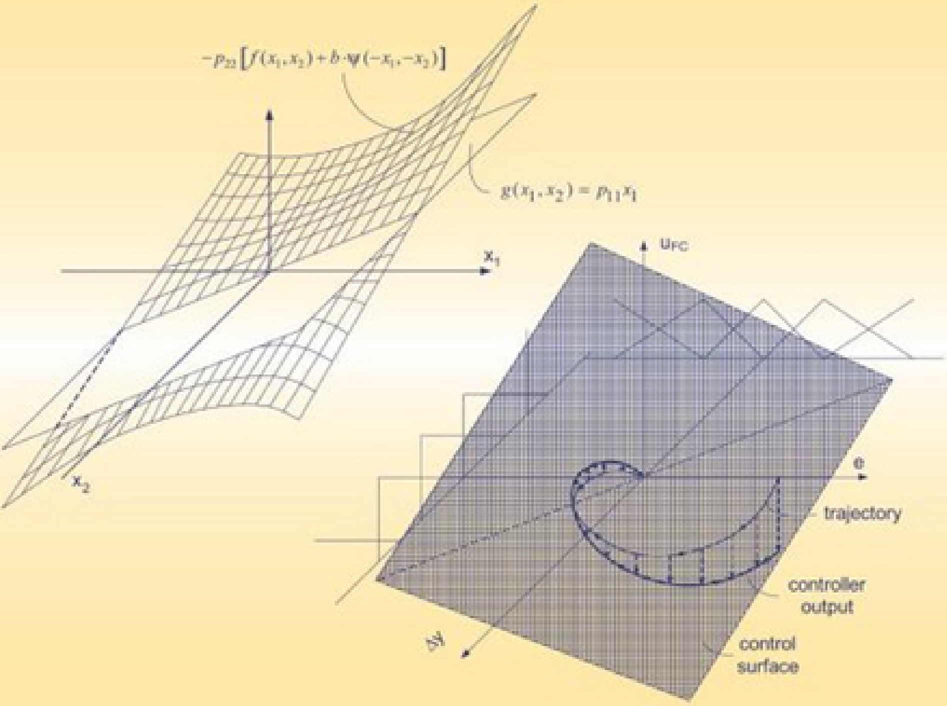
Control Engineering Series

Fuzzy Controller Design

Theory and Applications



Zdenko Kovačić
Stjepan Bogdan

Fuzzy Controller Design

Theory and Applications

Zdenko Kovačić
Faculty of Electrical Engineering and Computing
University of Zagreb
Zagreb, Croatia

Stjepan Bogdan
Faculty of Electrical Engineering and Computing
University of Zagreb
Zagreb, Croatia

CRC Taylor & Francis
Taylor & Francis Group
Boca Raton London New York

A CRC title, part of the Taylor & Francis imprint, a member of the
Taylor & Francis Group, the academic division of T&F Informa plc.

CONTROL ENGINEERING

A Series of Reference Books and Textbooks

Editor
FRANK L. LEWIS, PH.D.
Professor
Automation and Robotics Research Institute
University of Texas at Arlington
Fort Worth, USA

