choosing a standard test plant on which to base all tests. While there is perhaps no way of avoiding the inherent testing problems completely, we make reasonable selections in an unbiased manner and explain the rationale and assumptions on which these are based.

The testing is based as much as possible on standard procedures discussed in the conventional control literature; however we make one exception when it comes to a peculiarity of our steam engine test plant. We discuss this peculiarity and how we deal with it before continuing our discussion.

The heat-input control to the boiler of the steam engine, as described in Chapter 7, is effective for controlling the boiler pressure, but it is subject to certain limitations. The steam engine is designed to use solid alcohol as fuel, and once the burner is ignited, this fuel is consumed in about 10 minutes. It is possible to restoke the burner, but it is difficult to perform this operation in a precisely controlled way. Furthermore because the engine has no provision for recycling the steam, the boiler would run dry after only about 20 minutes; to allow this to happen would risk serious damage to the engine. Therefore as previously explained, we based the model on experiments with the engine between 2 minutes after ignition (about the time the pressure had built up sufficiently to operate the engine smoothly) and about 8 or 9 minutes after ignition (when pressure began to drop off due to fuel consumption). We also explained in previous chapters that the heat-control device did influence the pressure, but part of this was a long term influence, so that as an overall trend, the pressure generally increased during this period. These conditions present a problem in testing, especially in testing long term performance of control designs. Consider Fig. 10.1, a plot of pressure based on the model, designs, and software discussed in the previous two chapters. Just as the pressures resulting from using the various control designs are about to reach steady state conditions, they all suddenly begin increasing and moving away from the setpoint at the same rate. This particular test did not even run very long (less than 90 seconds), yet the results were distorted in such a way that they become meaningless for comparison purposes.

This situation occurred because the pressure had naturally increased to the point that even with the heat control set to zero, it still rose above the setpoint. All four control designs recognized that the best response possible at that time was to set the heat to zero, which is why they all diverge in unison from the setpoint. Actually the situation is exacerbated here by the choice of a late starting time after ignition (7 minutes, 30 seconds) and a baseline pressure value (16.5 PSI) already approaching the low end of the range attainable at that time. However a similar situation occurs even with an earlier starting time or a more carefully chosen base pressure. Furthermore, it is desirable not only to avoid reaching the limits of attainable pressures but also to avoid unintentionally approaching them too closely. If there is a reason to test performance under extreme conditions, then we can do so, but

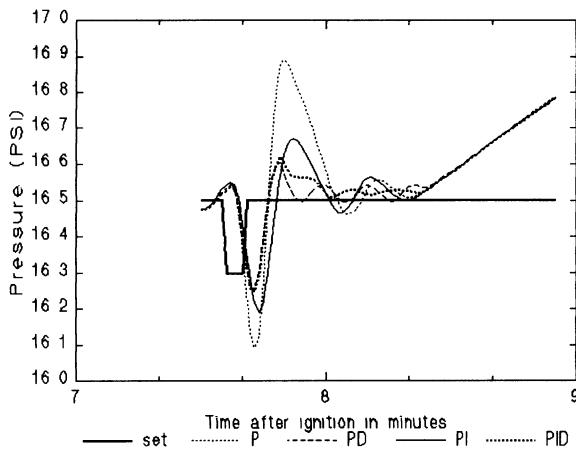


Figure 10.1. An example of the potential problem in testing control of boiler pressure.

otherwise we should avoid the possible distortions that can result from entering such a state unknowingly; therefore this problem requires some careful thought.

We begin with an experiment on the steam engine model itself. The steam engine model is virtually disconnected from all control devices, then run with constant settings of heat control and throttle. Note: This is not the same as running the complete system with constant setpoints for pressure and speed, but rather the control actuators are run at static settings. Three different pairs of settings are used

Case	Heat Control Setting	Throttle Setting
1	0.0	0.0
2	5.0	5.0
3	10.0	10.0

In all three cases the simulation is started from the same initial conditions (pressure = 10.8, speed = 0.070) at ignition + 1:30 (1 minute 30 seconds after ignition), an earlier time than the usual start of operation.

The results of this experiment are illustrated in Fig. 10.2, which shows a rather narrow band of attainable pressures between the two extreme pairs of settings. Note: This result is based on the neural model of the steam engine, not the steam engine itself. However as a strictly subjective judgment based on informal experimentation with the physical device, this is a more or less accurate description of the steam-engine behavior. In any case, in order to avoid distortions in the testing here, it is desirable to design the test functions in such a way that the pressure setpoint stays well within this band of attainable pressures wherever that is possible. One way to

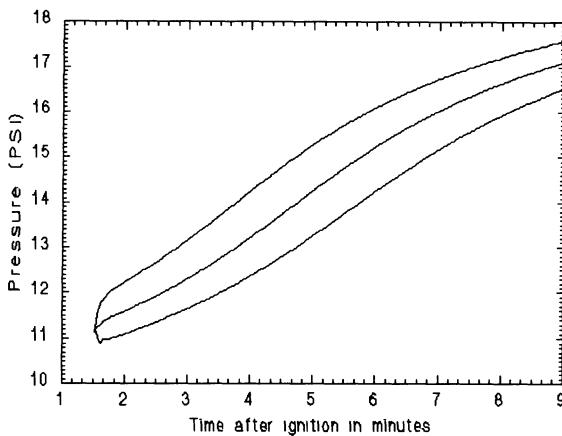


Figure 10.2. Pressure as a function of time after ignition under various static actuator settings.

accomplish this is obviously to keep the pressure setpoints fairly close to the curve that illustrates the steam-engine response to the constant intermediate settings of 5.0 for both the heat control and the throttle.

To make this more convenient, we fit a line to that curve by the usual regression method. We see from the graph that the behavior is very close to linear for the 5-minute period from 2:30–7:30; this is also the range in which most of the physical experimentation took place. Therefore we henceforth limit our testing to this range, and we perform the regression for this range also; the result is illustrated in Fig. 10.3. We refer to this as the baseline pressure, and it can be expressed precisely as follows:

$$\text{Baseline pressure} = 0.93660(\text{time in minutes}) + 9.5152$$

for times in the range of 2.5–7.5 minutes

Thus, the pressure tends to rise at a rate of slightly less than one PSI per minute.

The test functions will therefore be slightly modified from their standard forms, as will be discussed momentarily. Furthermore we consider carefully the size of the steps to be used in each case. This same type of peculiarity does not exist in the case of speed. A plot for speed, similar to Fig. 10.2 for pressure, is shown in Fig. 10.4. The only significant precaution necessary is to avoid a speed setpoint that is above the speed attainable at the existing pressure levels.

Let us now consider the matter of initial conditions. We use various types of test functions for pressure and speed. For each type of function, it is possible to consider an infinite number of initial conditions, step sizes, rates, and so on. Because we wish to show plots and discuss the results of each case considered, we cannot

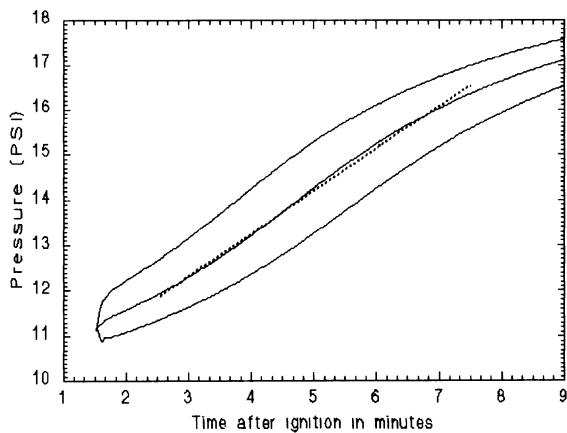


Figure 10.3. Pressure under various static settings, with regression line fit to middle settings.

consider too many combinations of test function types, initial conditions, and other parameters. Therefore we use one standard set of initial conditions for all test functions but additionally show one of the test functions under a few other sets of initial conditions as well.

The next step is to decide what these standard initial conditions should be. One key factor is time, which has a direct influence on pressure behavior and an indirect influence on speed. To avoid the effects of extreme conditions, and for the lack of a better guide, we design the test functions so that the performance evaluation occurs

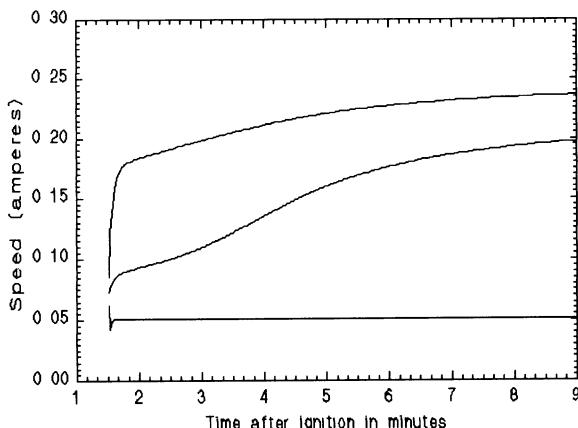


Figure 10.4. Speed as a function of time after ignition under various static actuator settings.

during the center section of the effective performance range. In the case of pressure it is best to allow about 1.5 minutes to observe behavior as the system settles back toward steady state conditions, somewhat less in the case of speed. Therefore let us assume the standard that the start of the test deviation (step, or beginning of impulse or ramp) occurs at ignition + 4:30. Under the typical conditions for pressure, the evaluation continues to ignition + 6:00, but this can be extended if necessary without exceeding the limits of the performance range. It is also desirable to allow the system to stabilize for a few seconds before the test function deviation begins. Therefore for the standard initial conditions, the simulation begins at ignition +4:20.

Other initial conditions can also be easily decided, by tending toward the central values in the ranges. For heat input and throttle settings, settings of precisely 5.0 arbitrary units are used in each case. As one would expect from the way we arrived at the pressure baseline, this heat-input setting is almost precisely correct for maintaining that baseline. This means that if both the pressure setpoint and the pressure actual value are initialized to the baseline value of about 13.57 PSI at 4:20, then there is little need for the control device to attempt to adjust the pressure until the setpoints are adjusted away from the baseline. As for speed, by experimentation we see that this pressure level and original throttle setting are about right for maintaining a reading of 0.150 A; therefore the initial actual value and the speed setpoint are set to that point. Furthermore during the course of a pressure test function simulation, the speed setpoint is maintained at 0.150 A throughout. On the other hand, in the case of a speed test function simulation, the pressure setpoint moves with the pressure baseline throughout.

We limit ourselves to piecewise linear test functions here; at times quadratic functions are used in conventional control evaluation, but this is by no means assumed necessary in all cases. Thus we are interested here in step, impulse, and ramp functions, which is typical of the standard procedures used in the conventional theory. However unlike conventional control analysis, there is no particular reason to emphasize the concept of unit functions, such as unit-step functions. This means we can choose step sizes, rates, and durations according to what seems most appropriate to the example rather than what would be most convenient in the analytical calculations.

To distinguish easily among the several types of test functions we will use, let us adopt the following simple terminology. We refer to each step function as either a step-up or a step-down function, depending on the direction we move the setpoint at time $t = 0$, which in this case means ignition + 4:30. For impulse functions a step-up-down function indicates a case where the setpoint is moved up at $t = 0$, then back down again at some later time $t = k$; a step-down-up function indicates an impulse function moving in the opposite directions. Some may argue that this is a misnomer, because impulse functions are distinguished from step functions in the usual terminology, but we feel that step-up-down is easily understood and intuitively appealing. A ramp-up function refers to a ramp test function whose setpoint

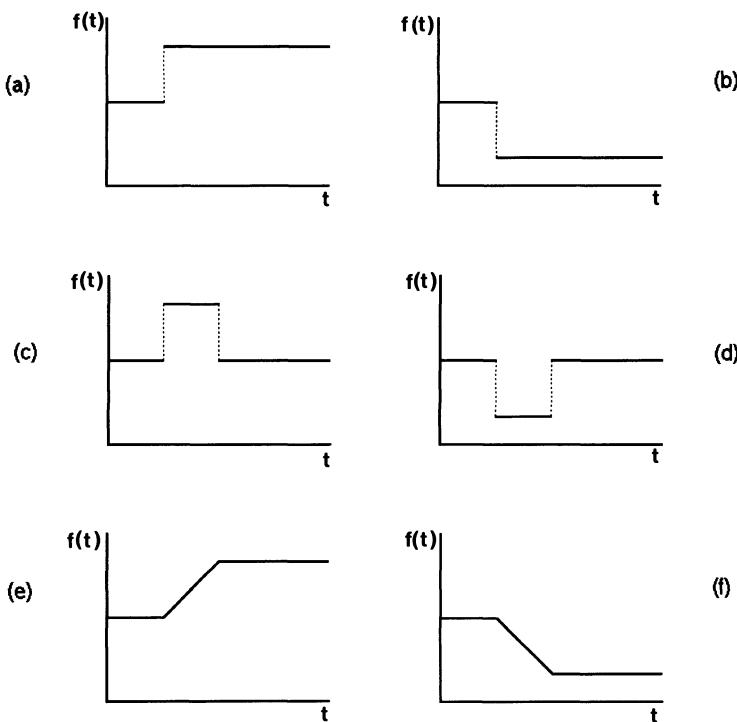


Figure 10.5. Basic test function types to be used; (a) step-up test function, (b) step-down test function, (c) step-up-down impulse test function, (d) step-down-up impulse test function, (e) ramp-up test function, and (f) ramp-down test function.

begins to increase at $t = 0$; a ramp-down function's setpoint decreases. Furthermore, we limit the period of increase or decrease of the ramp functions we used here to avoid approaching the limits of system performance.

This results in a total of six basic test function types, as illustrated in Fig. 10.5. Although it doubles the number of cases to be considered, we consider both directions because in some cases subtly different results are obtained for step-down functions as opposed to step-up for example. These function types are already fine in the case of speed, but in the case of pressure, they should be slightly modified to incorporate the concept of the sloping baseline previously discussed. The nature of these modified functions is illustrated in Fig. 10.6.

All that remains in identifying the test functions to be used is to determine a few more parameters; specifically step sizes, rates of change, and durations. For pressure step size is a critical issue because steps must be large enough to create significant effects but not so large that they push systems to the extremes of the fairly narrow attainable range. It turns out that 0.50 PSI is just about right, so it is

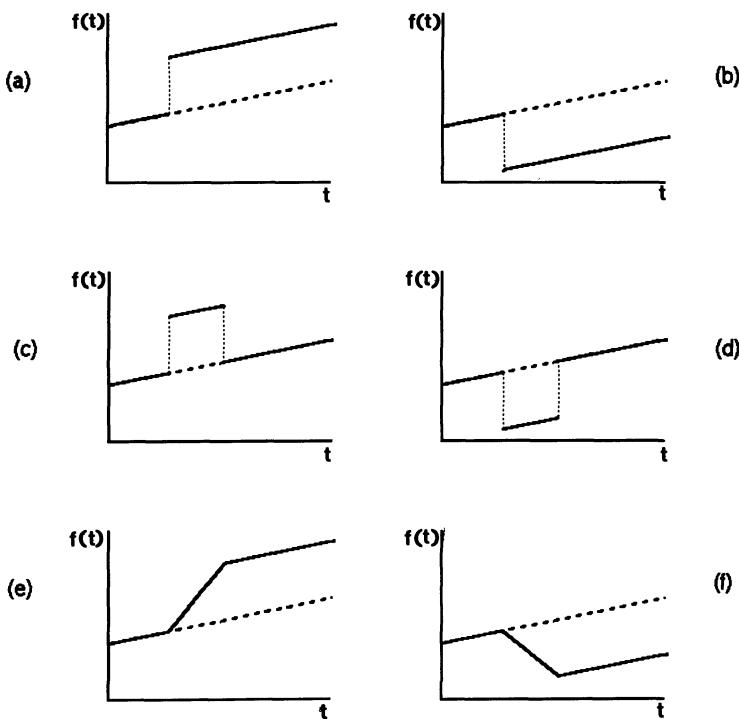


Figure 10.6. Test function types adapted to the concept of a sloping baseline for pressure; (a) step-up test function, (b) step-down test function, (c) step-up-down impulse test function, (d) step-down-up impulse test function, (e) ramp-up test function, and (f) ramp-down test function.

used for all four types of step and impulse functions. For impulse functions we chose a step (impulse) duration (k) of only 10 seconds, meaning that the setpoint returns to the baseline at ignition + 4:40. Because the response time for pressure is quite slow, 10 seconds is a relatively short period that creates a dramatic result for comparing the control-design performances. In the case of the ramp functions it is appropriate to choose a rate of change that is near the limit to which the control designs can respond but does not push too hard at that limit. Experimentation showed 1.50 PSI/min seemed to be about right. If the overall magnitude of change (away from the baseline) is to be 0.50 PSI once again, then this implies that the ramp will continue for exactly 20 seconds, that is until ignition + 4:50.

For speed, the choice of parameters is somewhat less critical, but it is still important to choose values large enough to demonstrate dramatically the capabilities of the control designs but small enough not to push the systems too much to the extremes. Furthermore all other factors being equal, it is desirable to keep the functions as consistent as possible with those used for pressure and to choose round

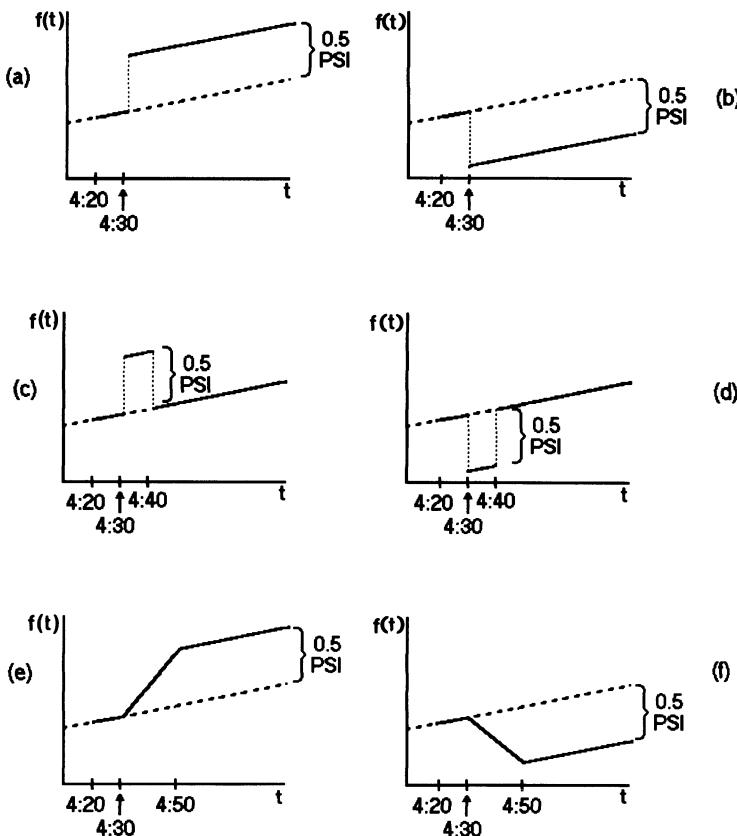


Figure 10.7. Nature of the 12 test functions used; (a) pressure step-up test function, (b) pressure step-down test function, (c) pressure step-up-down impulse test function, (d) pressure step-down-up impulse test function, (e) pressure ramp-up test function, (f) pressure ramp-down test function, (g) speed step-up test function, (h) speed step-down test function, (i) speed step-up-down impulse test function, (j) speed step-down-up impulse test function, (k) speed ramp-up test function, and (l) speed ramp-down test function.

values for the parameters. With this in mind it is reasonable to use 0.05 A as a step size. Because the initial condition is 0.15, this results in a leap either up to 0.200 or down to 0.100, which is well within the attainable range in speeds for moderate pressures. We can choose a 10-second step duration again for the sake of consistency with the pressure test functions. Because speed responds relatively quickly, this period is not particularly short, but nonetheless it is clear that it leaves insufficient time for the system to approach a steady state. Similarly, for the ramp functions, we use a 20-second duration, again for the sake of consistency with the

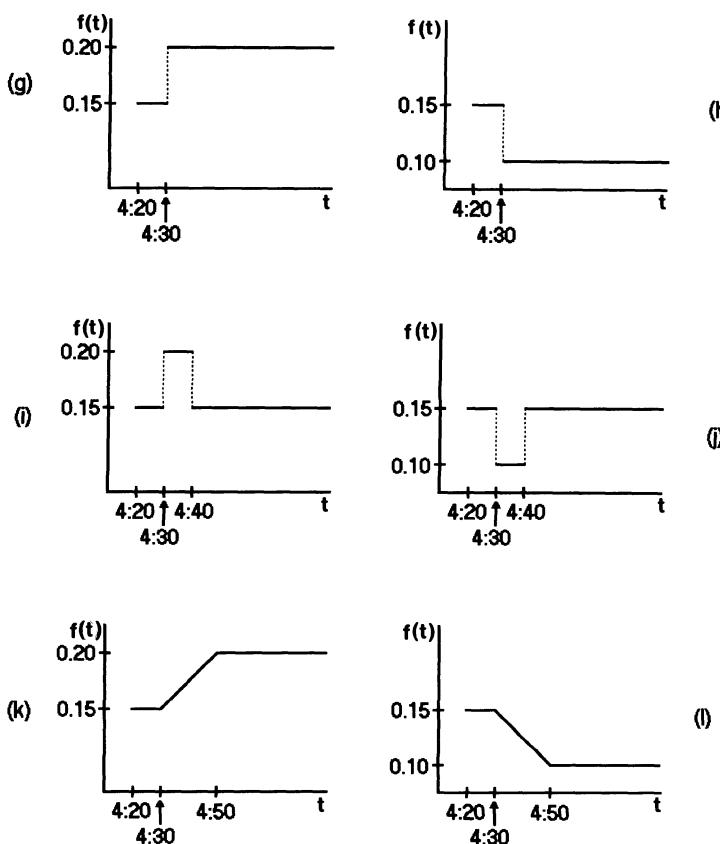


Figure 10.7. (Continued)

pressure case. If we also wish to maintain the same magnitude of change as for the step functions, this implies a rate of change of 0.150 A/min. The 12 resulting functions are illustrated in Fig. 10.7. Note: We deviate slightly from standard practice by limiting the ramps.

10.2. Results of Simulations

Results of the various test functions just discussed should provide useful information, however we cannot expect interpreting those results to be perfectly straightforward due to certain complexities, which we mention before discussing the results themselves.

To address the major complexity we must consider the nature of the benefits we expect to derive from dynamic compensation and the nature of auto-tuning processes. In Chapter 2 we considered some of the views held by those who study conventional control theory; these views should apply to fuzzy control methods as well. One such view is that control-device design typically involves difficult compromises or tradeoffs between competing demands for better responsiveness, better stability, and better accuracy. Thus when we say for example that a derivative term allows greater stability, we mean that a derivative element somewhat eases the tension in the conflict between stability performance and the other two performance aspects. The derivative term should therefore allow the designer to improve stability while leaving responsiveness and accuracy unchanged. However we can just as accurately say that the derivative term allows the designer to improve responsiveness and accuracy while leaving stability unchanged or that all three can be improved in moderate degrees. It all depends on how the designer tunes the control device on the basis of the designer's perceptions of how the compromises should be resolved in the performance goals.

For various reasons explained in Chapter 9, we rely heavily on auto-tuning in developing the designs used here. Therefore it is important to consider what auto-tuning means relative to the view that control design is a process of compromise. Perhaps the most reasonable way to express this connection is to say that the auto-tuning mechanism does resolve the compromise issue, but it does so in an obviously rather mindless way. In the particular approach used here, the mechanism made the compromises strictly on the basis of the particular test function chosen and the weighted ITAE-scoring method. Thus although the tradeoff is made, there is no obvious, detailed explanation of what it is. If for example a derivative term is useful in fuzzy control, then we can expect the performance of the fuzzy PD design to be superior to that of the fuzzy P design in some overall sense; however there is no way of predicting precisely which aspects of performance the auto-tuning mechanism will emphasize in that superiority.

This factor causes more potential confusion still in the sense that the control designs here are all structured and tuned in a truly multivariate way. The result is we cannot predict with precision how much the various designs emphasize pressure performance as opposed to speed performance.

It is reasonable to ask whether this difficulty could have been avoided by using a different overall approach in the current research. One apparent possibility would have been for example to design the fuzzy PD controller in such a way that its responsiveness and accuracy were identical to that of the fuzzy P controller, then see how much its stability was improved over that of the fuzzy P. If such an approach were possible, then obviously it would be easier to compare the two designs specifically in regards to stability. Unfortunately this sort of approach appears to be impossible. First how do we state in a general sense that two control designs are identical in their responsiveness or accuracy? Rigorous comparisons can be made

only if we agree on a specific test function and a specific method of measuring the attribute. Second, even if it were possible to state that two designs are identical in one aspect of performance, how would it be possible to ensure that they remain identical in that regard in the face of changes in some other aspect? It is the nature of control systems that nearly every aspect of the design influences nearly every aspect of the performance.

A second thing that could have been done differently in the overall approach is to have based the designs on simpler structures, a simpler test plant. First this would not eliminate the problem entirely, though obviously avoiding a multivariate structure would make the situation somewhat simpler. Second, we considered this possibility when we meticulously discussed in previous chapters the rationale behind our choice of test plant and overall design structure.

In spite of this difficulty, we can still find meaning in the results in terms of overall comparisons. It is only in the point-by-point comparisons that we need to avoid being disappointed by our preconceived notions.

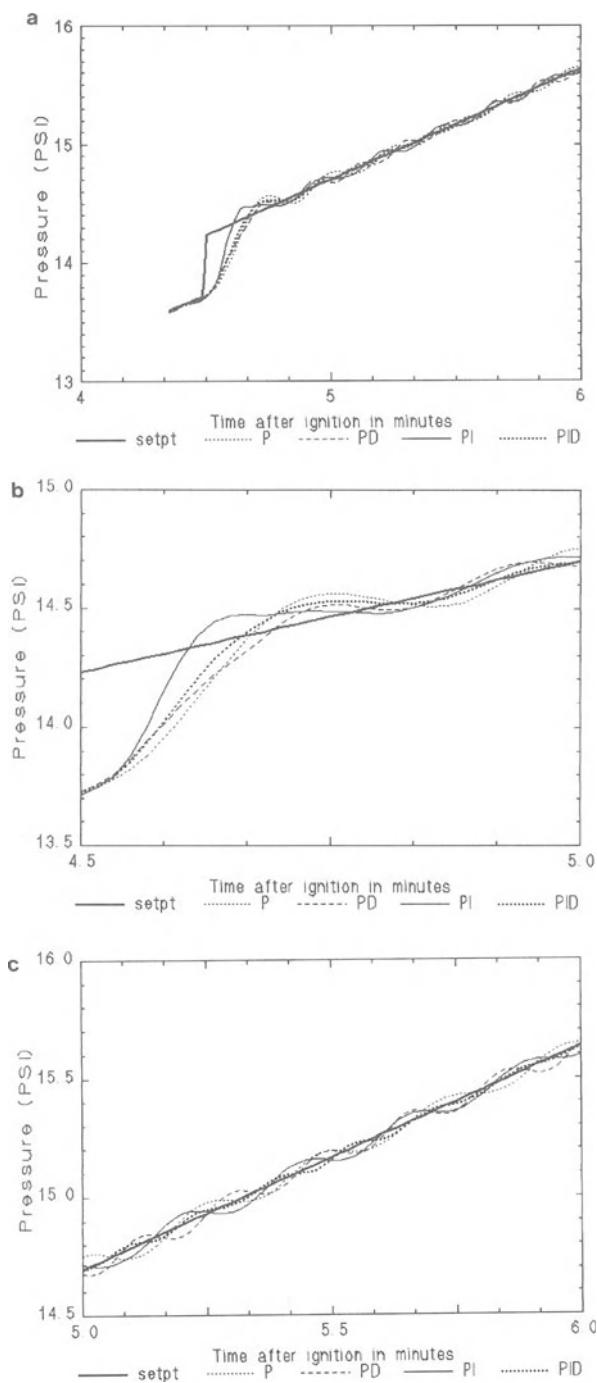
There are a few minor points to keep in mind to avoid confusion: Time lags are very different for pressure and speed. Also, we decided not to use an integral term for speed errors, which means that the influences of the integral terms on speed performance are of a somewhat indirect nature only.

Although it is somewhat difficult to see in this way, we plotted the performances of all four designs for each test function on the same axes to make four-way comparisons. To help with clarity we include three views for each of the functions presented. These show the overall function, the transient behavior segment, and the more or less steady state segment, respectively. This allows different scalings, which should make the images somewhat clearer.

10.2.1. Pressure Step-up Test Function (see Fig. 10.8a–c)

- Responsiveness (80% rise time): Best—PI, second best—PID, third best—P, worst—PD
- Transient stability (maximum percent overshoot): Best—PD (10.3), second best—PID (15.1), worst—P and PI (19.1)
- Steady state accuracy and stability (average absolute error in PSI from 5:00–6:00): Best by far—PID (0.0107), second best—PI (0.0255), third best—P (0.0264), worst—PD (0.0269)

Figure 10.8. Response to pressure step-up functions; (a) full view of response under standard conditions, (b) view of transient response under standard conditions, (c) view of steady state response under standard conditions, (d) full view of response under early start conditions, (e) view of transient response under early start conditions, (f) view of steady state response under early start conditions, (g) full view of response under late start conditions, (h) view of transient response under late start conditions, and (i) view of steady state response under late start conditions.



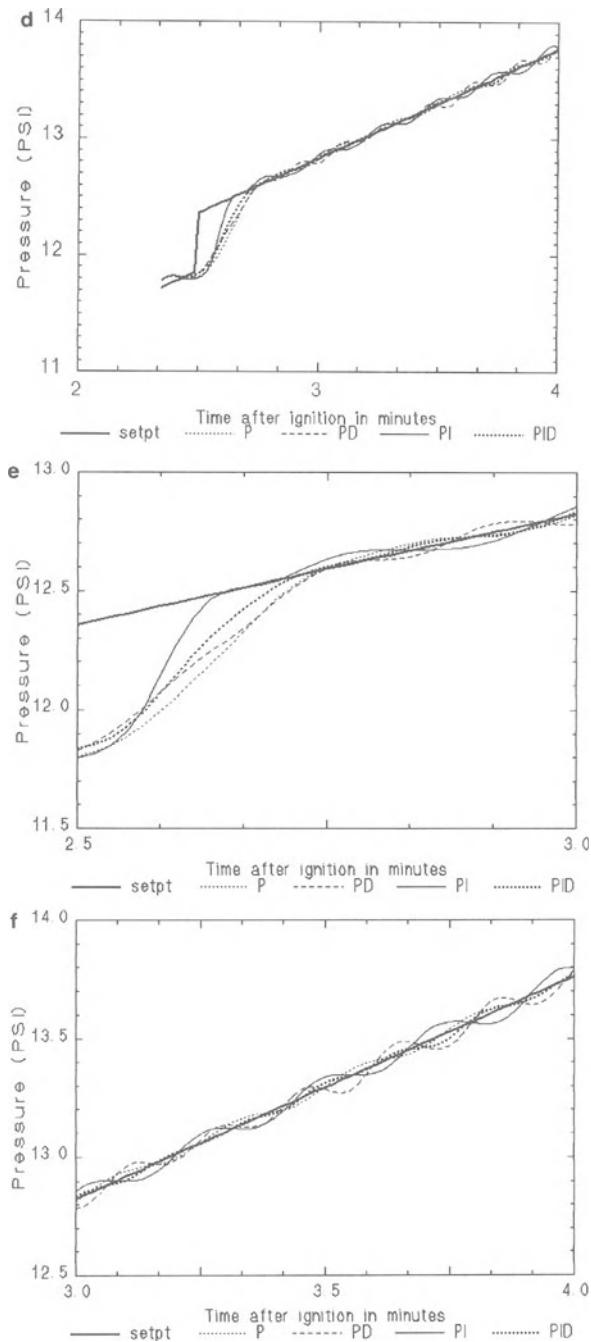


Figure 10.8. (Continued)

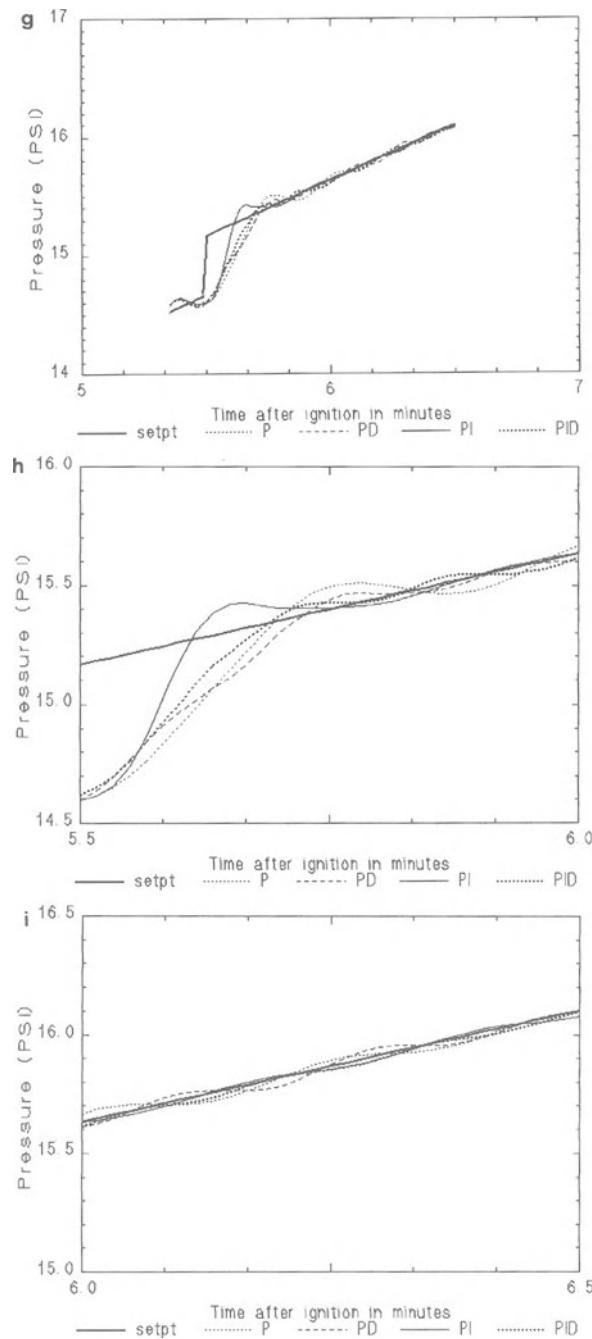


Figure 10.8. (Continued)

10.2.2. Pressure Step-up Test Function under Early Initial Conditions (see Fig. 10.8d–f)

- Responsiveness (80% rise time): Best—PI, second best—PID, worst—P and PD
- Transient stability (maximum percent overshoot): Best—PID (3.4), second best—P (5.4), third best—PD (6.7), worst—PI (9.1)
- Steady state accuracy and stability (average absolute error in PSI from 3:00–4:00): Best by far—PID (0.0096), second best—P (0.0145), third best—PI (0.0291), worst—PD (0.0302)

10.2.3. Pressure Step-up Test Function under Late Initial Conditions (see Fig. 10.8g–i)

- Responsiveness (80% rise time): Best—PI, second best—PID, third best—P, worst—PD
- Transient stability (maximum percent overshoot): Best—PID (8.1), second best—PD (8.9), third best—P (18.2), worst—PI (22.5)
- Steady state accuracy and stability (average absolute error in PSI from 6:00–6:30): Best—PI (0.0118) and PID (0.0119), third best—PD (0.0182), worst—P (0.0189)

10.2.4. Pressure Step-down Test Function (see Fig. 10.9)

- Responsiveness (80% rise time): Best—PI, second best—PID, third best—PD, worst—P
- Transient stability (maximum percent overshoot): Best—PI (13.1), second best—PD (13.5), third best—PID (15.8), worst—P (16.3)
- Steady state accuracy and stability (average absolute error in PSI from 5:00–6:00): Best by far—PID (0.0073), second best—PI (0.0282), worst—P (0.0371) and PD (0.0375)

10.2.5. Pressure Step-up-down Impulse Function (see Fig. 10.10)

- Responsiveness (80% rise time): Best—PI, second best—PD and PID, worst—P
- Transient stability (maximum percent overshoot): Best—PD (15.5), second best—PID (21.5), third best—P (34.0), worst—PI (63.9)
- Steady state accuracy and stability (average absolute error in PSI from 5:00–6:00): Best by far—PID (0.0137), second best—P (0.0377), worst—PD (0.0432) and PI (0.0433)

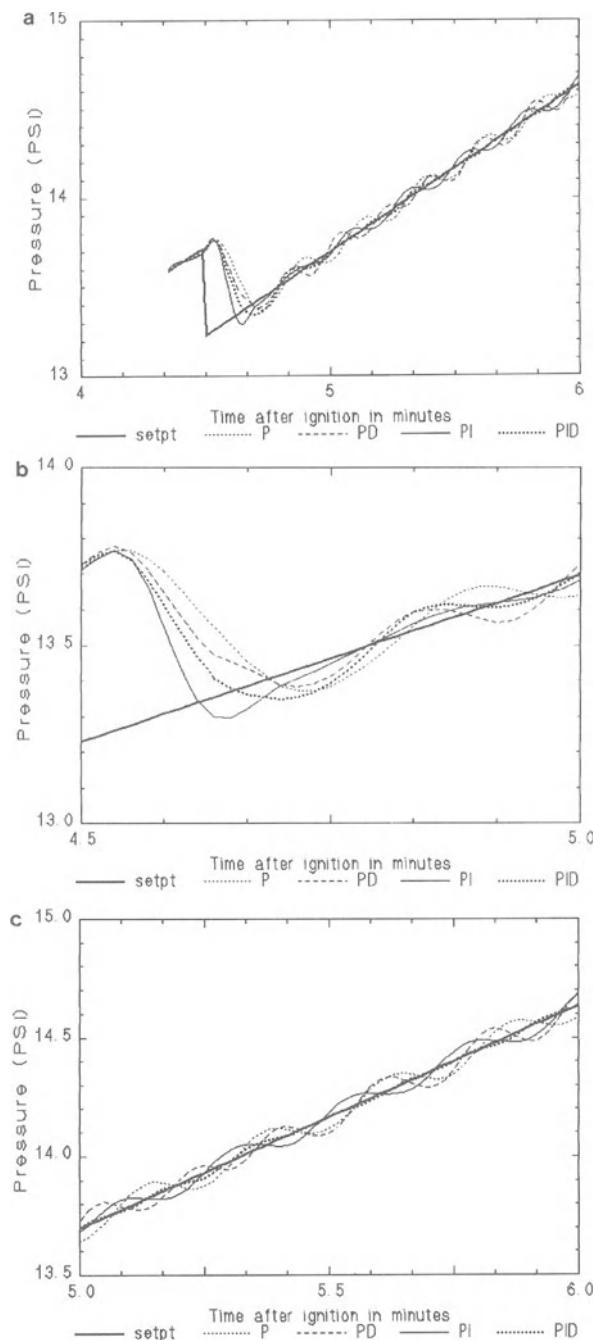


Figure 10.9. Response to pressure step-down function; (a) full view of response, (b) view of transient response, and (c) view of steady state response.

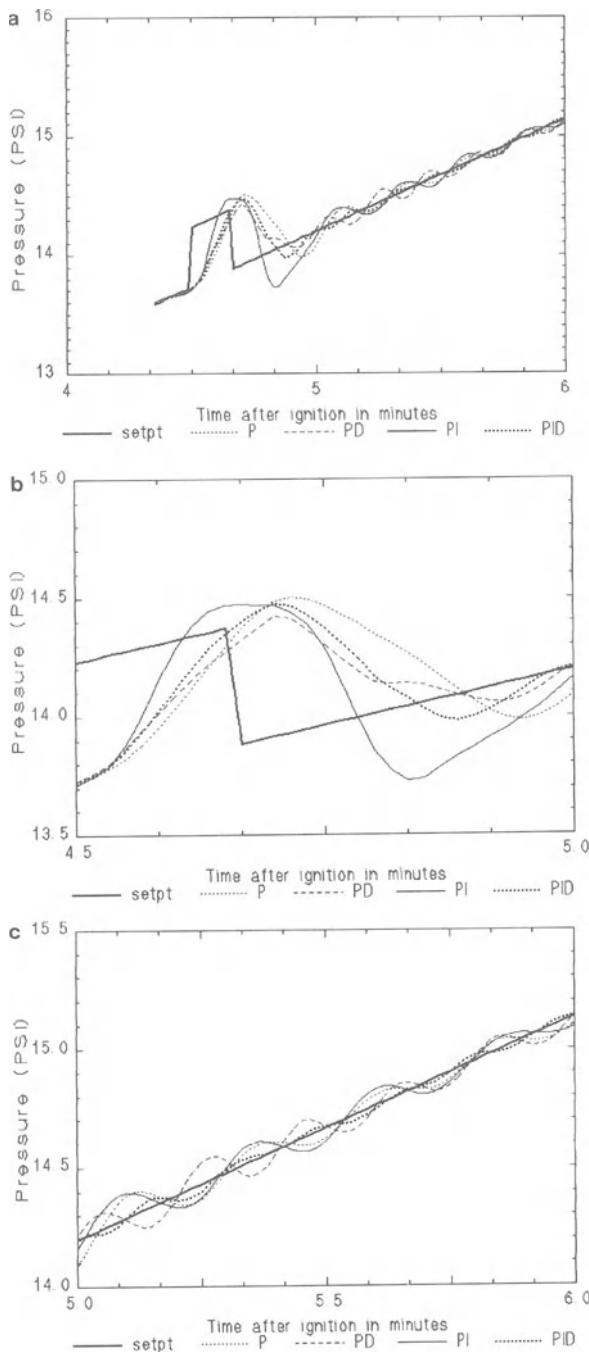


Figure 10.10. Response to pressure step-up-down impulse function; (a) full view of response, (b) view of transient response, and (c) view of steady state response.