1. Nonlinear Control of Electric Machinery, *Darren M. Dawson, Jun Hu, and Timothy C. Burg*
2. Computational Intelligence in Control Engineering, *Robert E. King*
3. Quantitative Feedback Theory: Fundamentals and Applications, *Constantine H. Houpis and Steven J. Rasmussen*
4. Self-Learning Control of Finite Markov Chains, *A. S. Poznyak, K. Najim, and E. Gómez-Ramírez*
5. Robust Control and Filtering for Time-Delay Systems, *Magdi S. Mahmoud*
6. Classical Feedback Control: With MATLAB®, *Boris J. Lurie and Paul J. Enright*
7. Optimal Control of Singularly Perturbed Linear Systems and Applications: High-Accuracy Techniques, *Zoran Gajif and Myo-Taeg Lim*
8. Engineering System Dynamics: A Unified Graph-Centered Approach, *Forbes T. Brown*
9. Advanced Process Identification and Control, *Enso Ikonen and Kaddour Najim*
10. Modern Control Engineering, *P. N. Paraskevopoulos*
11. Sliding Mode Control in Engineering, *edited by Wilfrid Perruquetti and Jean-Pierre Barbot*
12. Actuator Saturation Control, *edited by Vikram Kapila and Karolos M. Grigoriadis*
13. Nonlinear Control Systems, *Zoran Vukić, Ljubomir Kuljača, Dali Donlagič, and Sejid Tesnjak*
14. Linear Control System Analysis & Design: Fifth Edition, *John D'Azzo, Constantine H. Houpis and Stuart Sheldon*
15. Robot Manipulator Control: Theory & Practice, Second Edition, *Frank L. Lewis, Darren M. Dawson, and Chaouki Abdallah*
16. Robust Control System Design: Advanced State Space Techniques, Second Edition, *Chia-Chi Tsui*
17. Differentially Flat Systems, *Hebertt Sira-Ramirez and Sunil Kumar Agrawal*

18. Chaos in Automatic Control, *edited by Wilfrid Perruquetti and Jean-Pierre Barbot*
19. Fuzzy Controller Design: Theory and Applications, *Zdenko Kovačić and Stjepan Bogdan*
20. Quantitative Feedback Theory: Fundamentals and Applications, Second Edition, *Constantine H. Houpis, Steven J. Rasmussen, and Mario Garcia-Sanz*

Published in 2006 by
CRC Press
Taylor & Francis Group
6000 Broken Sound Parkway NW, Suite 300
Boca Raton, FL 33487-2742

No claim to original U.S. Government works
Printed in the United States of America on acid-free paper
10 9 8 7 6 5 4 3 2 1

International Standard Book Number-10: 0-8493-3747-X (Hardcover)
International Standard Book Number-13: 978-0-8493-3747-5 (Hardcover)
Library of Congress Card Number 2005054270

Library of Congress Cataloging-in-Publication Data

Kovacic, Zdenko.
Fuzzy controller design : theory and applications / by Zdenko Kovacic and Stjepan Bogdan.
p. cm. -- (Control engineering ; 19)
Includes bibliographical references and index.
ISBN 0-8493-3747-X
1. Automatic control. 2. Fuzzy systems. 3. Soft computing. I. Bogdan, Stjepan. II. Title. III. Control engineering (Taylor & Francis) ; 19.

TJ213.K655 2005
629.8--dc22 2005054270

informa
Taylor & Francis Group
is the Academic Division of Informa plc.

Visit the Taylor & Francis Web site at
http://www.taylorandfrancis.com

and the CRC Press Web site at
http://www.crcpress.com

Preface

Fuzzy control methods represent a rather new approach to the problems of controlling complex nonlinear systems, the systems whose mathematical model is difficult or impossible to describe, and the systems with multiple inputs and outputs characterized by hardly defined internal interference. It must be said that fuzzy logic control techniques earned respect from the engineering population after numerous applications on technical and nontechnical systems, especially complex systems in industry, economy, and medicine.

Fuzzy logic and the theory of fuzzy sets are the result of a broader comprehension of practical control problems and control actions, performed by human operators, which could not have been correctly interpreted by using classical bivalent logic and conventional methods of automatic control. In the beginning of his globally successful professional career, "the father of fuzzy logic," Professor Lotfi A. Zadeh, affiliated with the University of California at Berkeley, USA, realized that the existing control theory was very limited and that it did not provide real solutions for the above mentioned classes of systems. In the 1960s, Professor Zadeh made an ingenious shift from standard thinking and interpretation and created the fundamentals of a new system control theory, which got full recognition and obtained numerous followers, after almost 20 years of struggle with fuzzy control opponents. Because most of the opponents were Americans, the well-known Latin proverb "*Nemo propheta in patria sua*" proved to be true once again.