10.2.6. Pressure Step-down-up Impulse Function (see Fig. 10.11)

- Responsiveness (80% rise time): Best—PI, second best—PD and PID, worst—P
- Transient stability (maximum percent overshoot): Best—PD (9.8), second best—PID (13.4), third best—P (35.0), worst—PI (52.9)
- Steady state accuracy and stability (average absolute error in PSI from 5:00–6:00): Best by far—PID (0.0195), second best—P (0.0392), worst—PD (0.0404) and PI (0.0405)

10.2.7. Pressure Ramp-up Test Function (see Fig. 10.12)

- Responsiveness to ramp (average absolute error in PSI in following the ramp.): Best—PI (0.036), second best—P and PD (0.043), worst—PID (0.045)
- Transient stability (maximum percent overshoot after ramp): Best—PD (6.0), second best—PID (6.7), third best—P (11.9), worst—PI (12.8)
- Steady state accuracy and stability (average absolute error in PSI from 5:00–6:00): Best by far—PID (0.0135), second best—PD (0.0257), worst—P (0.0303) and PI (0.0336)

10.2.8. Pressure Ramp-down Test Function (see Fig. 10.13)

- Responsiveness to ramp (average absolute error in PSI in following the ramp.): Best—PID (0.052), second best—PD (0.053), third best—P (0.055) worst—PI (0.065)
- Transient stability (maximum percent overshoot after ramp): Best—PI (6.7), second best—P (8.1), third best—PID (12.3), worst—PD (14.9)
- Steady state accuracy and stability (average absolute error in PSI from 5:00–6:00): Best by far—PID (0.0137), second best—P (0.0271), worst—PI (0.0367) and PD (0.0424)

10.2.9. Speed Step-up Test Function (see Fig. 10.14)

- Responsiveness (80% rise time): Best—P, PD, and PI all about the same, worst—PID
- Transient stability (maximum percent overshoot): Best—PD (0.5) and PID (slight undershoot), worst—P and PI (10.6)
- Steady state accuracy and stability (average absolute error in mA from 5:00–5:30): Best—PD (0.495), second best—PID (0.570), third best—P (0.695), worst—PI (0.931)

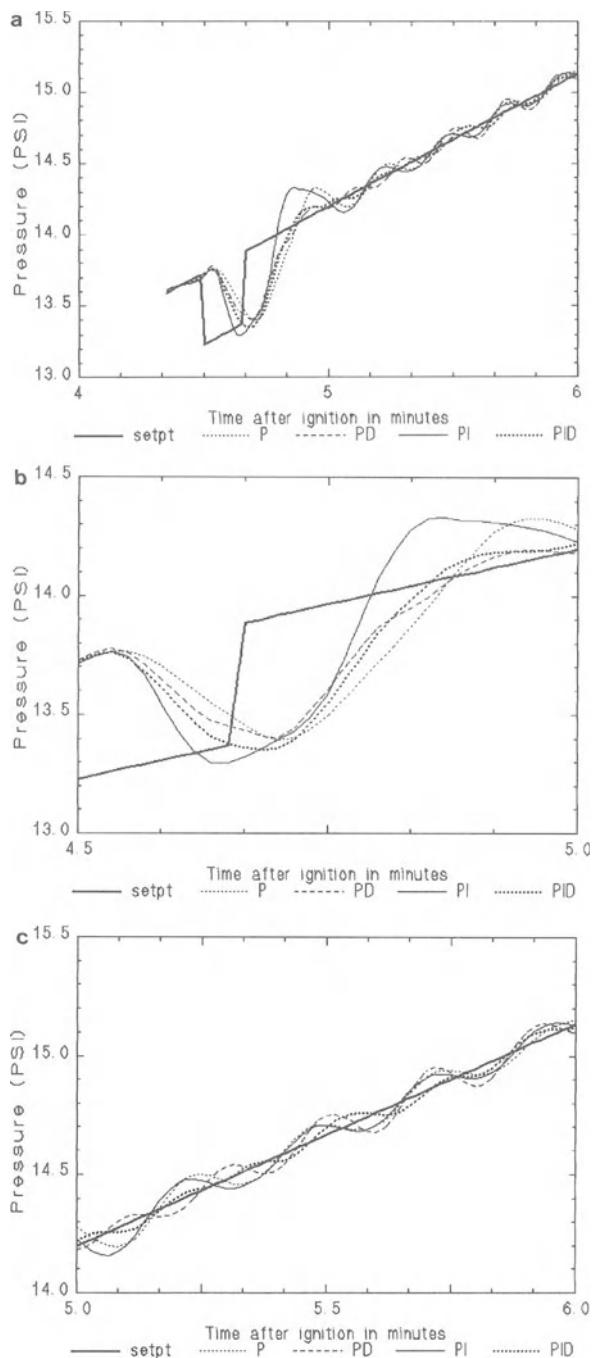


Figure 10.11. Response to pressure step-down-up impulse function; (a) full view of response, (b) view of transient response, and (c) view of steady state response.

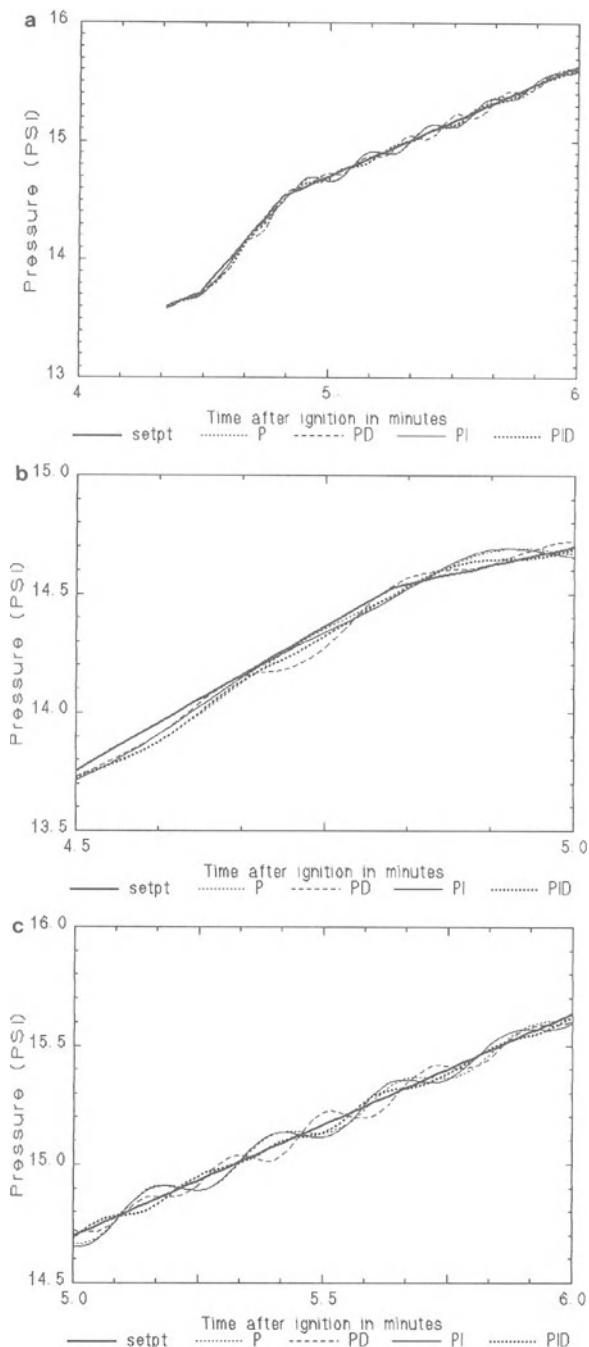


Figure 10.12. Response to pressure ramp-up function; (a) full view of response, (b) view of transient response, and (c) view of steady state response.

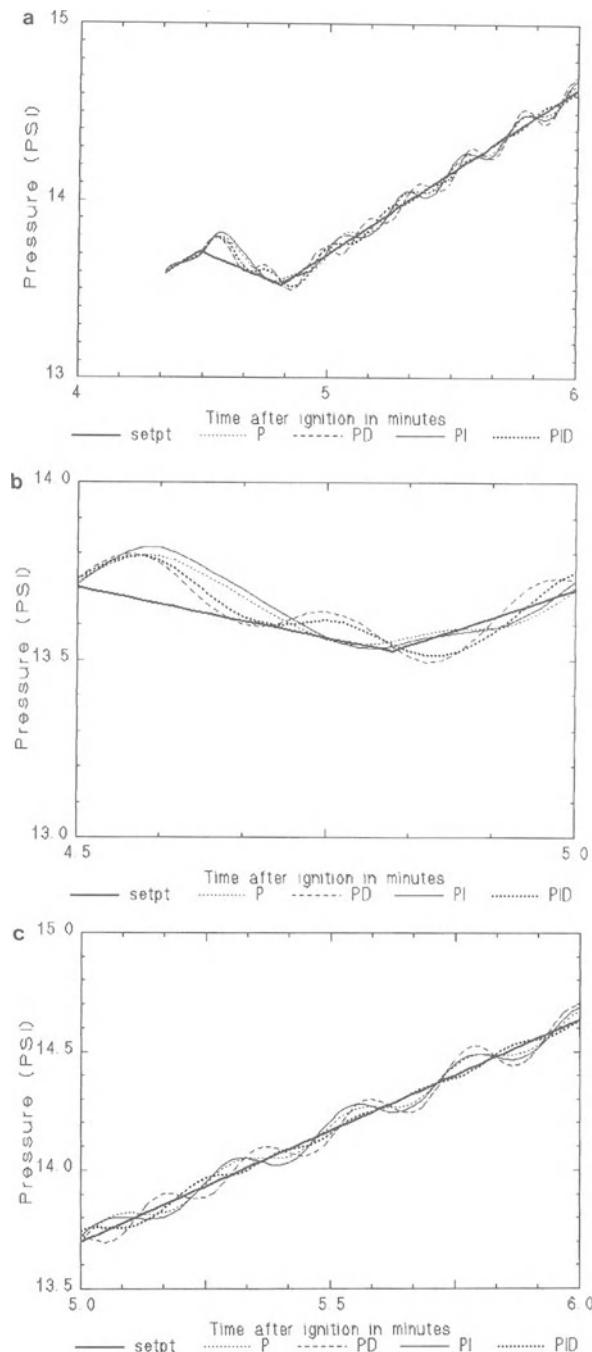


Figure 10.13. Response to pressure ramp-down function; (a) full view of response, (b) view of transient response, and (c) view of steady state response.

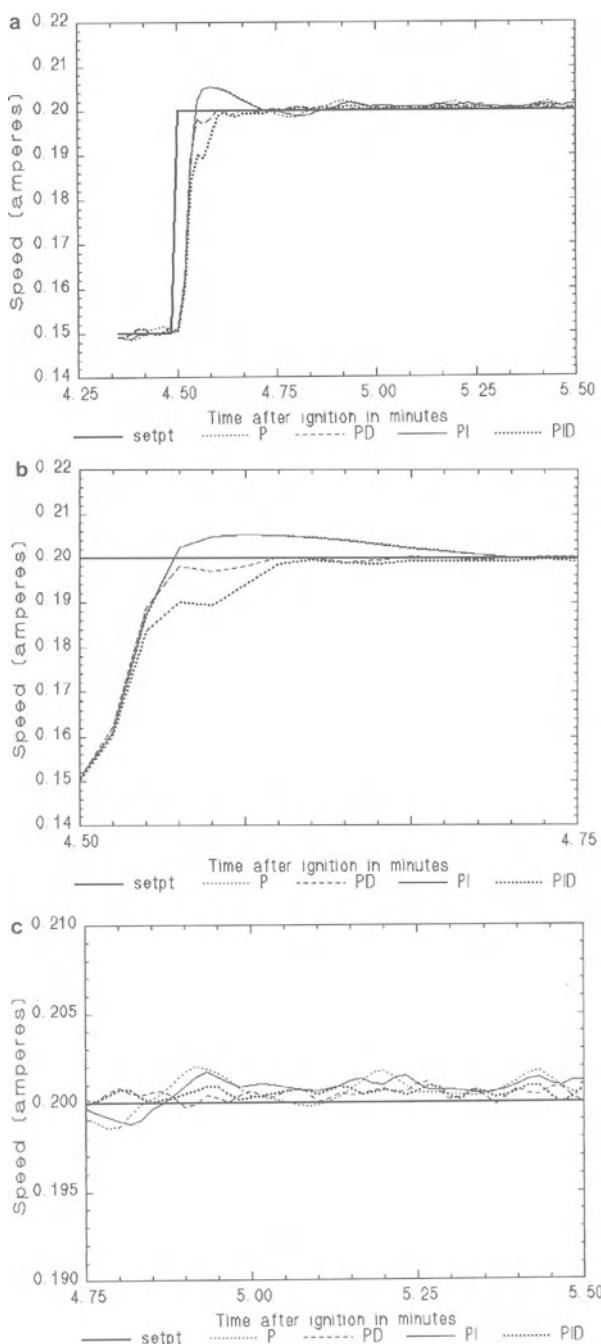


Figure 10.14. Response to speed step-up function; (a) full view of response, (b) view of transient response, and (c) view of steady state response.

10.2.10. Speed Step-down Test Function (see Fig. 10.15)

- Responsiveness (80% rise time): Best—PI, second best—P and PD, worst—PID
- Transient stability (maximum percent overshoot): Best—PID (6.7), second best PD (24.1), third best—P (56.5), worst—PI (67.3)
- Steady state accuracy and stability (average absolute error in mA from 5:00–5:30): Best—PID (0.328), second best—PD (0.584), third best—PI (0.688), worst—P (0.831)

10.2.11. Speed Step-up-down Impulse Test Function (see Fig. 10.16)

- Responsiveness (80% rise time): Best—P and PI, second best—PD, worst—PID
- Transient stability (maximum percent overshoot): Best—PD (1.4), second best—PID (2.2), third best—P (43.7), worst—PI (46.8)
- Steady state accuracy and stability (average absolute error in mA from 5:10–5:30): Best—PD (0.399), second best—PI (0.433), third best—PID (0.523), worst—P (0.859)

10.2.12. Speed Step-down-up Impulse Test Function (see Fig. 10.17)

- Responsiveness (80% rise time): Best—P and PI, second best—PD, worst—PID
- Transient stability (maximum percent overshoot): Best—PID (11.5), second best—PD (23.1), worst—P (57.8) and PI (59.0)
- Steady state accuracy and stability (average absolute error in mA from 5:10–5:30): Best—PD (0.258), second best—P (0.578) and PID (0.589), worst—PI (0.870)

10.2.13. Speed Ramp-up Test Function (see Fig. 10.18)

- Responsiveness to ramp (average absolute error in mA in following the ramp.): Best—PD (2.62), second best—PID (2.95), third best—P (4.34), worst—PI (4.50)
- Transient stability (maximum percent overshoot): Best—PID (1.5), second best PD (1.8), worst—P (2.5) and PI (3.0)
- Steady-state accuracy and stability (average absolute error in mA from 5:10–5:30): Best—PD (0.438) and PID (0.427), third best—PI (0.881), worst—P (0.996)

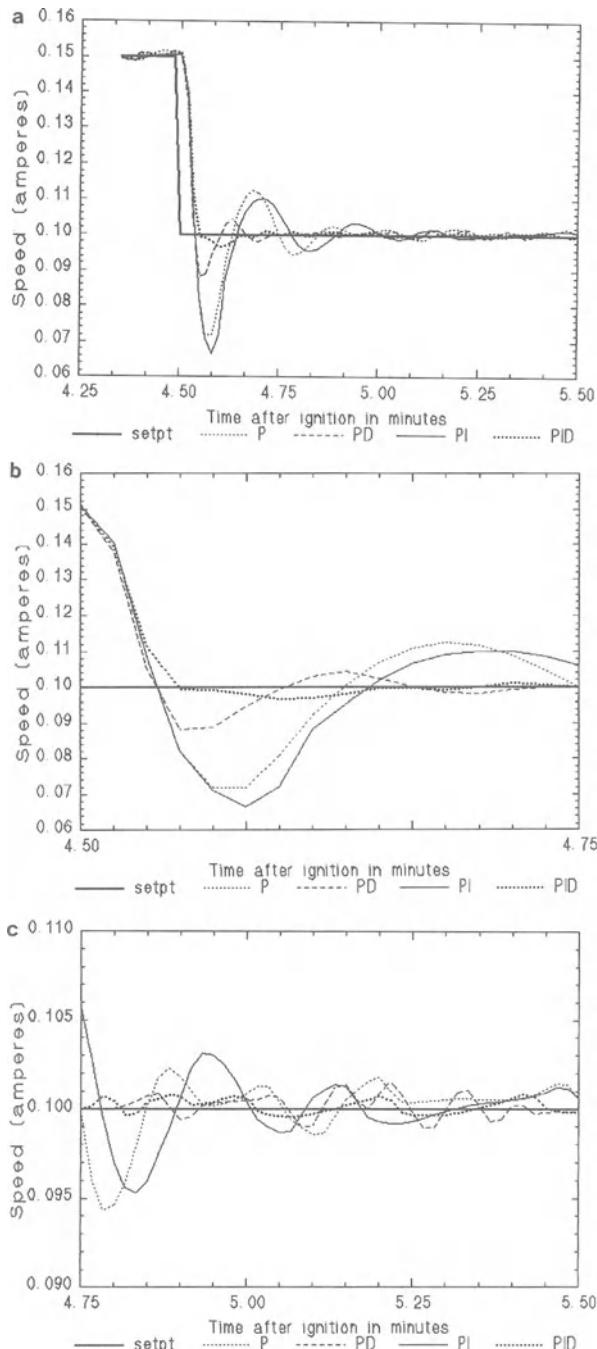


Figure 10.15. Response to speed step-down function; (a) full view of response, (b) view of transient response, and (c) view of steady state response.

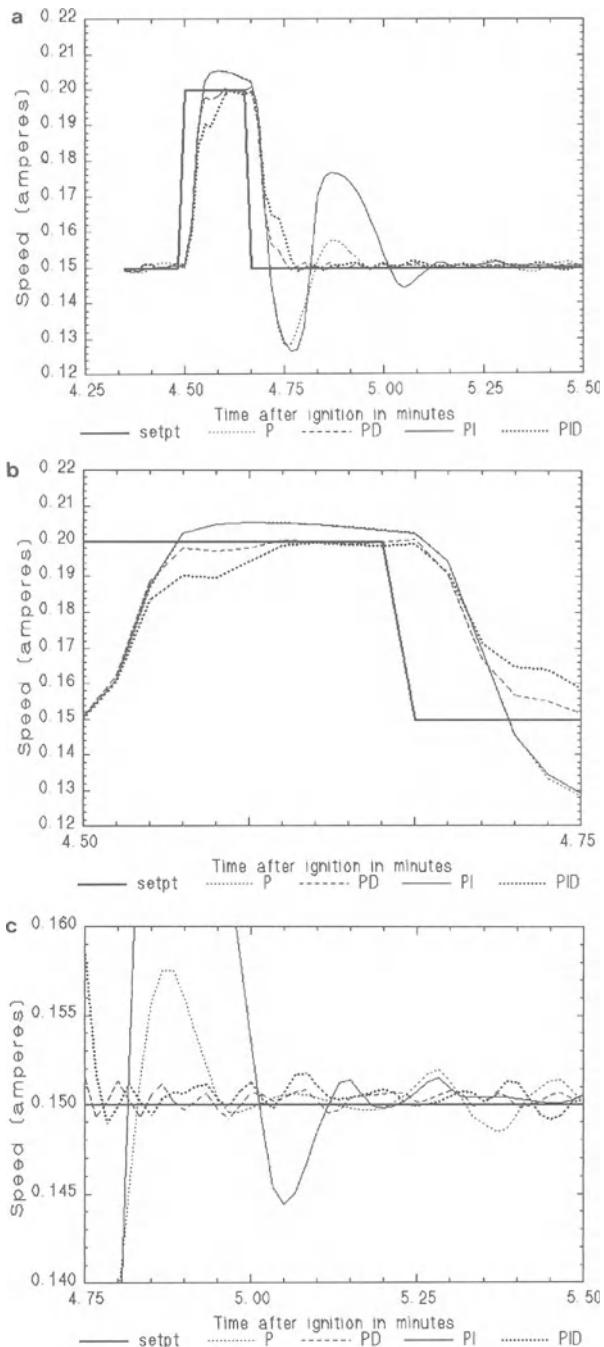


Figure 10.16. Response to speed step-up-down impulse function; (a) full view of response, (b) view of transient response, and (c) view of steady state response.

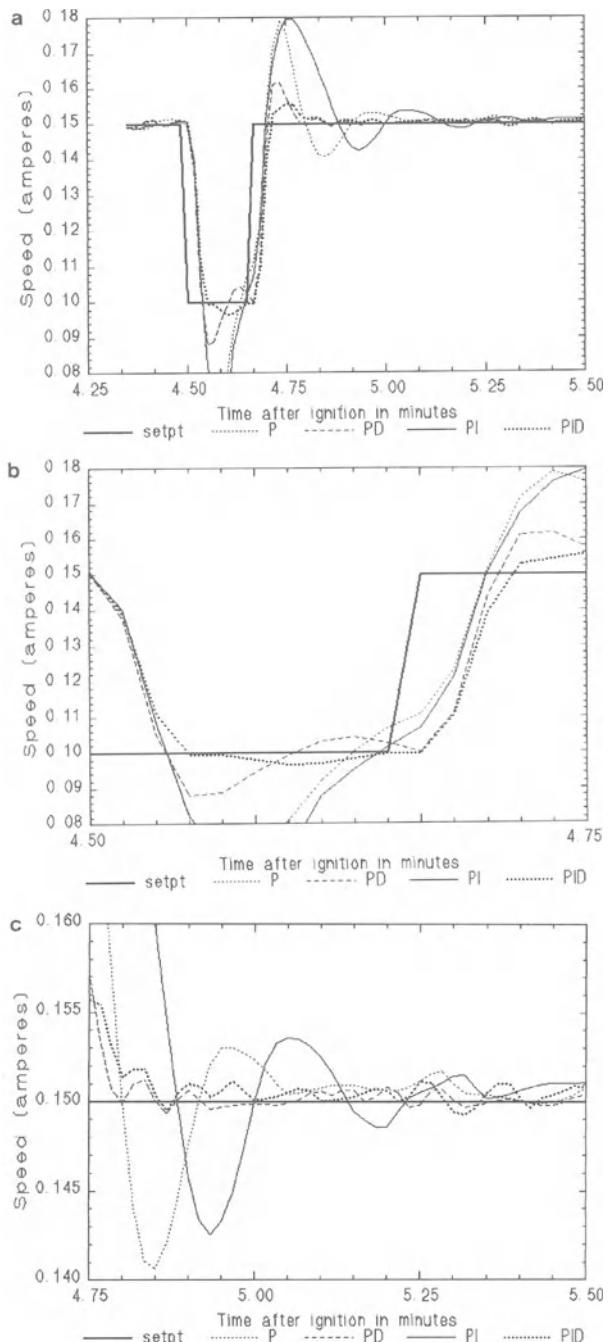


Figure 10.17. Response to speed step-down-up impulse function; (a) full view of response, (b) view of transient response, and (c) view of steady state response.

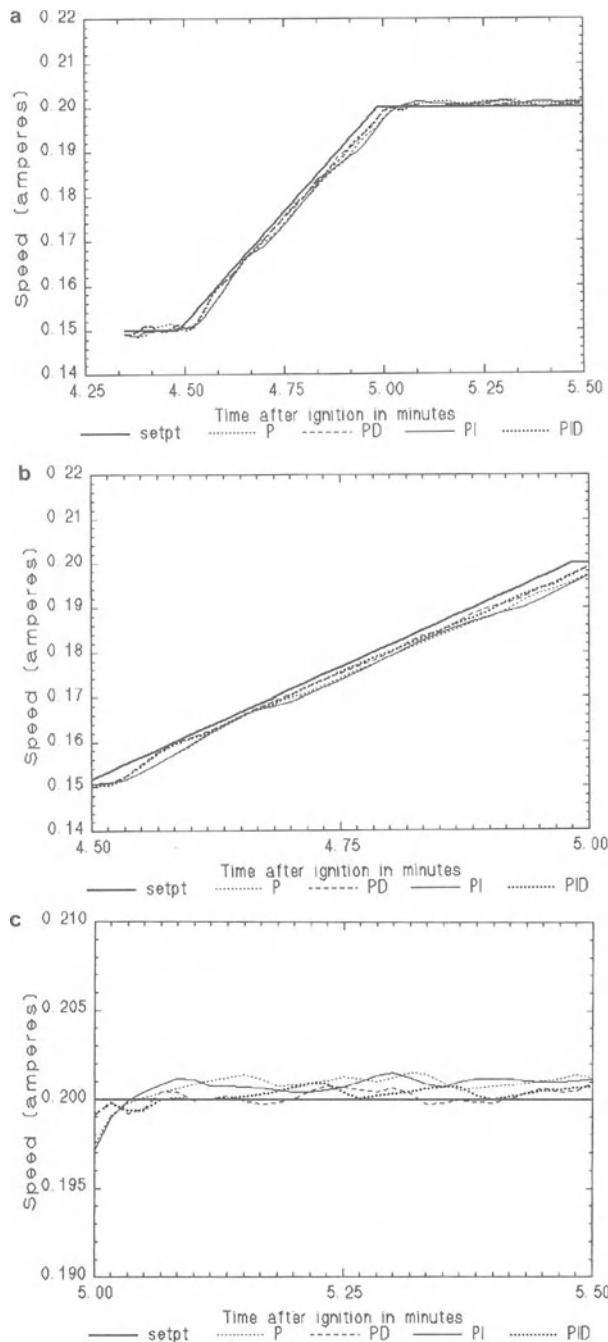


Figure 10.18. Response to speed ramp-up function; (a) full view of response, (b) view of transient response, and (c) view of steady state response.

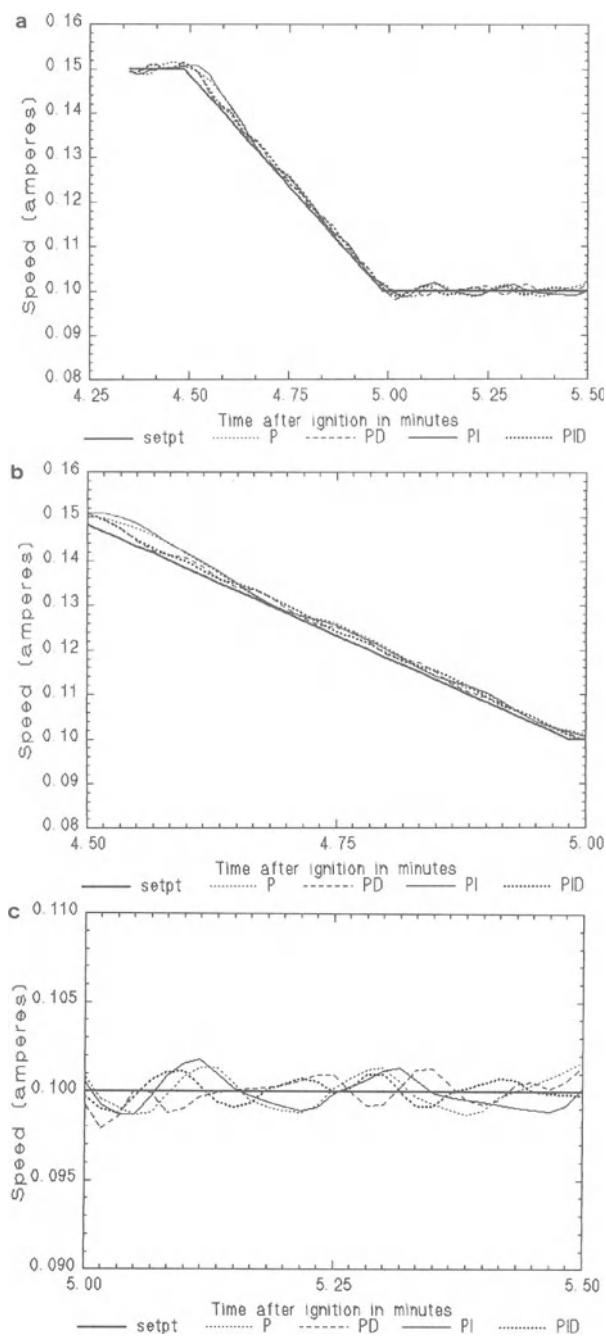


Figure 10.19. Response to speed ramp-down function; (a) full view of response, (b) view of transient response, and (c) view of steady state response.

10.2.14. Speed Ramp-down Test Function—(See Figure 10.19)

- Responsiveness to ramp (average absolute error in mA in following the ramp.): Best—PID (1.81), second best—PD (2.10), third best—P (3.02), worst—PI (3.57)
- Transient stability (maximum percent overshoot): Best—P (4.8), PD (4.9), and PID (4.8), worst—PI (9.2)
- Steady state accuracy and stability (average absolute error in mA from 5:10–5:30): Best—PID (0.311), second best—PD (0.515), third best—PI (0.708), worst—P (0.808)

Obviously, it is difficult to find consistencies in all of this information. We compare the various other designs to the fuzzy P design as well as we can. One of the few consistencies is that the fuzzy PI control design tends to be the most responsive in nearly every case; however in most cases, the differences in responsiveness are not great. Any advantage this design has in this area is probably more than outweighed by its inconsistency in other areas. Its stability is not always the worst, but when it is, it is sometimes much worse than all other designs. It also shows inconsistent performance under steady state conditions.

The fuzzy PD control design is consistently somewhat slow to respond, but the difference does not seem to be major. It compensates by being consistently good if not always the best in the area of transient stability, for both pressure and speed. It was also very good in its steady state performance with respect to speed. The one major failing is that this fuzzy PD design performed surprisingly poorly in its steady state performance with respect to pressure; it tended to continue to oscillate badly, and in fact if we very closely examine some cases (such as the pressure step-down test function used here), it is critically unstable, since the oscillation amplitude grows, though only very slightly. Even in light of what we said earlier about the auto-tuning mechanism making compromises in a mindless way, this behavior is difficult to explain.

Interestingly, the greatest strength of the fuzzy PID design is that its relative performance is consistently excellent on just this point of steady state performance relative to pressure. Although it is not always the best in the various other areas, it is consistently good overall on almost all points. Before stating, in the next chapter, any precisely summarized conclusions, it may be worthwhile to consider a few other aspects of behavior.

10.3. Further Examination of the Auto-Tuning Process

The investigation based on the test functions presented in the previous two sections has been quite informative, but it is also useful to look back again on just what happened during the auto-tuning process. Genetic algorithms are stochastic

in nature, but 30 or more individuals per generation makes for reasonably large sample sizes. Therefore it may be appropriate to look to genetic methods not just for the near optimal solutions they provide but also for other insights into the nature of the problem space. The results of even the 0th generation (the original population, generated randomly) can be interesting.

Specifically, in cases where a genetic algorithm is used to optimize a design, the difference in the best-of-generation scores between early and later generations can be taken as a rough measure of the sensitivity of the design to parameter changes. For example if improvements in the best-of-generation performance score are marginal after the 0th generation, then this indicates that the genetic based auto-tuning mechanism is relatively unimportant and the overall design structure is relatively insensitive to the design details as expressed by the parameters; this is one aspect of robustness.

Admittedly, this is a slight simplification. To speak of the best-of-generation for even the 0th generation is already taking a first step toward optimization because this is essentially a small random search. Therefore comparing the best performance scores between early and later generations is not the same as comparing the average design to the best; it is more in the nature of comparing a good design to the near best. We could speak of generational average scores, but there may be an advantage in focusing on the best-of-generation.

Let us hypothesize the heuristic that a fuzzy control designer of moderate ability using the classical approach (without an auto-tuning mechanism) can select the parameters for the design about as well as a small scale, model-based search of the parameter space. This is a vaguely stated, untested premise, but one could find some way to state it more concretely if necessary. Furthermore, it would be quite possible to test this premise given the right sort of circumstances, such as while teaching a semester-long introductory course on fuzzy control to several students. In any case, even without stating this hypothesis in a crisp way or testing it, it is quite clear that the type of analysis we propose in this section can be taken as at least a very rough guide for considering relative sensitivity to parameter selection.

Figure 10.20a illustrates the best-of-generation ITAE scores for the genetic based auto-tuning of the four steam-engine fuzzy control designs. The plots are shown up to 95 generations simply because that was the extent to which data were collected. (This was about a 24-hour tuning session in the case of the fuzzy PID design.) Note: Ranges of scores differ considerably for fuzzy P and PI, and for PD and PID; therefore it is helpful to plot these separately as seen in Fig. 10.20b and c. The extreme data are follows:

Generation	Fuzzy P	Fuzzy PD	Fuzzy PI	Fuzzy PID
0	74.07	29.50	81.50	30.18
95	53.91	20.37	44.72	20.04

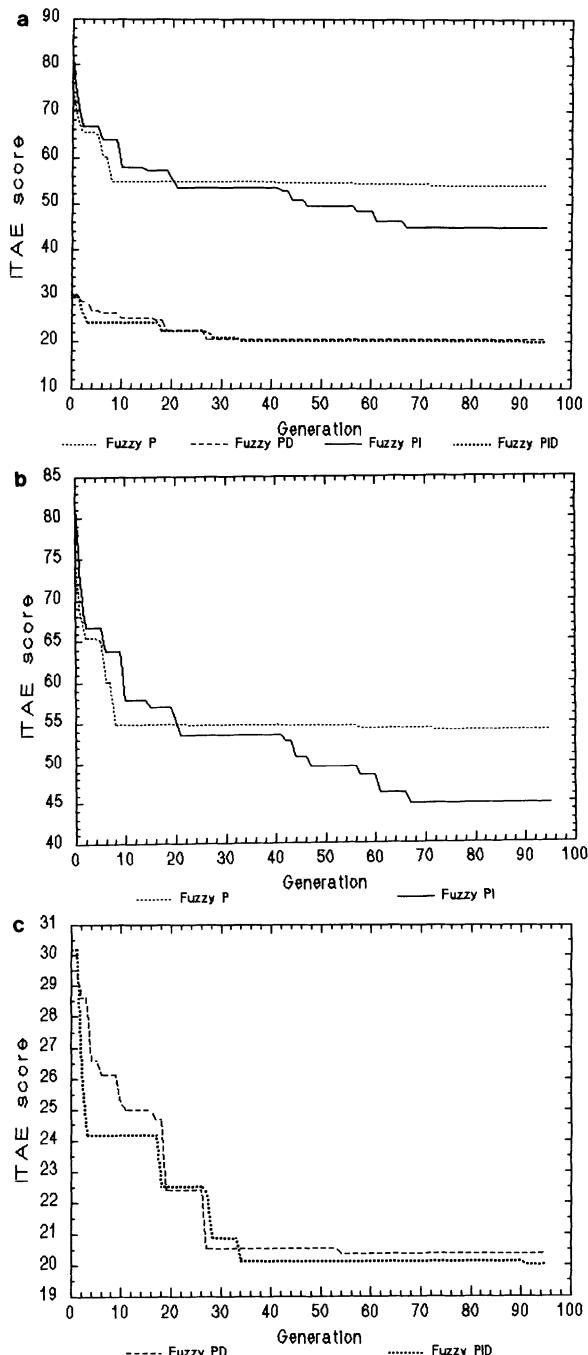


Figure 10.20. Best-of-generation ITAE scores by generation under GA-based auto-tuning; (a) full view; (b) close-up high range view, and (c) close-up low range view.

The most obvious points observed in these data and plots are

- The designs with derivative terms scored much better than those without.
- The genetic based tuning mechanism was more of a benefit for the designs without derivative terms, but these designs were so much worse to begin with that the near-optimal designs were still much worse than the corresponding designs with derivative terms.
- The designs with integral terms actually scored slightly worse at first than the corresponding designs without integral terms.
- However, the designs with integral terms also benefitted considerably more from genetic tuning than the designs without, so that in the end, they score better.

The first two points are dramatic, while the last two are not quite so extreme. Particularly when comparing fuzzy PD and fuzzy PID, the differences are so small, that it is appropriate to question whether they are statistically significant; although we have not made a complete study on this point, we have some evidence that they are. In the process of developing the genetic based tuning mechanism, we experimented several times before making the actual runs; each time results were similar. For example on each test run, the fuzzy PD score was slightly better than fuzzy PID at the 0th generation but fell behind fuzzy PID after a few dozen generations.

CHAPTER 11

Conclusions and Prospects

Throughout this book we considered how a conceptual, and in some ways unorthodox, study of conventional control theory, AI, and fuzzy set theory leads to an unusually clear understanding of the basic concepts of fuzzy control. We also saw that further study in applied neural and genetics methods can lead to important refinements in the fuzzy control-design process. In Part II we provided a detailed example of how all of these concepts can work together in a practical design process, and at the same time we provided evidence concerning the general question of how important dynamic compensation is in fuzzy control. All along we provided a clear rationale for fuzzy control in general and for our own approach in particular.

The first half of Chapter 11 summarizes what we have learned about the specific issue of dynamic compensation in fuzzy control. The second half considers the overall nature of fuzzy control and some prospects for further applications and investigations.

11.1. Summary of the Nature of a Derivative Term in Fuzzy Control Designs

We briefly summarize what the present research indicates about the overall nature of a derivative term in fuzzy control design. As we already discussed it is difficult to separate the performance factors of stability, responsiveness, and accuracy or even to separate variables in the multivariate example used, but we can summarize the overall effectiveness and a few of the factors that may have bearing.

We restate what we found for the derivative term in general with respect to the following questions:

- What is its overall effectiveness in assisting in the simultaneous attainment of high levels of stability, responsiveness, and accuracy for both pressure and speed in this multivariate sample system?
- How sensitive was this effectiveness to design details as expressed by the values of the design parameters?
- How much reason is there to believe that this effectiveness is sensitive to the discrete time cycle used in the control design, more specifically the

relation of this elemental time unit to the response time of the system (plant, actuators, and sensors)?

In the next section we summarize the integral term according to the same three points.

As seen in Section 10.2, fuzzy PD control was much superior to fuzzy P control for almost every performance aspect and for almost every test function tried. Fuzzy PID control was also quite superior to fuzzy PI in most aspects. It was sometimes true in the case of step functions that the design without a derivative term reached the setpoint level slightly more quickly, but this was at the expense of much larger overshoots. Although both quantitative and general comparisons cannot be made, as an overall qualitative evaluation it is clear that the derivative terms result in more effective performance. It is important to keep in mind that this comparison is based on the auto-tuning mechanism, which presumably results in near optimal designs.

The informal analysis in Section 10.3 suggests that furthermore the derivative term decreases the sensitivity to the values in the design parameters. The auto-tuning mechanism in this case clearly did less to improve the designs with derivative terms than similar designs without. If the designs with derivative terms are better overall under near optimal tuning, then the difference is even more significant in the case of less-than-optimal tuning.

Concerning the last question, we found that the derivative term was very effective in both the pressure and the speed controls. This does not constitute a detailed exploration of the question, but it does confirm what we would expect to find. If discrete control-cycle time of the fuzzy control device is relatively long compared to system-response time, then the derivative term can be expected to be especially useful because it is then particularly important to have some way of estimating where the error value will go next. However even where the discrete cycle time is somewhat shorter or the system-response time is somewhat longer, it is clear that the derivative term still contains information that is very useful for its near term anticipatory effects.

According to evidence here, the derivative term results in designs that are much superior in overall performance, less sensitive to how carefully the design is tuned, and better to some degree regardless of the relative length of the control-cycle time and system-response time. The implications, already somewhat obvious, are taken up again in Section 11.3.

11.2. Summary of the Nature of an Integral Term in Fuzzy Control Designs

Summarizing the characteristics of the integral term according to the same questions considered in the previous section, we start again with overall design

effectiveness. For the test plant developed in this research, there seems to be an overall tradeoff in the performance comparison of the fuzzy P and fuzzy PI designs. The fuzzy PI was generally, more responsive, but sometimes the percentage overshoot was worse. Sometimes it behaved more stably as the system settled toward a steady state but sometimes less so. It seemed to respond more accurately to the ramp-type test functions, but the differences were not major. It is probably appropriate to give a slight edge to the fuzzy PI design overall, partly because the ITAE score finally attained for it was considerably better than for the fuzzy P design, but clearly there is some ambiguity in this conclusion.

When comparing the fuzzy PD and fuzzy PID designs, the overall evaluation is considerably less ambiguous. The fuzzy PID design was quite consistently more responsive, though sometimes it had a slightly greater overshoot. In general it was much more accurate and stable as the system settled back toward steady-state conditions. Overall the fuzzy PID design is quite clearly superior in its performance when compared to the fuzzy PD design, though the differences are not nearly so significant as in the comparisons of designs with and without the derivative term.

According to the rough method of comparison discussed in Section 10.3, adding an integral term tends to make the performance more, not less, sensitive to the values of the parameters and details of the design. This is true for both the fuzzy PI and fuzzy PID designs, but particularly for fuzzy PI. This factor is significant enough, so that when the designs are less than well tuned, adding the integral term may result in an inferior performance.

The effects of dynamic compensation are in one sense more difficult to generalize for fuzzy control methods than for conventional control methods, because the fuzzy control device functions on a fixed, discrete time cycle—a significant factor for the integral term. In conventional control design—analog in overall nature, with time viewed continuously—there is good reason to believe that an integral of the error over some interval will always be a useful extra piece of information to have in addition to the most recent instantaneous error value. If system response time is very quick, then the interval (between the limits) of this definite integral may be quite short or the integral gain may be small; however the integral term still seems to be useful for some parameter settings. Because fuzzy control designs are based on discrete time cycles, the derivative and integral terms must be approximated, and there are some potential limitations associated with this, the most serious being that it is no longer possible to the integral term interval arbitrarily short.

This can be a serious problem if the control-cycle time is quite long in comparison to the overall response time of the system (resulting from time lags in the actuators, plant, and sensors). If the ratio of the control-cycle time to the system-response time (which we have not precisely defined, but we speak of it here in vague terms) becomes large enough, then the approximated integral term fails to

be useful; it functions as a distraction if it has a nonzero gain and causes more harm than good.