We know today that Professor Zadeh has become one of the most popular scientists in the "fin de siecle" period, spreading the idea of "computing with words" at world leading scientific gatherings and institutions. As a frequent flyer, he stopped two times in our homeland Croatia. In 1968 he was in Dubrovnik, which was the venue of the extremely important scientific symposium that gathered leading control scientists from the West and the East for the first time after the Second World War. The authors of this book had the honor of being Professor Zadeh's hosts at the 9th Mediterranean Conference on Control and Automation MED'01 in Dubrovnik in 2001, where he delivered a keynote lecture "From Computing with Numbers to Computing with Words to Computation with Perceptions — A Paradigm Shift." We share deep impressions about these few days with Professor Zadeh, while photographs taken during the event will remain our dearest memories. During a friendly conversation with Professor Zadeh, he unveiled a very interesting detail — his lecture in Dubrovnik in 1968 was his first lecture about fuzzy logic delivered outside of the United States.

The authors of this book have been actively involved in the fuzzy control area for more than a decade. Having rather good connections with local industry and attempting to raise interest for fuzzy control technology, we have found that this technique has not been accepted as readily as it should have due to a heuristic character of the fuzzy controller design; namely, complex nonlinear control problems are usually related to critical engineering applications, of which project managers demand strong guarantees for the stability and functionality of the fuzzy control system, which are sometimes hard to give. We believe that the gap between fuzzy control theory and practice can be resolved by developing fuzzy controller design techniques that are simple enough, easily implemented, and most of all, effective. The purpose of this book is to present the reader with different techniques of fuzzy controller design that can be applied to a wide range of practical engineering applications without much difficulty. This book does not pretend to cover all fuzzy logic control theory but only those fundamentals that are needed to understand the concept and make a successful design. Most of the attention is paid to the design of hybrid, adaptive, and self-learning fuzzy control structures. We explain the strategies of automated fuzzy controller design suitable for offline and online operations. Our intention was to create examples that would give the reader a better insight into the design methodology and the design steps, in particular.

Professor Lotfi A. Zadeh and the authors of this book with their students at the 9th Mediterranean Conference on Control and Automation in Dubrovnik, 2001.

About the Organization of the Book

This book is divided into seven chapters, with contents covering a wide range of fuzzy controller design topics — from a basic introductory level to a professional application-oriented level. After a brief introduction and review of fuzzy logic systems are given in Chapter 1, Chapter 2 describes the basic definitions of fuzzy sets and operators on fuzzy sets. Consequently, we explain the meaning of linguistic variables, fuzzy rules, fuzzy implications, and inference engines. Then we focus on the description of the most commonly used fuzzy controller structure — the double input–single output (DISO) fuzzy controller. We also look into the important issue of fuzzy control system stability. The heuristic character of fuzzy controller design causes difficulties in assessing the stability of a closed-loop system. In Chapter 2, we describe fuzzy controller design methods based on the well-known Lyapunov theory of stability. We present one method that is suitable for systems that may be approximated with a second-order process model. We also describe a fuzzy controller design procedure that exploits geometric properties of state space during the investigation of system's stability. We show the practical value of this method in a particular case when state space is reduced to phase plane (i.e., in the case of second order systems). In addition, we present a fuzzy controller design method based on the concept of fuzzy Lyapunov stability criterion utilizing fuzzy numbers and fuzzy arithmetic. The aim of examples shown in this chapter is to help the reader understand each design procedure better.

Chapter 3 is concerned with the main drawback of standard fuzzy controller design — its heuristic nature, which turns the tuning of fuzzy controllers into a tedious and time-consuming job, even when it is done with specialized development tools. In order to overcome this problem, we describe several easy-to-implement fuzzy controller design methods that are closely related to the synthesis of well-known control concepts and existing controllers: fuzzy emulation of P-I-D control algorithms, model reference-based design, and design by using phase plane trajectories. For better assessment of these methods, we describe their implementation on a laboratory control process and give useful experimental results. These methods can be used for the automated initial setting of a fuzzy controller used in nonlinear inherently stable time varying SISO high-order systems, which can be linearized in a selected operating point. Examples of such systems may often be found in the process industry (e.g., control of temperature, pressure, flow, level, angular speed, and position).

Many practical control systems are nonlinear and work in conditions of continuous process parameter variations, changing operating modes, and in the presence of external disturbances. Control quality that must be achieved usually cannot be maintained at a desired level with standard controllers. Chapter 4 discusses possibilities of using complex fuzzy controller structures, such as hybrid or adaptive control structures, which would be able to keep control quality almost unchanged, regardless of the above mentioned influences. We describe a hybrid fuzzy controller that contains, in addition to a fuzzy controller, other control elements known from classical control practice. In general, hybrid fuzzy controllers exhibit higher robustness to process parameter variations than standard fuzzy controllers. When parameter variations become excessive and even hybrid fuzzy controllers cannot cope with them, then adaptive fuzzy control structures can be useful to solve the problem. We describe several approaches to an adaptive control design, but emphasis is on the design of fuzzy model reference adaptive control (FMRAC) systems. In the first presented adaptive control concept, adaptation is not oriented toward the fuzzy controller itself, but to the parameters of a lead–lag compensator added in series with the fuzzy controller. An integral criteria-based and sensitivity model-based adaptation of lead–lag compensator parameters is explained in detail and illustrated with worked-out design examples. In this chapter we also describe the design of FMRAC algorithms, which have a high speed of adaptation, produce no oscillations in the steady-state, and add an adaptation signal directly to the feedback controller input or adjust the value of feedback controller output. Because every practical fuzzy controller is designed for constrained input and output universes of discourse, the operating range is also constrained. The operating range can be extended by applying a multiple fuzzy rule table-based adaptation technique described as well in Chapter 4. The reader can gain a deeper understanding of fuzzy adaptive control methods by reading the case studies of FMRAC contact force control and FMRAC angular speed control.

Adaptive control strategies that employ fuzzy inverse models and so-called fuzzy adaptation mechanisms are heuristic as they require the user's active participation in the setting of fuzzy control parameters. An alternative to the heuristic approach lies in the automated online setting of fuzzy controller parameters using stable and fast convergent self-organizing (self-learning, self-tuning) procedures. The distinction between classical self-tuning and considered self-learning procedures is that the former depend on the process model and latter do not. In Chapter 5, we describe several model reference-based self-learning concepts with their common and specific features: one based on the direct Lyapunov method, another with a learning mechanism that utilizes a second-order reference model and a polynomial of the model tracking error, and the third one based on the second-order reference model and a sensitivity model relating the changes of the system output with the changes of fuzzy controller parameters. The stability of the first self-organizing fuzzy control system is assessed by applying a direct Lyapunov method, and stability conditions obtained for a selected Lyapunov function are used for determination of the learning coefficient value. We show that self-organization of a fuzzy rule table based on the learning algorithm that exploits a third-degree model tracking error polynomial with "position," "velocity," and "acceleration" components can be synthesized according to the classical Hurwitz stability criteria, provided the user can foresee the maximal range of process gain variations. A model reference-based and a sensitivity model-based learning algorithm make changes to the fuzzy controller parameter vector once in every run of the system. The reason behind the learning algorithm is that a particular fuzzy controller parameter should be changed when its influence on the system response is the highest. We show how the sensitivity model of a DISO fuzzy controller can be derived and how the second-order reference model is used instead of an unknown control process in the learning algorithm. Several worked-out examples applied to demanding nonlinear servo systems demonstrate the effectiveness of self-learning fuzzy control methods. Taking advantage of the automated fuzzy controller self-organization, we describe again a multiple fuzzy rule table-based adaptation technique applied to a selected positioning servo system. We conclude Chapter 5 with a description of a self-learning PD-type fuzzy controller capable of compensating for steady-state errors caused by external disturbances due to the presence of a self-learning integral term added in parallel. We describe how independent learning of the integral gain coefficient is organized and what results were achieved in a practical experiment.