This seems to be precisely what occurs for the speed control in the plant and control devices as designed in this research. Though this is not a complete experiment, it does seem to confirm the existence of this problem.

In conclusion an integral term can be quite effective in improving overall performance, especially if combined with a derivative term. However an integral term tends to make the performance much more sensitive to design tuning; therefore this performance superiority may depend on precision tuning. The benefit from an integral term also seems to depend on the relation between the discrete control-cycle time and the overall system-response time, so much so that in some circumstances, there may be no potential superiority no matter how careful the tuning. Some implications of these results are considered in the next section.

11.3. Some Practical Implications of the Dynamic Compensation Results

The analysis and summary in Chapter 10 and previous sections of this chapter are based on the example of only one test plant and one overall design structure. However it is reasonable to believe that these results provide some indication of the general utility of dynamic compensation in fuzzy control designs. If we accept the validity of this statement, we can consider a few practical points concerning general strategies in fuzzy control-device design.

First a derivative term can be expected to be useful in nearly every circumstance. As discussed in Section 11.1, the derivative term results in significantly better tradeoffs between the various performance criteria and multiple variables; there is no reason to believe that there is any ratio of cycle/response times for which it would fail to be effective. All of this is true for the optimized or auto-tuned design; the relative insensitivity to parameter values means that it is all the more true where the design is not tuned perfectly.

Fuzzy PD control is almost certainly more effective than fuzzy P control for any given application, and this is all the more true for applications based on the classical approach, where the heuristic knowledge is primarily derived from human intuition. Thus there is a good reason why PD structure has been well favored by fuzzy control researchers since the earliest applications. Although not quite so obvious, our research also indicates that fuzzy PID control is more effective than fuzzy PI control in most cases. Although a derivative term obviously adds somewhat to the design complexity, in most cases fuzzy PD design is preferable to fuzzy P design, and similarly fuzzy PID design is superior to fuzzy PI design.

The research confirms that an integral term can also be very useful, as summarized in Section 11.2, but there are certain conditions involved, perhaps more

so than in conventional control. First fuzzy control designs to date have been based on a discrete time cycle for the control device, and in cases where the length of this cycle is relatively long in comparison to the system-response time, an integral term may have no value at all. Second, the relative sensitivity of the integral term to design parameters and possibly other details means that even in cases where the integral term would be effective in the optimally tuned design, it may not be effective if the design cannot be perfectly tuned. Given these two conditions, it is not surprising that fuzzy control researchers in previous decades were discouraged from using integral terms in their designs.

It is important to recognize that the relative impact of these conditions is changing due to various research developments. First due to developments in hardware and possibly in software as well, there must be many cases where the discrete control-cycle time can be decreased relative to system-response time. One reason for this is an emphasis by some researchers on special purpose fuzzy inference chips. Furthermore even where general purpose logic chips continue to be used, there are dramatic increases in the performance of such chips for the same price and size. Finally it is not impossible for the inference algorithms to be further optimized. Therefore except for new applications emphasizing very rapid response times, the first condition should be much less binding than in the past.

The second condition, is now much less of a problem as well. Auto-tuning approaches have become the norm, so there is little reason to settle for a design that has not attained a near-optimal level of tuning.

Given these developments, fuzzy PID should be preferred for most new applications, and indeed it now seems to be quite common. Fuzzy PI should now also have greater potential than fuzzy P, but as a practical matter, this is somewhat insignificant because we know that the derivative term is likely to be so useful that what really matters is the comparison between fuzzy PD and fuzzy PID.

Overall there is a general agreement between what our research here suggests to be the most useful forms of dynamic compensation in fuzzy control and what the overall applications research community has in fact been using. We can see good reasons why fuzzy PD would be the best structure to consider in the earlier decades of fuzzy control and why fuzzy PID should be increasingly popular now.

Therefore, one benefit of our inquiry here on dynamic compensation is our confirmation that the overall structural approaches most often used in applications of fuzzy control have been the most appropriate. More significantly we were able to explore reasons behind the utility of those approaches more systematically and analytically than has been done in the past.

11.4. Concerning the Rationale of Fuzzy Control

Fuzzy control, as by far the most widespread application area of fuzzy set theory to date, can provide us with useful clues on the nature of fuzzy set

applications in general. We believe the starting point should be a discussion of rationale. Perhaps the most significant questions are the following:

- What do we hope to gain by using fuzzy set concepts the way that we do in fuzzy control (or in another application area)?
- Is this an appropriate way to apply fuzzy set concepts and how so?

At least in the overall sense, there are straightforward answers to both of these questions for the classical approach to fuzzy control. For some of the nonclassical approaches, the answers exist, but are not so straightforward.

For the classical approach, the cement kiln control discussed by Holmlund and Østergaard continues to be a powerful example of what we can gain by applying fuzzy set concepts to control. To summarize the view considered throughout this discussion, real world control problems usually contain highly nonlinear and multivariate relationships among variables, often to such a degree that engineers can make virtually no progress toward an overall control strategy by relying strictly on the precise models and analytical techniques of conventional control theory. However the intuition of human operators is often sufficient to deal with these problems effectively. If one can capture that intuition, then it may be possible to automate and to analyze control where it would otherwise be impossible. But human intuitive knowledge tends to make use of vague concepts even when dealing with quantifiable phenomena. The only body of theory developed specifically to model this human vagueness is fuzzy set theory, particularly the concepts of linguistic variables and of approximate reasoning based thereon. Thus, the explanation of what we hope to gain by the way we use fuzzy set concepts in the classical approach to fuzzy control is perfectly clear, and based on ironclad arguments.

That it is appropriate to apply fuzzy set concepts in this way is obvious. In the classical approach, a control device is designed strictly on the basis of the knowledge of a skilled operator's or a control engineer thinking intuitively. In expressing this knowledge, one must deal with the problem of expressing its implicit vagueness. This is precisely what fuzzy sets are designed to express. There is no question here of using fuzzy set concepts to express other types of uncertainty than the vagueness of human concepts, nor is there any other sort of hint that fuzzy sets are casually used for purposes outside the intended ones.

There are many different ways of deviating from the classical approach, but here let us consider specifically the nonclassical approach used in this research. That is the approach based on some type of model, and some mechanism for auto-tuning based on that model along with a machine-learning technique. This is quite representative of the major recent trends in fuzzy control methods.

The preceding questions become somewhat more difficult to answer in this case. The first problem is that we find ourselves backpedaling on the matter of fundamental rationale. Whereas we justified fuzzy control methods initially by

saying that intuition can be more powerful than models and rigorously structured analysis, we are now attempting to justify our move beyond the classical approach by saying more or less the opposite. This is not a contradiction in any absolute sense. We can explain in some regards intuition is more powerful, but in others structured analysis is. However we should probably remain at least a bit uncomfortable over this point.

Concerning what we hope to gain by using fuzzy set concepts the way that we do in an auto-tuning approach to fuzzy control, there are at least two possible explanations. It is not entirely clear which of these two is true. Probably they both are, in varying degrees depending on the details of implementation. One explanation, which the term “auto-tuning” is designed to suggest, is that we are trying to begin with as much intuitive knowledge as we can obtain, and only then to use the model and optimization mechanism to fine tune it. Thus it is necessary to use fuzzy set concepts because otherwise it would be impossible even to begin with intuitive knowledge.

The second explanation, which would be suggested by such labels as “optimal fuzzy control” or “fuzzy control design automation,” is that little human intuitive knowledge is used except to design the overall structure; the machine-learning mechanism is responsible for sorting out all details of the design for itself. Although these labels are rarely used in the literature, they seem to be just as accurate as “auto-tuning” in describing the intent of the approach.

To the extent that this second explanation is true, we must think carefully about why we use fuzzy set concepts in this way. We cannot claim that these fuzzy sets (or fuzzy relations) express human vagueness, the most we can claim is that they represent human-like vagueness. In the case of fuzzy modeling techniques in the spirit of Takagi and Sugeno, it is valid to say that this humanlike vagueness may be useful because it allows humans to understand better the system for which the model is constructed. But control designs are usually intended as ends in themselves, so generally this argument would not apply to them. How then do we explain what we are trying to do here? The only justification seems to be that we are attempting to create not just control devices but an entire control-design process in our own image. That is we want the control design to be based on fuzzy set concepts of human-like vagueness because our human vagueness is a good thing and we think that not only machines but also machine design processes would benefit by being more like us.

We have attempted to answer the first question (as it relates to the nonclassical, auto-tuning approach) a way that suggests an answer to the second question. Even if auto-tuning is really more a matter of design automation, then fuzzy set concepts are presumably as appropriate a way of representing human-like vagueness as human vagueness. However this discussion already takes us far beyond most discussions in the literature of rationale for specific problems in the application of fuzzy sets. For both the classical and nonclassical approaches to fuzzy control, the

rationale behind the applications is very closely tied to the philosophy of artificial intelligence. These problems deserve much more widespread discussion.

As a general comment on all applications, we believe that it is appropriate to use fuzzy sets, fuzzy relations, and fuzzy logic to model human vagueness, and we explain our reasons throughout this book. Since similar explanations are given in most introductory literature on fuzzy set theory, we need not apologize for using this body of theory to model human vagueness as part of a practical application. However we remain agnostic on the question of whether it is appropriate to use fuzzy sets and fuzzy relations to represent forms of uncertainty other than human vagueness. It may be appropriate to use these methods without modification to represent stochastic phenomena related more to natural randomness than human vagueness, but if so why? This question is frequently ignored in the literature on applications.

11.5. Rational Approach to Research in Fuzzy Control and Other Applications of Fuzzy Set Theory

Though we have no formal training in the philosophy of science or engineering, it seems clear that there are two necessary conditions for a given approach to be applied science. First it must work or at least there must be some reason to believe that it could work in the future. Second, there must be a clear and rational understanding of why it works or could work. If an approach is impractical, and it has no hope of leading to anything practical, then it may be intellectually interesting, but it is not applications research. On the other hand if a given approach seems to work, but there is no clear explanation for why it does, then it is akin to folklore or mysticism.

The field of fuzzy control has developed well in some directions but until now has largely failed to develop in others. Provided that each specific approach developed is both practical and based on a rational application of available tools, then the field as a whole is on solid ground. Alternative views of fuzzy control mechanisms should be encouraged; this includes comparisons with related fields. Researchers should also continue to compare different approaches from the perspective of practical results. Since control applications involve complex behaviors, relative utility is difficult to evaluate objectively. Despite this difficulty, many researchers have realized that this is important and have made comparative studies in a manner that is as objective as possible. The comparison we undertake in Part II is an example of the type of study.

What is true of fuzzy control research is true of other potential applications of fuzzy set theory as well. Both aspects of justification should be viewed as essential.

11.6. Prospects for Further Applications and Research

We hope this book opens many doors to fruitful applications and further research into several questions about fuzzy control. Our manner of presentation makes it easier for a nonspecialist to understand conceptual and analytical aspects of fuzzy control, thereby placing the technology within easy reach of more people. We hope this increases the number of practical applications, including industrial automation, consumer or commercial products, agriculture, and general robotics. Potential new applications, some simple, some very complex, abound almost everywhere we look.

For those interested in general research we remark that rather than specific practical applications, a clear grasp of the conceptual aspects of any field always leads to further awareness of what research needs to be done. Furthermore this book presents many ideas about specific issues worthy of further investigation. These range from overall general conceptual directions to specific, pragmatic studies which could be made. Let us now review of few of these.

We are fascinated by how best to present and interpret mechanisms of approximate reasoning based on linguistic variables and fuzzy logic, as demonstrated by the alternative interpretation in Chapter 5. However that interpretation is presented applies to the classical fuzzy logic operators and the standard interpretation of fuzzy inference; how well would it function with other operators? A related question of perhaps greater practical significance is how well would it function with other types of production-rule systems. Production rule systems we are accustomed to dealing with in fuzzy control applications are of a relatively simple type. The rule bases of most fuzzy control systems do not directly require rule chaining, nor do they generally associate uncertainty with the rules themselves. In both these ways they are simpler than the general expert system rule base. Would an alternative interpretation shed light on this more generalized type of production-rule system as well?

On a still more practical level, many aspects of the auto-tuning concept bear further investigation, but some are particularly interesting. We discussed some reasons why the ITAE approach is more appropriate than an IAE approach for an evaluation index to the auto-tuning mechanism. It would be simple and meaningful to compare evaluation functions produced by IAE, ITAE, ISE, and ITSE approaches for a given sample application. A related and more conceptually challenging problem is to attempt to find a generally applicable approach to searching the auto-tuning problem space in a way that is more efficient than the genetic approach.

Similarly it would be useful to investigate other approaches to modeling the plant, including fuzzy models. There is a tendency to go to extremes when developing these models—some researchers tend toward extremely mathematical methods for developing such models while others construct models on the basis of intuitive knowledge. For the field of fuzzy modeling to become widely understood

and practically applied, the middle ground between these extremes must be better developed.

To study more fully the utility of the various forms of dynamic compensation in fuzzy control requires investigations based on other test plant applications. Furthermore several specific questions should be studied in the process, for example factors that make an integral term more or less useful in a given type of application. Finally, as a very general direction, our presentation of fuzzy control, which emphasizes its relation to both conventional control theory and the broad field of AI suggests that by studying the literature of all three fields, more useful approaches can be developed.

APPENDIX A

Concerning Definitions of Artificial Intelligence

The ideal definitions of AI satisfy the following criteria:

- It would be precise, concrete, and objective
- It would be strictly functional rather than structural in nature
- It would be meaningful in the practical sense as it relates to current research and the current state of applications and consistent with the way the term is typically used in the literature

If such an ideal definition existed, it could be used in all books and papers, so there would be no reason to consider any other perspectives. The meaning of the first criterion is quite obvious. We seek a definition that is not vague—not based on subjective interpretation. This already makes us somewhat pessimistic as to the existence of an ideal definition. In Chapter 3 we noted that we have found no evidence that philosophers can agree on a rigorously stated and universally acceptable way of determining which forms of human behavior are intelligent and which are not. It would therefore be quite remarkable if somehow a definition of artificial intelligence could be superior in this sense to the definitions of natural intelligence.

According to the second criterion, whether or not a given application of computing devices is AI would have nothing whatsoever to do with the structural characteristics of the application design; we would not be permitted for example to say anything at all about heuristics as opposed to algorithms. It is desirable for the definition to be strictly functional because in everyday use the word intelligent normally describes types of behavior or performance, not mechanisms for attaining those behaviors or levels of performance. It would seem then appealing that artificial intelligence should be defined in a similar manner.

The third criterion requires the definition to be in at least basic agreement with what most people who work in the computing field mean when they use the term “artificial intelligence” as part of their work. For the moment let us limit ourselves to a discussion of what we call extreme discrepancies.

First, we note that aside from a few skeptics, most people in the field consider AI to be a current reality. The large number of applications-oriented books and papers with something about artificial intelligence in their titles are testaments to this point. Therefore, any proposed definition of AI that is so strict that according

to it no current applications would be classified as AI is an unrealistic definition in the practical sense that it plainly does not agree with how the term is widely understood and used. Let us refer to such an unrealistic definition as having a Type 1 extreme discrepancy. On the other hand, most computer users obviously do not consider all applications of computers to be AI. Few people consider mundane accounting programs or similar types of ordinary data processing to be examples of artificial intelligence. This means that any definition so broad that it classifies all uses of computational devices as AI would also be unrealistic. We refer to this type of definition as having a Type 2 extreme discrepancy. Such a definition would have no practical significance whatsoever because we might just as well say “computer application” as say “artificial intelligence”.

It seems to us that few people would argue with these criteria for an ideal definition, but this does not settle the problem. One question is, does such a definition exist? A second is, if not, then what should be done about the situation?

We suspect that the answer to the first question is that no, unfortunately such an ideal definition does not exist. To prove that it does not would be proving a negative, and we will not attempt that here, but we can respond to what some people have proposed to be ideal definitions of artificial intelligence. As an extreme case, let us start with the idea, sometimes still stated, that the *Turing test* or some variation of it is an ideal definition.

Along with the Turing machine, the Turing test is one of the two things for which British mathematician and heroic wartime cryptanalyst, Alan Turing, is perhaps most widely remembered. The Turing machine is another matter, but as for the Turing test this seems somewhat ironic in the sense that there is little reason to believe that Turing himself considered this to be a central theme in his own work. In any case, Turing's own description of his well known test is clearly described along with many other points in his 1950 essay, “Can a Machine Think?”

Turing said that a way of settling the philosophical question of whether a machine is capable of thinking would be to subject it to the following test. Prepare two rooms so that their only mode of communication is by teletype line. In one room situate a human subject and the machine; in the second room, a human interrogator. The interrogator knows that there are two entities X and Y, in the other room and one is a human, the other a machine. Via teletype the interrogator can address questions to X and Y and receive their responses via teletype. After asking several questions of X and Y, the interrogator's goal is to attempt to determine whether X is the human and Y is the machine's or *vice versa*. The machine is instructed or designed so that its goal is to deceive the interrogator; the goal of the human subject is to help the interrogator reach the correct conclusion presumably by answering the questions as well as possible and hoping that those answers were more humanlike than the machine's answers. Turing said that if the machine were capable of deceiving the interrogator during approximately half of the times this

test were conducted, then this would be a solid case for saying that the machine can think.

We make a few points that Turing makes very clear in this essay about his own intentions. First, in the form of the test that Turing describes, he intended that the interrogator would be allowed to ask any sort of questions whatsoever, thus neither the machine nor the designer of the machine would have any control over which types of questions it would be expected to answer. There would be absolutely no restrictions on the subject matter, and if the machine hoped to succeed, then it would somehow have to be able to give a humanlike response to questions of all types. To date no one has produced a machine capable of doing this, and most researchers would likely agree that no such machine will exist anytime soon. Thus if the Turing test, precisely as Turing described it is used as a test for artificial intelligence, then AI does not yet exist. This definition therefore exhibits Type 1 of extreme discrepancy so it violates our third criterion for an ideal definition.

Second, Turing makes no claim that his proposed test is the only reasonable one for answering the question of whether a machine can think. He clearly presented his test as a sufficient condition for establishing that a given machine thinks, but not as a necessary condition. For that matter Turing does not even actually claim much reasonableness for his own test. He notes for example that in some ways the test is unreasonably strict and unfair to the machine. For one thing, Turing recognizes that if the roles of machine and human subject were reversed so that the human subject tried to deceive the interrogator and the machine tried to help the interrogator reach the correct conclusion, then the human subject would always lose. This is because the interrogator could simply pose an arithmetic problem and whoever took longer to answer correctly would be the human subject. Turing reasons that a test so strict that the human subject was bound to fail if the roles were reversed is unfair.

Finally, Turing never explicitly uses the term, artificial intelligence, in his essay, which is to be expected because the term was apparently not coined until a few years after his death. Some will say that this is a trivial point in the sense that “a machine that thinks” obviously should mean the same as “artificial intelligence”. But it is worth noting that Turing was writing at a time before there was an established field known by a particular label; that is, long before the term AI had acquired a widespread technical meaning corresponding to certain types of computer applications. It seems reasonable to assume that Turing was simply addressing a philosophical question, not attempting to anticipate the precise boundaries of a technical field that was then emerging or would soon emerge. One can argue that as a philosophical point, it is not what a word does mean, but rather what it should mean. One could then say that Turing was right, and that all of those engineers and programmers out there in the 1980’s and 1990’s who use the term AI to describe their work are wrong. But we wonder whether Turing himself, if he were still alive, would have proceeded so far with such an esoteric line of argumentation.

We see two points in this. First, we believe that Turing did not intend for his test to be used as the sole basis for a definition of a future technical field. The second point is that if one does insist on using the Turing test, in the form that Turing stated it, as the sole basis for defining AI, then the field, at least in the applied sense, does not yet exist.

Short of coming up with an amazing breakthrough that does create a computer capable of passing the Turing test, this last problem leaves us with only a few alternatives:

- We discard the notion that there is a field called artificial intelligence, and we say that all of those people claiming to work in that field are deluding themselves
- We discard the notion that the Turing test should be used as the sole basis for defining the field known as AI
- We sidestep the issue by saying that what we really mean is not the Turing test itself but some variation on the idea of the Turing test

We prefer the second alternative; extreme AI skeptics probably prefer the first alternative. However this appendix is addressed to those who choose the third alternative.

One example of a variation on the idea of the Turing test is to say that a computer application is AI if and only if its performance at a given task is indistinguishable from that of a human attempting the same task. We refer to this statement as the given task variation on the Turing test. This statement differs from the actual Turing test because to pass that test consistently, the machine must be capable of making humanlike responses not to a given task but to all questions the interrogator could conceivably throw at it.

Next we consider whether given task variation meets the three criteria of an ideal AI definition. It meets the second criterion because it is a strictly functional definition. As for the first and third criteria, careful consideration reveals that depending on exactly how it is interpreted, the given task variation can possibly meet one or the other of these, but certainly not both.

The problem is, what exactly does one mean by “given task”? If we interpret this quite literally, then the definition would satisfy the first criterion, but not the third one. This is because it contains a Type 2 extreme discrepancy. For example summing five integers is surely an example of “a given task”. We can write a very simple program to do this calculation, and if desirable we could add a feature to make the program wait a suitable, random number of seconds before giving the answer, so that it would seem more humanlike in its slowness. Is this AI? Certainly it would be if the given-task variation is to be interpreted literally and this is accepted as the definition of AI. But if such an interpretation is accepted, then every computer application essentially becomes AI.

Because a literal interpretation of a “given task” has a serious problem, one may wish to consider other possible interpretations. One might say for example that what we really mean is a given task of a type that would be said to require intelligence in the case that a human is required to perform it. But first let us point out that this takes us now so far from the original Turing test that there seems little reason to bring Turing into the question anymore. We may just as well say, as some do say, that AI is any application of a computing device designed for the performance of a task that is associated with intelligence when performed by humans. However, this latter interpretation is still not an ideal definition. It fails almost completely on the first criterion. Whether consulting philosophers, psychologists, or the person in the street, it is unlikely that we could can arrive at a precise and objective definition of which tasks require intelligence when performed by human beings. Furthermore this definition can, at best, only be forced with difficulty to satisfy the third criterion.

Consider, for example, taking a survey in which we asked the question, “Which of the following tasks more clearly requires intelligence to perform by a human: playing a decent game of Chinese checkers, or inverting a (square, non-singular) matrix?” Assume that the survey was taken of people with only a limited knowledge of mathematics or computing. We believe that in such a case the typical response would be that certainly inverting a matrix (a very mysterious process in the minds of some people as one soon finds if one ever tries to teach it to people who are not mathematics, engineering, or physics majors) is a much more intelligent activity than playing Chinese checkers (which any child can learn to do quite quickly). Therefore on this basis, the task constituting artificial intelligence if performed in a computer, is matrix-inversion, not Chinese checker-playing. However, this is the opposite of the correct answer in the eyes of most computer scientists.

Incidentally, obviously our own preferred definition, based on the distinction between applications that use heuristics, and applications that are solely algorithmic; would be much more accurate here. Inverting a matrix is a strictly algorithmic task. Playing Chinese checkers is very definitely based partly on heuristics. Even when the mini-max game-tree search algorithm is used, it must nearly always be used in conjunction with an evaluation function; an evaluation function is of course an excellent example of a heuristic.

This last interpretation of the given task variation is somewhat similar to the well known definition of AI by Avron Barr and Edward Feigenbaum. The Barr-Feigenbaum definition (see Barr and Feigenbaum (1981), 1:3) considers “systems that exhibit the characteristics we associate with intelligence in human behavior.” However, although we have not studied the matter in detail, it seems that characteristics associated with human intelligence is a somewhat easier concept to define clearly than tasks associated with human intelligence. In this sense the Barr-Feigenbaum definition comes close to being an ideal definition, but even it does not satisfy the first criterion perfectly. Also the use of characteristics in this

definition is somewhat vague. If interpreted in the most obvious way, then this definition is at least to some degree structural, so it does not satisfy the second criterion perfectly either.

Thus we did not find an ideal definition of AI in the Turing test nor in any other of the definitions we have seen. There is absolutely nothing presumptuous in this statement, and it is by no means a slight against Turing or anyone else. There are very many concepts and several fields of study for which it is apparently impossible to present a precise definition suitable in all contexts, yet these concepts and fields are nonetheless useful. To be convinced of this, look at a high school biology textbook under the definitions of life. If all concepts could be defined precisely, then why would we bother with fuzzy set theory?

It seems common in much of science to make two allowances when considering definitions of concepts that are difficult to define. The first allowance is to permit some compromises. Compromises include definitions the interpretations of which are unavoidably subjective, definitions that are really just lists of characteristics that generally apply but not always, definitions that are somewhat structural in nature when a strictly functional definition would seem more appropriate or *vice versa*, definitions based on lists of examples, and so on. The second allowance is to accept that sometimes different definitions of the same concept are appropriate in different contexts. In Chapter 3 we emphasize heuristics as a key characteristic of AI, but we do not believe this is the only conceivable definition of AI—merely a practical way of viewing the field based on our experience and study and *also* the most relevant way of understanding the remainder of this book. Furthermore, we do not believe that our view is particularly unique. Several books, papers, and articles now emphasize structural rather than functional aspects when attempting to define AI. This includes such esteemed sources as *Encyclopedia Britannica*'s 1985 definition, by Bruce G. Buchanan: “Artificial intelligence is the branch of computer science that deals with ways of representing knowledge using symbols rather than numbers and with rules-of-thumb, or heuristic, methods for processing information.” What is unique about our presentation, is our emphasis on heuristics rather than symbolic processing. (We explain our reasons in Chapter 3.)

APPENDIX B

Sample Pascal Program Illustrating Fuzzy Inference Algorithms by a Simple Example with Three Inputs and Two Outputs

```
program Fuzzy_inference ;
{H. Lewis, Fukushima Univ., 6 Aug 94, revised 28 Feb
95}
{Simple example of fuzzy inference with 3 inputs, 2
outputs.}

uses crt, maxmin ;

const
  m = 3 ;  n = 2 ;
  ninf = -1E20 ;
  inf = 1E20 ;

type
  terms = (NB, NM, NS, ZO, PS, PM, PB) ;
  TFN_type = array[terms, 1..3] of real ;
  R_type = record a : TFN_type end ;
  Rx_type = array[1..m] of R_type ;
  Ry_type = array[1..n] of R_type ;
  mu_type = array[terms] of real ;
  prime_type = record mu : mu_type end ;
  f_type = array[terms, terms, terms] of terms ;
  base_defin_terms = (min_value, max_value, interval) ;
  base_defin_type = array[base_defin_terms] of real ;

const
  first_term : terms = NB ;
  last_term : terms = PB ;

var
  x : array[1..m] of real ;
```

```

xp : array[1..m] of prime_type ;
yp : array[1..n] of prime_type ;
y : array[1..n] of real ;
i, j : integer ;

const

Rx : array[1..m] of R_type
= ((a:((ninf,-12,-8), (-12,-8,-4), (-8,-4,0),
(-4,0,4), (0,4,8), (4,8,12), (8,12,inf))),,
(a:((ninf,-12,-8), (-12,-8,-4), (-8,-4,0),
(-4,0,4), (0,4,8), (4,8,12), (8,12,inf))),,
(a:((ninf,-12,-8), (-12,-8,-4), (-8,-4,0),
(-4,0,4), (0,4,8), (4,8,12), (8,12,inf)))) ;

Ry : array[1..n] of R_type
= (a:((ninf,-12,-8), (-12,-8,-4), (-8,-4,0),
(-4,0,4), (0,4,8), (4,8,12), (8,12,inf))),,
(a:((ninf,-12,-8), (-12,-8,-4), (-8,-4,0),
(-4,0,4), (0,4,8), (4,8,12), (8,12,inf)))) ;

f : array[1..n] of f_type
= (((((PB,PB,PB,PB,PB,PB), (PB,PB,PB,PB,PB,PB),
(PB,PB,PB,PB,PB,PM), (PB,PB,PB,PB,PM,PM),
(PB,PB,PB,PB,PM,PS), (PB,PB,PB,PM,PM,PS),
(PB,PB,PM,PM,PS,ZO)),,
((PB,PB,PB,PB,PB,PB), (PB,PB,PB,PB,PB,PM),
(PB,PB,PB,PB,PM,PM), (PB,PB,PB,PB,PM,PS),
(PB,PB,PB,PM,PM,PS), (PB,PB,PM,PM,PS,ZO),
(PB,PM,PM,PS,PS,ZO)),,
((PB,PB,PB,PB,PB,PM), (PB,PB,PB,PB,PM,PM),
(PB,PB,PB,PB,PM,PS), (PB,PB,PB,PM,PM,PS),
(PB,PB,PM,PM,PS,ZO), (PB,PM,PM,PS,PS,ZO,ZO),
(PM,PM,PS,PS,ZO,ZO,PS)),,
((PB,PB,PB,PB,PB,PM,PM), (PB,PB,PB,PB,PM,PM,PS),
(PB,PB,PB,PM,PM,PS), (PB,PB,PB,PM,PM,PS,ZO),
(PB,PM,PM,PS,PS,ZO,ZO), (PM,PM,PS,PS,ZO,ZO,NS),
(PM,PS,PS,ZO,ZO,NS,NS)),,
((PB,PB,PB,PB,PM,PM,PS), (PB,PB,PB,PM,PM,PS,PS),
(PB,PB,PM,PM,PS,PS,ZO), (PB,PM,PM,PS,PS,ZO,ZO),
(PM,PM,PS,PS,ZO,ZO,NS), (PM,PS,PS,ZO,ZO,NS,NS),
(PS,PS,ZO,ZO,NS,NS,NM)),,
((PB,PB,PB,PM,PM,PS), (PB,PB,PM,PM,PS,PS,ZO),
(PB,PM,PM,PS,PS,ZO,ZO), (PM,PM,PS,PS,ZO,ZO,NS),
(PM,PS,PS,ZO,ZO,NS,NS), (PS,PS,ZO,ZO,NS,NS,NM)),
((PB,PB,PM,PM,PS,PS), (PB,PB,PM,PM,PS,PS,ZO),
(PB,PM,PM,PS,PS,ZO,ZO), (PM,PM,PS,PS,ZO,ZO,NS),
(PM,PS,PS,ZO,ZO,NS,NS), (PS,PS,ZO,ZO,NS,NS,NM)),
((PB,PB,PM,PM,PS,PS,ZO), (PB,PM,PM,PS,PS,ZO,ZO),
(PM,PM,PS,PS,ZO,ZO,NS), (PM,PS,PS,ZO,ZO,NS,NS),
(PS,ZO,ZO,NS,NS,NM,NM)));

```

```

(PS,PS,ZO,ZO,NS,NS,NM), (PS,ZO,ZO,NS,NS,NM,NM),
(ZO,ZO,NS,NS,NM,NM,NM))),

(((PB,PB,PB,PB,PB,PM,PM), (PB,PB,PB,PB,PM,PM,PS),
(PB,PB,PB,PM,PM,PS,PS), (PB,PB,PM,PM,PS,PS,ZO),
(PB,PM,PM,PS,PS,ZO,ZO), (PM,PM,PS,PS,ZO,ZO,NS),
(PM,PS,PS,ZO,ZO,NS,NS)),,
((PB,PB,PB,PB,PM,PM,PS), (PB,PB,PB,PM,PM,PS,PS),
(PB,PB,PM,PM,PS,PS,ZO), (PB,PM,PM,PS,PS,ZO,ZO),
(PM,PM,PS,PS,ZO,ZO,NS), (PM,PS,PS,ZO,ZO,NS,NS),
(PS,PS,ZO,ZO,NS,NS,NM)),,
((PB,PB,PB,PM,PM,PS), (PB,PB,PM,PM,PS,PS,ZO),
(PB,PM,PM,PS,PS,ZO,ZO), (PM,PM,PS,PS,ZO,ZO,NS),
(PM,PS,PS,ZO,ZO,NS,NS), (PS,PS,ZO,ZO,NS,NS,NM),
(PS,ZO,ZO,NS,NS,NM,NM)),,
((PB,PB,PM,PM,PS,PS,ZO), (PB,PM,PM,PS,PS,ZO,ZO),
(PM,PM,PS,PS,ZO,ZO,NS), (PM,PS,PS,ZO,ZO,NS,NS),
(PS,PS,ZO,ZO,NS,NS,NM), (PS,ZO,ZO,NS,NS,NM,NM),
(ZO,ZO,NS,NS,NM,NM,NM)),,
((PB,PM,PM,PS,PS,ZO,ZO), (PM,PM,PS,PS,ZO,ZO,NS),
(PM,PS,PS,ZO,ZO,NS,NS), (PS,PS,ZO,ZO,NS,NS,NM),
(PS,ZO,ZO,NS,NS,NM,NM), (ZO,ZO,NS,NS,NM,NM,NM),
(ZO,NS,NS,NM,NM,NM,NB)),,
((PM,PM,PS,PS,ZO,ZO,NS), (PM,PS,PS,ZO,ZO,NS,NS),
(PS,PS,ZO,ZO,NS,NS,NM), (PS,ZO,ZO,NS,NS,NM,NM),
(ZO,ZO,NS,NS,NM,NM,NM), (ZO,NS,NS,NM,NM,NM,NM),
(NS,NS,NM,NM,NM,NB,NB)),,
((PM,PS,PS,ZO,ZO,NS,NS), (PS,PS,ZO,ZO,NS,NS,NM),
(PS,ZO,ZO,NS,NS,NM,NM), (ZO,ZO,NS,NS,NM,NM,NM),
(ZO,NS,NS,NM,NM,NM,NB), (NS,NS,NM,NM,NM,NB,NB),
(NS,NM,NM,NM,NB,NB,NB))) ;

base_defin_y : array[1..n] of base_defin_type
= ((-15,15,0.2),(-15,15,0.2)) ;

procedure Fuzzify(xi:real; Rxi:R_type; var
xpi:prime_type) ;
{Fuzzifies an input.}
var t : terms ;
begin
with Rxi do
  for t := first_term to last_term do
    xp[i].mu[t] := max(0, min( (x[i] -
      a[t,1])/(a[t,2] - a[t,1]),
      (a[t,3] - x[i])/(a[t,3] - a[t,2]) ) );
end ;

```

```

procedure Inference3in(xp1,xp2,xp3:prime_type;
  fj:f_type; var ypj:prime_type) ;
{Fuzzy inference with three inputs.}
var ty : terms ;
  tx : array[1..3] of terms ;
begin
  for ty := first_term to last_term do yp[j].
    mu[ty] := 0 ;
  for tx[1] := first_term to last_term do
    for tx[2] := first_term to last_term do
      for tx[3] := first_term to last_term do
        begin
          ty := fj[tx[1],tx[2],tx[3]] ;
          ypj.mu[ty] := max(yp[j].mu[ty],
            min3(xp[1].mu[tx[1]],
              p[2].mu[tx[2]],{xp}[3].mu[tx[3]]));
        end ;
  end ;

function Defuzzify(ypj:prime_type; Ryj:R_type;
  base_defin_yj:base_defin_type) : real ;
{Defuzzification of an output by center of gravity.}
var yb, muyb, sumybmuyb, summuyb, yjr, yj, dist,
  yjdist : real ;
  t : terms ;
begin
  sumybmuyb := 0 ;  summuyb := 0 ;
  yb := base_defin_yj[min_value] ;
  with Ryj do
    while yb <= base_defin_yj[max_value] do
      begin
        muyb := 0 ;
        for t := first_term to last_term do
          muyb := max(muyb,
            min(yp[j].mu[t],
              max(0,
                min((yb - a[t,1])/(a[t,2] - a[t,1]),
                  (a[t,3] - yb)/(a[t,3] - a[t,2]))))) ;
        sumybmuyb := sumybmuyb + yb*muyb ;
        summuyb := summuyb + muyb ;
        yb := yb + base_defin_yj[interval] ;
      end ;
  yjr := sumybmuyb / summuyb ;
  yjdist := abs((base_defin_yj[min_value] - yjr)) ;
  yj := base_defin_yj[min_value] ;
  yb := base_defin_yj[min_value] +
    base_defin_yj[interval] ;

```

```

while yb <= base_defin_yj[max_value] do
begin
  dist := abs(yb - yjr) ;
  if dist <= yjdist
    then begin yj := yb ; yjdist := dist ; end ;
  yb := yb + base_defin_yj[interval] ;
end ;
Defuzzify := yj ;
end ;

procedure Display_x(i:integer; xi:real; xpi:prime_type);
var t : terms ;
begin
  write ('x[',i,'] = ',x[i]:6:2,' ':3,
        'mu of x'' = ') ;
  for t := first_term to last_term do
    write(xpi.mu[t]:4:2,' ':2) ;
  writeln ;
end ;

procedure Display_y(j:integer; ypj:prime_type; yj:real);
var t : terms ;
begin
  write('mu of y''[',j,'] = ') ;
  for t := first_term to last_term do
    write(ypj.mu[t]:4:2,' ':2) ;
  writeln(' y[',j,'] = ',yj:6:2) ;
end ;

begin {main}
clrscr ;
for i := 1 to m do
begin
  write('Input x[',i,'] : '); readln(x[i]) ;
end ; writeln ;

for i := 1 to m do Fuzzify(x[i],Rx[i], xp[i]) ;
for i := 1 to m do Display_x(i,x[i],xp[i]) ; writeln;

for j := 1 to n do Inference3in(xp[1],xp[2],xp[3],
                                 f[j], ypj[j]) ;

for j := 1 to n do y[j] := Defuzzify(ypj[j],
                                      Ry[j],base_defin_y[j]) ;
for j := 1 to n do Display_y(j,ypj[j],y[j]) ;
readln ;
end.

```

APPENDIX C

Symbols and Acronyms

Notation used in the literature on fuzzy control is not uniform. In this work we have tried to adhere to the most widely used forms, with only a few deviations; we list some potentially ambiguous symbols and abbreviations here. Concerning the acronyms we have tried to use meaningful mnemonics that are as close as possible to standard forms in the literature; unfortunately the result is still somewhat confusing—proportional, pressure, and positive all begin with P.

Symbol	Meaning	Form
$\chi_c(x)$	Characteristic function representing crisp set C	(Traditional form)
$C(x)$	Characteristic function representing crisp set C	(Abbreviated form)
$\mu_A(x)$	Membership function representing fuzzy set A	(Traditional form)
$A(x)$	Membership function representing fuzzy set A	(Abbreviated form)
$x \in C$	Element x is a member of (crisp) set C	
$x \notin C$	Element x is not a member of (crisp) set C	
$A \subseteq B$	Set A is a (crisp or fuzzy) subset of set B	
\bar{A}	Set complement of (crisp or fuzzy) set A	
$A \cap B$	Intersection of (crisp or fuzzy) sets A and B	
$A \cup B$	Union of (crisp or fuzzy) sets A and B	
$ A $	Cardinality (either crisp or fuzzy) of set A	
$d_1/x_1 + d_2/x_2$	Fuzzy set for which $\mu(x_1) = d_1$, $\mu(x_2) = d_2$, and all other membership grades are zero; similarly Σ -notation can be used or \int -notation for sets with membership functions defined on continuous spaces.	
$\mathcal{P}(C)$	Crisp power set of (crisp) set C	
$\mathcal{F}(A)$	Fuzzy power set of (crisp or fuzzy) set A	
\mathcal{R}	Set of all real numbers	
\mathcal{R}^+	Set of all nonnegative real numbers	
$\min(a, b, c)$	Minimum value of quantities a , b , and c	
$\min_{i=1}^n x_i$	Minimum of all x_i values for $i = 1, 2, \dots, n$;	(Analogous to Σ -notation for summation)
Similar use is made of max and Max for taking maximum values.		
\neg	Logical negation (not)	
\wedge	Conjunction (and)	

Symbol	Meaning	Form
\vee	Disjunction (or)	
\Rightarrow	Implication	
\Leftrightarrow	Logical equivalence	
A_α	α -cut of (fuzzy) set A	
\boxplus	Addition defined on intervals of real numbers (as in α -cuts of fuzzy numbers)	
$\oplus \ominus$	Addition or subtraction defined for fuzzy numbers	
P	Proportional (control)	
PD	Proportional-plus-derivative (control)	
PI	Proportional-plus-integral (control)	
PID	Proportional-plus-integral-plus-derivative (control)	
NB	Negative big (term used as a linguistic value)	
NM	Negative medium	
NS	Negative small	
Z0	near zero	
PS	Positive small	
PM	Positive medium	
PB	Positive big	
P	Pressure error (as either base or linguistic variable)	
PC	Pressure-error change	
PS	Pressure-error sum (average)	
S	Speed error	
SC	Speed error change	
HC	Heat-control-setting change	
TC	Throttle-setting change	
IAE	Integral of absolute value of error	
ISE	Integral of square of error	
ITAE	Integral of time times absolute value of error	
ITSE	Integral of time times square of error	

References

- Adams, James L. (1986). *Conceptual Blockbusting: A Guide to Better Ideas*. Addison-Wesley, Reading, MA.
- Baldwin, J. F. (1981). Fuzzy Logic and Fuzzy Reasoning. In *Fuzzy Reasoning and its Applications* (E. H. Mamdani and B. R. Gaines, eds.). Academic Press, London.
- Baldwin, J. F., and N. C. F. Guild (1980). Modelling Controllers Using Fuzzy Relations. *Kybernetes* 9:223.
- Baldwin, J. F., and B. W. Pilsworth (1980). Axiomatic Approach to Implication for Approximate Reasoning with Fuzzy Logic. *Fuzzy Sets and Systems* 3:193.
- Barr, Avron, and Edward A. Feigenbaum (1981). *The Handbook of Artificial Intelligence*. William Kaufman, Los Altos, CA.
- Berrel, Annabel C. (1987). *Expert Systems: Strategic Implications and Applications*. Horwood, Chichester, West Sussex, UK.
- Berenji, Hamid R. (1992a). A Reinforcement Learning-Based Architecture for Fuzzy Logic Control. *International J. of Approximate Reasoning* 6:267.
- (1992b). Fuzzy Logic Controllers. In *An Introduction to Fuzzy Logic Applications in Intelligent Systems* (R. R. Yager and L. A. Zadeh, eds.). Kluwer, Boston.
- Cao, Zhiqiang, Abraham Kandel, and Lihong Li (1990). A New Model of Fuzzy Reasoning. *Fuzzy Sets and Systems* 36:311.
- Carbonell, Jaime, ed. (1990). *Machine Learning: Paradigms and Methods*. MIT Press, Cambridge, MA.
- Chen, Chieh-Li, Pey-Chung Chen, and Cha'o-Kuang Chen (1993). Analysis and Design of Fuzzy Control System. *Fuzzy Sets and Systems* 57:125.
- Chen, Yu, Zhiqiang Cao, and Abraham Kandel (1990). Application of Fuzzy Reasoning to the Control of an Activated Sludge Plant. *Fuzzy Sets and Systems* 37:1.
- Chiu, Stephen, and Masaki Togai (1988). A Fuzzy Logic Programming Environment for Real-Time Control. *International J. of Approximate Reasoning* 2:163.
- Cichocki, Andrzej, and Rolf Unbehauen (1993). *Neural Networks for Optimization and Signal Processing*. Wiley, Chichester, UK.
- Deboeck, Guido J. (1994). Why Use Fuzzy Modeling? In *Trading on the Edge: Neural, Genetic, and Fuzzy Systems for Chaotic Financial Markets* (G. J. Deboeck, ed.). Wiley, New York.
- Deboeck, Guido J., and Masud Cader (1994). Pre- and Postprocessing of Financial Data. In *Trading on the Edge: Neural, Genetic, and Fuzzy Systems for Chaotic Financial Markets* (G. J. Deboeck, ed.). Wiley, New York.
- DiStefano, J. J., III, A. R. Stubberud, and I. J. Williams (1967). *Feedback and Control Systems*. McGraw-Hill, New York.
- Dorf, Richard C. (1974). *Modern Control Systems*. Addison-Wesley, Reading, MA.
- Dubois, D., H. Prade, and R. R. Yager, eds. (1993). *Readings in Fuzzy Sets for Intelligent Systems*. Kaufmann, San Mateo, CA.
- Encyclopedia Britannica. *Yearbook of Science and the Future* (1985).
- Filev, Dimiter (1991). Fuzzy Modeling of Complex Systems. *International J. of Approximate Reasoning* 5:281.

- Filev, D., and R. R. Yager (1991). A Generalized Defuzzification Method under BAD Distributions. *J. of Intelligent Systems* 6:687.
- Furukawa, Toshiyuki, ed. (1989). *Artificial Intelligence in Medicine: Current State and Future of Medical Artificial Intelligence*. Organizing Committee of the 5th TOYOB0 Biotechnology Foundation Symposium, Chuo-ku, Tokyo.
- Gallant, Stephen I. (1993). *Neural Network Learning and Expert Systems*. MIT Press, Cambridge, MA.
- Garey, Michael R., and David S. Johnson (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, New York.
- Gause, Donald C., and Gerald M. Weinberg (1982). *Are Your Lights On?* Winthrop, Cambridge, MA.
- Geng, Nianfeng, and Itsuya Muta (1993). Design, Realization, and Improvement of Optimum Fuzzy Controller for Running Train. *J. of Japan Society for Fuzzy Theory and Systems* 5:149.
- Ginsberg, Matt (1993). *Essentials of Artificial Intelligence*. Kaufmann, San Mateo, CA.
- Goldberg, David E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, Reading, MA.
- Golten, Jack, and Andy Verwer (1992). *Control System Design and Simulation*. McGraw-Hill, London.
- Gupta, M. M., R. K. Ragade, and R. R. Yager, eds. (1979). *Advances in Fuzzy Set Theory and Applications*. North-Holland, Amsterdam.
- Gupta, M. M., and E. Sanchez, eds. (1982). *Fuzzy Information and Decision Processes*. North-Holland, Amsterdam.
- Hayashi, Isao (1993). Fusion Method between Fuzzy Control and Neural Networks (in Japanese: *Fuzzy-seigo to Neural Network to no Yuugouhou*). *Jouhou Shori* (Information Processing) 34:44.
- Hayashi, Isao, Hiroyoshi Nomura, and Noboru Wakami (1990). Acquisition of Inference Rules by Neural-Network-Driven Fuzzy Reasoning (in Japanese: *Neural-net-kudougata Fuzzy-suiron ni yoru Suiron-rule no Kakutoku*). *J. of Japan Society for Fuzzy Theory and Systems* 2:585.
- Hayashi, K., A. Ohtsubo, I. Muta, T. Hoshino, and N. Matuso (1993). Improvement of Control Performances for Low-Dimensional Labelings (in Japanese: *Teijigen-bunkatsu ni okeru Fuzzy-seigo-seinou no Kaizen*). *Proc. of 9th Fuzzy System Symposium* (Sapporo, May 1993, Japan Society for Fuzzy Theory and Systems), 321.
- Hirota, Kaoru, ed. (1992). *Fuzzy Gijutsu no Jitsuyouka-ouyou* (in Japanese: The Practical Application of Fuzzy Technology). Springer-Verlag Tokyo.
- Holland, John H. (1992). *Adaptation in Natural and Artificial Systems*. MIT Press, Cambridge, MA.
- Holmblad, Lauritz P., and Jens-Jorgen Østergaard (1982). Control of a Cement Kiln by Fuzzy Logic. In *Fuzzy Information and Decision Processes* (M. M. Gupta and E. Sanchez, eds.). North-Holland, Amsterdam.
- Hunter, Lawrence, ed. (1993). *Artificial Intelligence and Molecular Biology*. AAAI Press, Menlo Park, CA.
- Johnson, Colin R. (1994). Recent US Trends in Fuzzy Inference Chips. *J. of Japan Society for Fuzzy Theory and Systems* 6:446.
- Jones, Joseph L., and Anita M. Flynn (1993). *Mobile Robots: Inspiration to Implementation*. Peters, Wellesley, MA.
- Jones, Peter (1989). Uncertainty Management in Expert Systems. In *Expert Systems: Principles and Case Studies* (R. Forsyth, ed.). Chapman and Hall, London.
- Jumarie, Guy (1990). *Relative Information: Theories and Applications*. Springer-Verlag, Berlin.
- Kandel, Abraham (1986). *Fuzzy Mathematical Techniques with Applications*. Addison-Wesley, Reading, MA.
- Kasper, C., and Hans-Jurgen Zimmermann (1994). Hardware for Fuzzy Logic Applications. *J. of Japan Society for Fuzzy Theory and Systems* 6:451.
- Katayama, R., Y. Kajitani, K. Kuwata, M. Watanabe, and Y. Nishida (1993). Various Self-Generating Algorithm for Neuro Fuzzy Model with Radial Basis Function (in Japanese: *Neurofuzzy Model ni*

- okeru Shiju no Jiko-zoushoku-gata Algorithm). *Proc. of 9th Fuzzy System Symposium* (Sapporo, May 1993, Japan Society for Fuzzy Theory and Systems), 421.
- Kaufmann, Arnold, and Madan M. Gupta (1985). *Introduction to Fuzzy Arithmetic: Theory and Applications*. Van Nostrand Reinhold, New York.
- (1988). *Fuzzy Mathematical Models in Engineering and Management Science*. Elsevier Science, Amsterdam.
- Kawase, Shin, and Niro Yanagihara (1993). On the Truth Space Approach and the Direct Approach in Fuzzy Reasoning (in Japanese: *Fuzzy Suiron ni okeru Kansetsuhou to Chokusetsuhou ni tsuite*). *Proc. of 9th Fuzzy System Symposium* (Sapporo, May 1993, Japan Society for Fuzzy Theory and Systems), 545.
- Klimasauskas, Casimir C. (1994). Neural Network Techniques. In *Trading on the Edge: Neural, Genetic, and Fuzzy Systems for Chaotic Financial Markets* (G. J. Deboeck, ed.). Wiley, New York.
- Klir, George J. (1990). A Principle of Uncertainty and Information Invariance. *International J. of General Systems* 17.
- Klir, George J., and Tina A. Folger (1988). *Fuzzy Sets, Uncertainty, and Information*. Prentice-Hall, Englewood Cliffs, NJ.
- Klir, George J., and Bo Yuan (1995). *Fuzzy Sets and Fuzzy Logic: Theory and Applications*. Prentice-Hall, Englewood Cliffs, NJ.
- Kosko, Bart (1992). *Neural Networks and Fuzzy Systems: A Dynamical Systems Approach to Machine Intelligence*. Prentice-Hall, Englewood Cliffs, NJ.
- (1993). *Fuzzy Thinking*. Hyperion, New York.
- Koza, John R. (1992). *Genetic Programming*. MIT Press, Cambridge, MA.
- Kung, Sun Yuan (1993). *Digital Neural Networks*. Prentice-Hall, Englewood Cliffs, NJ.
- Levy, Steven (1992). *Artificial Life: The Quest for a New Creation*. Penguin, London.
- Lewis, Harold, and Gaik Sim Lewis (1994). Concerning the Role of Machine Learning in Business and Public Administration. *Shougaku Ronshuu* (J. of Commerce, Economics, and Economic History) 63:35.
- Losee, Robert M., Jr. (1990). *Science of Information: Measurement and Applications*. Academic, San Diego, CA.
- Magrez, P., and P. Smets (1975). Fuzzy Modus Ponens: A New Model Suitable for Applications in Knowledge-Based Systems. *International J. of Intelligent Systems* 4:181.
- Mamdani, E. H. (1974). Applications of Fuzzy Algorithms for Control of Simple Dynamic Plant. *Proc. IEE* 121:1585.
- Mamdani, E. H., and S. Assilian (1975). An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller. *International J. of Man-Machine Studies* 7:1.
- Mamdani, E. H., and B. R. Gaines, eds. (1981). *Fuzzy Reasoning and Its Applications*. Academic, London.
- Mammone, Richard J., and Yehoshua Y. Zeevi, eds. (1991). *Neural Networks: Theory and Applications*. Academic, Boston.
- Maxwell, James C. (1868). On Governors. *Proc. Royal Society* 16:270.
- Minsky, Marvin L., and Seymour A. Papert (1988). *Perceptrons*. MIT Press, Cambridge, MA.
- Mizumoto, Masaharu (1992a). Fuzzy Reasoning (1) [in Japanese: *Fuzzy Suiron (1)*]. *J. of Japan Society for Fuzzy Theory and Systems* 4:256.
- (1992b). Fuzzy Reasoning (2) [in Japanese: *Fuzzy Suiron (2)*]. *J. of Japan Society for Fuzzy Theory and Systems* 4:433.
- (1993a). Equivalence of Fuzzy Singleton-Type Reasoning Method and Product-Sum-Gravity Method (in Japanese: *Fuzzy Singleton-gata-suiron-hou to Daisuu-seki-kazan-juushin-hou no Toukasei*). *Proc. of 9th Fuzzy System Symposium* (Sapporo, May 1993, Japan Society for Fuzzy Theory and Systems), 313.

- (1993b). Are Max and Min Suitable Operators for Fuzzy Control Methods? (in Japanese: *Fuzzy-seigyo ni oite Max to Min no Shiyou ha Tekitou ka?*). *Proc. of 9th Fuzzy System Symposium* (Sapporo, May 1993, Japan Society for Fuzzy Theory and Systems), 317.
- Mizumoto, Masaharu, and Hans-Jurgen Zimmermann (1982). Comparison of Fuzzy Reasoning Methods. *Fuzzy Sets and Systems* 8:253.
- Müller, Berndt, and Joachim Reinhardt (1990). *Neural Networks: An Introduction*. Springer-Verlag, Berlin.
- Nafarieh, Asghar, and James M. Keller (1991). A New Approach to Inference in Approximate Reasoning. *Fuzzy Sets and Systems* 41:17.
- Nakanishi, H., I. B. Turksen, and M. Sugeno (1993). A Review of Six Reasoning Methods. *Fuzzy Sets and Systems* 57:257.
- Negoita, Constantin V., and Dan Ralescu (1987). *Simulation, Knowledge-Based Computing, and Fuzzy Statistics*. Van Nostrand Reinhold, New York.
- Nguyen, Hung T. (1993). Aspects of Robustness in Fuzzy Control. *J. of Japan Society for Fuzzy Theory and Systems* 5:1051.
- Nilsson, Nils J. (1990). *Mathematical Foundations of Learning Machines*. Kaufmann, San Mateo, CA.
- Nomura, Hiroyoshi, Isao Hayashi, Noboru Wakami (1992). A Self-Tuning Method of Fuzzy Reasoning by Delta Rule and Its Application to a Moving Obstacle Avoidance (in Japanese: *Delta-rule ni yoru Fuzzy-suiron no Jidou-tuning Shuhou to Shougaibutsu-kaihi he no Oyou*). *J. of Japan Society for Fuzzy Theory and Systems* 4:379.
- Peatman, John B. (1977). *Microcomputer-Based Design*. McGraw-Hill and Kogakusha, Tokyo.
- Pedrycz, W. (1991). Fuzzy Modelling: Fundamentals, Construction, and Evaluation. *Fuzzy Sets and Systems* 41:1.
- Peretto, Pierre (1992). *An Introduction to the Modeling of Neural Networks*. Cambridge University Press, Cambridge, MA.
- Pierce, John R. (1980). *An Introduction to Information Theory: Symbols, Signals, and Noise*. Dover, New York.
- Piskunov, Alexandre (1992). Fuzzy Implication in Fuzzy Systems Control. *Fuzzy Sets and Systems* 45:25.
- Polya, George (1945). *How to Solve It: A New Aspect of Mathematical Method*. Princeton University Press, Princeton, NJ.
- Procyk, T. J., and E. H. Mamdani (1979). A Linguistic Self-Organizing Process Controller. *Automatica* 15:15.
- Raggett, Jenny, and William Bains (1992). *Artificial Intelligence from A to Z*. Chapman and Hall, London.
- Ralescu, Anca, and H. Narazaki (1991). Integrating Artificial Intelligence Techniques in Linguistic Modeling from Numerical Data. *Proc. of IFES Symposium* (Yokohama, Nov. 1991), 328.
- Rich, Elaine, and Keven Knight (1991). *Artificial Intelligence*. McGraw-Hill, New York.
- Rowland, James R. (1986). *Linear Control Systems: Modeling, Analysis, and Design*. Wiley, New York.
- Saitou, Yuuji, Isao Hirano, and Michio Sugeno (1993). Flight Control of an Unmanned Helicopter Using Fuzzy Theory (in Japanese: *Fuzzy Riron wo Mochiita Myjin Helicopter no Hikou-seigyo*). *Proc. of 9th Fuzzy Systems Symposium* (Sapporo, May 1993, Japan Society for Fuzzy Theory and Systems), 37.
- Sanseido Co., Ltd. (1990). *Sanseido's Daily Concise Japanese-English Dictionary*, Tokyo.
- Savory, Stuart E., ed. (1988). *Expert Systems in the Organisation: An Introduction for Decision-Makers* (Barbara Chapman, trans.). Horwood, Chichester, West Sussex, UK.
- Siler, William, and Hao Ying (1989). Fuzzy Control Theory: The Linear Case. *Fuzzy Sets and Systems* 33:275.
- Simon, Herbert A. (1981). *Sciences of the Artificial*. MIT Press, Cambridge, MA.

- Smith, Peter (1988). *Expert System Development in Prolog and Turbo Prolog*. Sigma, Wilmslow, Cheshire, UK.
- Sugeno, M., and G. T. Kang (1986). Fuzzy Modelling and Control of Multilayer Incinerator. *Fuzzy Sets and Systems* 18:329.
- Sugeno, Michio (1985). An Introductory Survey of Fuzzy Control. *Information Sciences* 36:59.
- (1987). Control (in Japanese: *Seigyo*). In *Fuzzy Systems Primer* (in Japanese: *Fuzzy System Nyuumon*) (T. Terano, K. Asai, and M. Sugeno, eds.). Ohm, Tokyo.
- Takagi, Tomohiro, and Michio Sugeno (1985). Fuzzy Identification of Systems and Its Applications to Modeling and Control. *IEEE Transactions on Systems, Man, and Cybernetics* 15:116.
- Tanaka, Kazuo, and Manabu Sano (1993). CO Concentration Control Simulation by Neuro-Fuzzy Control. *Proc. of 9th Fuzzy System Symposium* (Sapporo, May 1993, Japan Society for Fuzzy Theory and Systems), 445.
- Terano, Toshiro (1981). *A Recommendation of Vague Engineering: Engineering from New Concepts* (in Japanese: *Aimai Kougaku no Susume: Atarashii Hassou kara no Kougaku*). Koudansha, Tokyo.
- Terano, T., K. Asai, and M. Sugeno, eds. (1987). *Fuzzy Systems Primer* (in Japanese: *Fuzzy System Nyuumon*). Ohm, Tokyo.
- Terano, T., K. Asai, and M. Sugeno, eds. (1989). *Applied Fuzzy Systems Primer* (in Japanese: *Ouyou Fuzzy Systems Nyuumon*). Ohm, Tokyo.
- Togai, Masaki (1994). Perspectives and Trends of Digital Fuzzy Inference Processors (in Japanese: *Digital Suiron Chip no Genjou to Hatten*). *J. of Japan Society for Fuzzy Theory and Systems* 6:440.
- Tong, Richard M. (1984). A Retrospective View of Fuzzy Control Systems. *Fuzzy Sets and Systems* 14:199.
- Tsuchiya, Toshio, Yukihiko Matsubara, and Mistuo Nagamachi (1993). Learning Fuzzy Rule Parameters Using Genetic Algorithm (in Japanese: *Fuzzy-suiron-rule no Gakushuu ni okeru Identeki-algorithm no Yuukousei ni Kan-suru Kousatsu*). *Proc. of 9th Fuzzy System Symposium* (Sapporo, May 1993, Japan Society for Fuzzy Theory and Systems), 121.
- Tsukamoto, Yahachiro (1979). An Approach to Fuzzy Reasoning Method. Reprinted in *Readings in Fuzzy Sets for Intelligent Systems* (D. Dubois, H. Prade, and R. R. Yager, eds., 1993). Kaufmann, San Mateo, CA.
- Turing, Alan (1950). Can a Machine Think? *Mind*, vol. 59. [Reprinted in *World Treasury of Physics, Astronomy, and Mathematics* (T. Ferris, ed., 1991)]. Little, Brown, Boston.
- Van de Vegte, John (1986). *Feedback Control Systems*. Prentice-Hall, Englewood Cliffs, NJ.
- Wang, Zhenyuan, and George J. Klir (1992). *Fuzzy Measure Theory*. Plenum, New York.
- Wiener, Norbert (1947). *Cybernetics*. Technology, Cambridge, MA.
- Winston, Patrick H. (1992). *Artificial Intelligence*. Addison-Wesley, Reading, MA.
- Yager, Ronald R., and Lotfi A. Zadeh, eds. (1992). *An Introduction to Fuzzy Logic Applications in Intelligent Systems*. Kluwer, Boston.
- Yamakawa, Takeshi (1994). Perspectives and Trends of Analog Fuzzy Inference Processors (in Japanese: *Analog Suiron Chip no Genjou to Hatten*). *J. of Japan Society for Fuzzy Theory and Systems* 6:426.
- Yamakawa, Takeshi (1988). *Conception of the Fuzzy Computer: Sixth Generation Computers* (in Japanese: *Fuzzy Computer no Hassou: Dai-6 Sedai Computer*). Koudansha, Tokyo.
- Yamamoto, Kazumi, Yuji Kubo, and Junzo Watada (1993). Learning Membership Functions for Fuzzy Controller with Genetic Algorithm (in Japanese: *Iden-algorithm niyoru Fuzzy-seigyo no Membership-kansuu no Gakushuu*). *Proc. of 9th Fuzzy System Symposium* (Sapporo, May 1993, Japan Society for Fuzzy Theory and Systems), 97.
- Yasunobu, S., S. Miyamoto, and H. Ihara (1983). Fuzzy Control for Automatic Train Operation System. *Proc. of 4th IFAC/IFIP/IFORS International Conf. on Control in Transportation Systems* (Baden-Baden), 33.
- Yip, Kenneth Man-Kam (1991). *Kam: A System for Intelligently Guiding Numerical Experimentation by Computer*. MIT Press, Cambridge, MA.

- Zadeh, Lotfi A. (1972). A Rationale for Fuzzy Control. *Trans. ASME, J. of Dynamic Systems, Measurement, and Control* **94**(Ser. G):3.
- (1965). Fuzzy Sets. *Information and Control* **8**:338.
- (1975). The Concept of a Linguistic Variable and Its Application to Approximate Reasoning, Part 1. *Information Sciences* **8**:199.
- Zhao, R. H., and R. Govind (1991). Defuzzification of Fuzzy Intervals. *Fuzzy Sets and Systems* **43**:45.
- Zimmermann, Hans J. (1987). *Fuzzy Sets, Decision Making, and Expert Systems*. Kluwer, Boston.

About the Author

The book developed in the context of cross-cultural experiences. A Japanese co-worker once advised me that the first few years of one's working career make up a crucial time in one's life. The habits and the mindset one develops during this period will influence the rest of one's life in ways that are quite dramatic. I suspect that this theory is correct, or at least it seems so in my case.

In 1977, as a 20-year old, I arrived in Japan with about \$100 in my pocket, ready to start my first "real" job. I had just completed a B.S., majoring in mathematics (primarily applied mathematics), and I had learned some of the Japanese language, mainly on my own. (Few American universities offered extensive formal classes in Japanese at that time.) In the 1970s it was still quite unusual for a Westerner to take a job in Japanese industry, but one medium-sized electrical manufacturing firm in the very historical city of Kyoto was willing to take a chance by offering to hire me in the same capacity as a newly graduated Japanese. This meant that my starting salary, including bonuses, was projected to be only the yen equivalent of U.S. \$4000 per year, a very small fraction of what my classmates were being offered in America, but for a romantically idealistic 20-year old, salary is never a particularly important consideration.

The late 1970s were a particularly difficult time to be a foreigner living alone in Japanese society, and in many ways the conditions were much more difficult than I had anticipated, but I was partly compensated in that I think I have never had a better opportunity to learn than I did while working in that company. After several months of "rotation" through the typical menial assignments, I was placed semi-permanently in what I will translate as the Special Projects Office.

The main line of the company was various types of lighting products, but for several reasons we desperately desired to diversify in completely new directions, and that was the function of our office. Under a rather complex and often ineffective management structure, there were about eight or ten of us who did the actual work, and we were all crowded into such a small room that there was hardly space to put the chairs between the desks. The air conditioning system with which we were provided was an old dilapidated one that had long since been retired from the factory floor. It amply chilled our small room when it worked, but it broke down so frequently that the three or four of us youngest workers were assigned the task of

spending our first hour or so nearly every summer morning coaxing it to function again.

Much of our research equipment was even more “hokey” than the air conditioner. In those days, the founder of the company and chairman of the board explained almost monthly a new reason for being on an austerity budget. But I can also say that, despite this obsession with cutting costs to the bone, there was less red tape than I have ever experienced in any other job when it came to getting supplies and equipment that were really necessary. Anyway, it was a rather colorful place to work.

Our office was responsible for suggesting new, alternative product lines (anything other than lighting products), conducting market research to determine their feasibility, developing and testing designs for them, arranging for their manufacture by subcontractors, testing their quality, and marketing them. In addition to doing all of this for alternative product lines, we were often called upon to assist the company in many other ways, such as automating various manufacturing processes in the lighting factories and starting up a winery as a subsidiary. As for the nature of the alternative product lines, some of these were quite major systems. Two of our most successful were industrial pollution-control systems and solar energy equipment.

None of us really knew what we were doing much of the time, but we usually worked together, especially when management stayed out of our hair, in a more perfect sense of teamwork than I could ever have imagined possible had I not experienced it, and it was fabulously exciting. We each wore very many different hats, but we all recognized each other’s strengths, and we were remarkably good at inspiring self-confidence in our teammates to be able to figure out how to get things done.

One fellow, about 30-years old, was our primary salesman, but all of us interacted with customers and potential customers in various roles. A goofy, young American, speaking broken Japanese, was very much a novelty in Japanese industry, and my co-workers and I found this to be quite an asset. All I had to do was bow to a few people and give them my name card, and they would be suddenly much more interested in our products. When conducting marketing surveys, particularly if I went out alone, people somehow trusted me so easily that they were quite willing to provide me with all sorts of information that otherwise would have been unavailable to us.

The technical leader of our group was a Mr. Kuwahara, then in his early thirties, who had been trained as a chemical engineer, and he taught me a great deal. This may come as a surprise to people without industrial experience, but often in technical development areas, occasionally even in supposedly high-tech areas, there are very few people who would be described as “nerds” of any sort. Rather, often these environments are populated with “macho” people who consider themselves as rugged pragmatists, with a somewhat anti-intellectual attitude. They may believe,

for example, that mathematics (outside of arithmetic and simple geometry) has almost nothing whatsoever to do with the real world of problem-solving. The company I worked for in Japan was very much like that, and I have encountered the same attitude many times in the United States as well. Kuwahara was a notable exception. He was an extremely nerdy fellow who seemed to feel very out of place and rather lonely in a company full of macho men. He took a special interest in me, and I suppose he thought of me as his fellow nerd.

Kuwahara was probably the one who arranged for me to come to the Special Projects Office, and once I was there he seemed to take more pleasure in my training than in that of anyone else. His attitude seemed to be, “You studied math in university without even knowing how useful it can be. Now, let us both enjoy the process of introducing you to the utility of your knowledge.” He supplied me with many textbooks (mostly in English, which I still read much more efficiently than Japanese) on heat-transfer analysis, control theory, systems design, and so on. He seemed to feel that I could, for example, absorb the contents of several thick books on heat transfer in a week or so, and become the resident expert on solar collector-design analysis. I do not know whether he really believed this or was only pretending, but such apparent confidence often motivates a young person to try something he hardly feels capable of himself. Soon I did find more elegant new ways to calculate the potential effectiveness of various solar collector designs. I did become the one to design all new control systems, and to improve the designs of the old ones. I became the one who did the preliminary custom designs for pollution-control systems in response to the requirements of given customers, and I even used one of those big, clumsy old-fashioned Japanese-language typewriters to type up the estimate forms based on my preliminary designs. In short, though I still wore many hats, I had become primarily the systems-design-and-analysis person in our intense little office.

I sometimes find myself talking to Americans who think this sort of thing is quite incredible. “After all,” they say, “aren’t the Japanese way ahead of us in math and computers? How could you have that sort of role in Japan?” It is true that, particularly on the secondary level, the average Japanese student is years ahead of the average American student in terms of mathematical knowledge, but one key word here is average. Differences between the mathematically inclined Japanese students and the mathematically inclined American students, though they do exist, are not nearly so significant. There is arguably less difference still when comparing college students. Furthermore, attitudes are often more important than mere knowledge. I find that one result of the intense Japanese examination system is to leave the average Japanese young adult totally fed up with mathematics. As for computers, there are many subfields to information technology, and a few of these are completely dominated by Japanese industry, but when the field is taken as a whole, the United States has much more justification in claiming dominance than Japan does. This is particularly true in the case of software, especially during the 1970s,

when surprisingly few technical people in Japan had any knowledge at all of how to write even a simple computer program.

I do not claim that my early industrial experiences were extraordinary. Surely, many other people have experienced excitingly intense technical working environments. What I do claim is that, as a result of my experiences in that office, it was quite natural for me to develop certain attitudes or strongly held beliefs about what is really important in that sort of work. Although I was still skeptical that there was any such thing as a good organization, I had learned that teamwork, when it is really right, can be a very beautiful thing. Unlike some self-styled rugged pragmatists, I do believe mathematics to be an essential tool in many types of real-world environments. I also believe, perhaps more so than most people trained only in engineering, that the most seemingly abstract approach often leads to the most widely applicable, and therefore practical, result. On the other hand, simplicity of concept and simplicity of approach are very great virtues, especially when one has very many other things to do. Furthermore, intuition is also a valuable tool under these circumstances, particularly when used in conjunction with mathematics. Finally, if faced with austerity budgets, one can just make the best of it and learn to improvise. It is probably the most fun to work that way anyway, provided that austerity is not taken to mean an increase in red tape.

As much as I had learned in that Kyoto-based company, in late 1981 I felt it was time to leave Japan. Having made the mistake of marrying too young, I already had a growing family at age twenty-five, and it began to seem that the still relatively low Japanese salary, the very high Japanese cost of living, and the lack of opportunity to further my formal education were all much more significant considerations than they had been before. We made the difficult move back to the United States, and soon I was able to find a position in a development laboratory of a major corporation.

After spending a year or two getting established in my new life in America, and finding I had an excess of mental energy seeking release, I began to use my evenings to do graduate work at nearby Binghamton University or, as it was then called, SUNY–Binghamton. I soon became acquainted with Binghamton's Department of Systems Science, and as I began to understand what systems science meant, I found it quite appealing. It struck me that this was an academic and slightly more theoretical version of just that type of work I had done in Japan. I had completed my bachelor's degree in 3 years so at first it seemed only natural to earn graduate degrees as quickly as was possible for a part-time student. But I soon turned into one of those people for whom "going back to school" becomes one of life's great pleasures, and degrees began to seem more of an afterthought. It was a joy to learn the viewpoints of the various professors, to work on research with some of them, and to teach some of one professor's classes whenever he was out of town.

In about 1988, when it came time to set up my doctoral program, I decided to work under the highly esteemed George Klir in the area of fuzzy sets and systems.

I had to be more specific than that, of course, and the environment in which I was studying contributed to my choice of topics in a rather ironic way. There is perhaps no one best answer as to what systems science should mean, and I do not mean this as a criticism, but something that seemed slightly unusual to me about Binghamton's Department of Systems Science, at least at that time, was that there appeared to be relatively little interest in control theory or cybernetics. In most places in the world, control is viewed as very much a central issue in any theoretical or applied study of systems, but Binghamton seemed to be largely an exception to that. Though George Klir was then already deeply involved in fuzzy sets and fuzzy measures theories, his emphasis and key contributions were more on fuzzy measure theory and its implications relative to information theory than on fuzzy control applications. It was not in any absolute sense that I was an expert on control or exclusively interested in the field, but in relative terms I was clearly more interested in it than those around me at that time. Also, it was by then obvious that Japanese researchers dominated fuzzy control research, and I knew some of the Japanese language. With these considerations in mind, it was quite appropriate for George to strongly encourage me to view fuzzy control as my specialized field of study. I am happy that he did, and happy that I agreed.

By this time, I was on the road to a significant career change. My experiences at Binghamton University convinced me that I thoroughly enjoyed preparing and delivering lectures as well as working on research in the relatively independent way that is often possible for academics. Also, there were several personal and professional reasons for which I felt it was time to move back to Japan if it was in any way possible to do that. Naturally by this time I had made several contacts in the Japanese academic world, both from Japanese researchers visiting Binghamton and from my own visits to Japanese universities when I took vacations in Japan. Thanks to one of these contacts, Professor Kyoji Hoshino of Fukushima University, I suddenly found myself with the prospect of achieving two goals at once—entering the full-time academic world and returning to Japan. By the spring of 1990, when I resigned from my position in industry and left for Fukushima, I had completed my comprehensive exam and written a dissertation prospectus. In the prospectus, I committed myself to the topic of examining the significance of dynamic compensation in fuzzy control.

It would be reasonable to assume that when I went back to Japan in 1990 I would be thrown into a rich environment for studying fuzzy control, and therefore would no longer be left to proceed so much in isolation, but this was not to be the case. My position was in Fukushima's Faculty of Economics, what in America might be called a School of Economics and Business Administration. No one else at Fukushima was interested in fuzzy control, and only one of my colleagues there at the time was even moderately interested in other applications of fuzzy set theory. In fact, some of my colleagues at Fukushima were (and still are) extremely skeptical about the field. For that matter, in all of the Tohoku region of northern Honshu,

there were perhaps only about a dozen or so researchers active in fuzzy control or related fields as we found when we attempted to start the regional chapter of SOFT, the main organization for fuzzy set research in Japan. I did eventually participate to some degree in various nationwide research groups, but for most of the time I was left to consider my research in isolation and according to my own approach as it evolved.

I feel that we have a strong tradition of academic freedom at Fukushima University, and although my research in fuzzy control seems slightly out of place in the Faculty of Economics, my colleagues generally offer their moral support for it. Many of them seem to be uncertain as to just what an associate professor of information systems should work on in such an environment anyway. This does not mean however that any sources of funding are readily available to someone in my position for setting up anything at all like an actual engineering laboratory. The type of research funding I have received has been a few thousand here or there, enough to obtain some personal computing equipment and some books, and to travel a few times per year to other universities in Japan.

Furthermore, from the point of view of my colleagues, the completion of my doctoral dissertation need not have been considered a high priority. I am told that it is extraordinarily difficult in Japan to obtain a doctorate in any of the social sciences. Therefore, in a school with a leaning toward the social sciences and in a relatively rural part of the country, it is considered very much the norm to never go beyond a master's degree. To publish scholarly papers on a regular basis and on a variety of relevant topics is considered more important than having the highest degrees attainable.

My point is not to complain about my research conditions, nor is it to offer excuses. Very much to the contrary, I have been generally very happy with the sort of academic life I have found in Fukushima. The point is that despite going to Japan, and despite entering the academic world, I still found myself quite outside the mainstream of fuzzy control research. When I communicate about my research to my colleagues and my students, I need to find ways to help them most easily understand a field that seems quite alien to them. From some perspectives, all of this may seem somewhat regrettable, but from the perspective of encouraging a sense of creativity or originality in one's thinking, these conditions have been ideal. With the one exception that the red tape often gets on my nerves, I have been much happier at Fukushima University than I have ever been before in my career, and I am deeply grateful to the faculty, the students, and the staff. In various subtle ways, they have all contributed to this book.

Harold W. Lewis, III
Troy, New York

Index

- Accuracy (of control devices), 13–14, 16, 18, 21, 209–210, 236–256
Actuators, 12, 167–171, 229
Adaptive control, 148–149
Algorithms, 32–33, 275
Analog electronics, 20–21, 122, 161
Analytical tools, 15, 123
Approximate reasoning, 6, 59–60, 77–92, 103–122, 124, 136, 143–144
Artificial intelligence, 6, 29–58, 92–93, 95, 137, 191, 270, 271–276
Auto-tuning, 7, 126, 137, 144–156, 191–226, 237, 256–259, 262, 266–267, 269–270

Back propagation networks, 52–53, 151–152, 153, 177–178, 179–185
Backward chaining, 48–49
Basic defuzzification distributions, 91–92

Cement kilns, 128–130
Characteristic functions, 61, 63, 82–83
Classical methodology of fuzzy control, 5–6, 95–137, 139, 266
Classical set theory, 60–62
Consumer appliances, 131–135, 150–152, 153–154
Control error, 10, 12, 16–20, 155–156, 193–195, 207–210, 228–229, 238–256, 269
Conventional control, 6, 7, 9–28, 95, 136–137, 270
Culture and fuzzy control, 4, 132–135
Cybernetics, 9, 11

Defuzzification, 78, 88–92, 104, 109–110, 113, 141, 197
Differential equations, 15, 23, 148
Digital electronics, 20–21, 122–125, 132–134, 142–143, 161
Domain-specific knowledge, 35–36, 45, 46, 50, 180–185
Drawbacks of conventional control, 9, 15, 26–28, 136
Drawbacks of fuzzy control, 136–137

Dynamic compensation, 7, 15, 16–20, 139, 156–157, 191–226, 227, 261–265

Education in fuzzy control, 3–4
Expert systems, 35–36, 41–50, 55–56, 78–80

Feedback, 10–14, 20, 152, 192
Feigenbaum, E., 31, 275
Forward chaining, 48–49
Fuzzification, 78, 108, 111
Fuzzy arithmetic, 62, 70, 72–73
Fuzzy composition, 76, 87, 89, 116
Fuzzy control in Japan, 3–4, 126, 130–136, 148–154
Fuzzy inference (logic), 59–60, 62, 77–92, 103–122, 123–125, 127–128, 130, 136, 143–144, 152, 195–197, 269, 277–281
Fuzzy linguistic values, 107, 140–141
Fuzzy management science, 70, 73
Fuzzy measure theory, 59
Fuzzy models, 145–147, 152–154, 162, 175–177
Fuzzy real values, 107, 140–141
Fuzzy set theory, 6, 59–94, 95, 265–268
 α -cuts, 67, 68, 71–72
 convex fuzzy sets, 68, 70
 extension principle, 67–68
 fuzzy numbers, 67, 70–73, 102, 125
 trapezoidal, 70–71, 102
 triangular, 71–73, 102, 125
 fuzzy power sets, 69, 106, 107, 140
 fuzzy relations, 68–70, 73–77, 101–103, 104, 106, 111, 115, 122, 210–226, 268
 normal fuzzy sets, 67, 70
 notation, 65–66
 subsets of fuzzy sets, 69
Fuzzy singleton method, 140–141

Gains (in control), 13–14, 17
Generalized *modus ponens*, 77, 86–87, 114–115
Genetic algorithms, 7, 46, 56–57, 139, 154–156, 207–218, 256–259
Genetic programming, 46

- Helicopter control, 135–136
 Heuristic knowledge, 9, 21, 36–40, 43–44, 50, 97–101, 125–126, 143, 148, 176–181, 266
 sources of, 97–98
 Heuristics, 31–33, 38–40, 43–44, 50, 58, 128, 174–175, 257, 275–276
 Hierarchical control, 130, 131, 135–136
 Holmblad, L.P., 113–114, 128–130, 266
 Implementation of control designs, 20–21, 122–125, 126–130, 142–143, 195–197, 211–218
 Inference engines, 47–50, 78–79, 88–89, 103–122, 195–197, 277–281
 Information processing, 30–31
 Intuition, 10, 11, 17, 19–20, 32, 38–40, 43, 136, 139, 143–144, 146, 266
 Knowledge bases, 31, 36, 41, 78–81, 139, 218–226
 Knowledge engineering, 41, 42, 44–45, 149, 151
 Knowledge processing, 30–31
 Knowledge representation, 6, 31, 34, 41–43, 47–50, 55–56
 Laplace transform, 15
 Linguistic hedges, 76–77
 Linguistic variables, 6, 59, 62, 73–77, 78, 101–103, 104, 106, 111, 121–122, 139, 146, 152–153, 155–156, 207–226
 LISP language, 34
 Machine learning, 42, 45–46, 50–57, 144–156, 266
 Mamdani, E.H., 6, 113, 127–128, 142
 Manual operators, 12, 19, 164–166
 Mathematical analysis, 14–15, 16, 148, 173–174
 Mathematical formalisms, 5
 Membership functions, 63–66, 84–92
 Models in control, 14–15, 23, 27, 28, 144–148, 154–156, 161–162, 173–189, 228
 Models of human expertise, 36–40
 Multivariate models, 15, 20, 27–28, 136, 164–166
 Naive physics, 97, 146
 Neural nets (models), 7, 35, 46, 50–56, 132, 139, 147–154, 162, 174, 176–185, 196
 Noise, 10, 12, 21, 22, 122
 Nonlinear models, 15, 27–28, 161
 On–off control, 11
 Overshoot, 12, 24
 Pascal language, 196, 277–281
 Percent overshoot, 24, 238–256
 Performance of control devices, 12–14, 15, 16–20, 21–26, 209–210, 227–259
 Plant (control), 10, 12, 144–148, 149, 173–180, 192
 Polya, G., 31, 66
 PROLOG language, 49
 Proportional (P) control, 13, 16, 17, 18, 20, 139, 191–193, 195, 196, 198, 201, 210–211, 218–220, 238–259, 261–265
 Proportional-plus-Derivative (PD) control, 16–17, 139, 157, 191–196, 199–201, 207, 220–222, 238–259, 261–265
 Proportional-plus-Integral (PI) control, 16, 17–19, 157, 191, 192, 194–196, 201, 222–223, 238–259, 261–265
 Proportional-plus-Integral-plus-Derivative (PID) control, 16, 19, 20, 157, 191, 192, 194–196, 198, 202–207, 223–226, 238–259, 261–265
 Rationale of fuzzy control, 265–268
 Responsiveness, 13, 16, 17, 18, 21, 209–210, 236–256
 Rise time, 24–25
 Robustness, 126, 137, 261
 Rule-based systems, 43–44, 48–50, 78–81, 97–101, 105–106, 111, 115, 121, 127–128, 129–130, 139, 142, 146, 197–207, 218–226
 formats of, 99–101
 Sendai subway, 130–131, 138
 Sensors, 11, 12, 167–171
 Sequential control, 10–11, 136
 Servomechanisms, 11
 Settling time, 25, 238–256
 Simulation, 7–8, 161, 173–189
 Software packages, 49, 123–125
 Stability, 12–13, 16, 17, 18, 21, 209–210, 236–256
 Steady-state conditions, 13–14, 18, 20, 21, 236–256

- Steam engines, 7, 12, 127–128, 142, 161–171, 173, 178, 257
Sugeno, M., 113–114, 135–136, 145, 175
Symbolic processing, 33–35, 122
System design, 95–96, 126, 192–195
- T-norms, 84–85
Term sets, 74–76, 78, 80, 99–100, 101–102, 104–105, 110, 130
Test input signals (functions), 21–23, 197–198, 207, 218, 227–256
 impulse, 22, 25, 232–236, 242–246, 250–253
 parabolic, 23
 ramp, 23, 25, 232–236, 245–248, 250–256
 step, 22, 24, 208, 232–236, 238–243, 245–251
Test plants, 7, 8, 161–171, 173–189, 228, 238, 270
- Time lags, 11–12
Trade-offs in design, 14, 16, 18, 237–238
Training data, 181–185
Train operation, 130–131, 138, 141
Transient conditions, 17, 18, 20, 21–25, 236–256
Tree search, 49
Tuning of control devices, 10, 17, 18–19, 20, 125–126, 137, 191
Turing test, 272–276
- Uncertainty representation, 43, 47–48, 58
- Validation data, 185–189
Variable reduction, 141–143
- Wiener, N., 9
- Zadeh, L.A., 27, 113, 134