Respecting the fact that MATLAB®+Simulink® is one of the most popular simulation software packages in use worldwide, in Chapter 6 we give several worked-out examples of fuzzy control systems in order to make it possible for readers to test several fuzzy controllers described in the book. They include the hybrid fuzzy controller described in Chapter 4 and two types of self-organizing fuzzy controllers — the model tracking error polynomial-based and sensitivity model-based, both described in Chapter 5. Self-organizing fuzzy controllers are created as CMEX S-functions PSLFLC and SLFLC contained within the respective MATLAB superblocks. Worked-out examples related to actual demo examples from MATLAB show the effects of considered fuzzy control algorithms. Chapter 6 concludes with an example of a MATLAB-based fuzzy controller design project — the simulation model of an electro hydraulic servo system. Main features of the project are the distinct complexity of the problem, having to deal with process nonlinearities, and coping with time-varying process parameters. In order to simplify the reader's work, some guidelines and useful advice are given.

Although applications are discussed throughout the book illustrating the results obtained with methods presented in each chapter, Chapter 7 is fully dedicated to industrial applications, particularly to different techniques of fuzzy controller implementation and different implementation platforms for industrial applications. Instead of describing many applications, an impossible mission, we focus on generic fuzzy controller implementation concepts, which can help the reader to make his or her future fuzzy control designs and implementations. While describing digital fuzzy controller implementations on the most often used platforms such as microcomputers, programmable logic controllers, and industrial PCs, we also describe a few selected applications in more detail to show the versatility of fuzzy control solutions — from the road tunnel ventilation system to the control of anesthesia carried out during demanding surgical operations.

Every chapter ends with a selected list of references related to the chapter's subject. For easier navigation to subjects, an index is added at the end of the book.

Many individuals have contributed to this book. We are indebted to the students who contributed by performing some of the MATLAB simulation and practical experiments while doing their student projects or working on their diploma and master theses. This list includes, in particular, Dr. Mario Balenović, Tomislav Reichenbach, Krešimir Petrinec, Mario Punčec, and Bruno Birgmajer. A credit for technical support during implementation of the industrial PLC-based adaptive fuzzy condensate level controller goes to Dubravko Lukačević, at that time the manager of the thermal power plant, Jertovec, Croatia. Our special thanks to Dr. Olaf Simanski from the University of Rostock, Germany, who made a valuable contribution to the book by describing an anesthesia control system that employs fuzzy control for controlling the depth of hypnosis of a patient undergoing a demanding surgical operation.

We would like to thank Professor Frank L. Lewis from the University of Texas, Arlington, U.S.A., who gave us initial support about the idea of writing a book. We would also like to thank Professor Robert E. King for his open-minded and witty comments about the contents, style, and other important things during the writing of this book.

Last, but not least, we would like to thank our dear families, especially our wives Dubravka and Jasenka, for their continuous support and encouragement to finish this book.

Authors

Zdenko Kovačić earned his Ph.D.E.E. in 1993, M.S.E.E. in 1987, and B.S.E.E. in 1981 at the University of Zagreb, Croatia. He is currently an associate professor and head of the Department of Control and Computer Engineering at the Faculty of Electrical Engineering and Computing, University of Zagreb. His areas of interest are robotics, flexible manufacturing systems, intelligent, adaptive, and optimal control, and artificial intelligence in control. During 1990–1991, he spent one year working as a researcher in the motion control laboratory of Professor Krishnan Ramu at the Bradley Department of Electrical Engineering, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, U.S.A. He is a principal investigator and project leader of several R&D projects funded by international and Croatian industry partners as well as by the Croatian government. He is a coauthor of the book *Fundamentals of Robotics* (in Croatian). He has been an author and coauthor of numerous papers published in books, journals, or presented at international and national conferences. He is a member of IEEE and KoREMA — Croatian Society for Communication, Computer, Electronics, Measurement, and Control. He is also the founder and vice-president of the Croatian Robotics Society.

Stjepan Bogdan earned his Ph.D.E.E. in 1999, M.S.E.E. in 1993, and B.S.E.E. in 1990 at the University of Zagreb, Croatia. He is currently an assistant professor at the Faculty of Electrical Engineering and Computing, University of Zagreb. His areas of interest are robotics, flexible manufacturing systems (FMS), discrete event systems, intelligent control, adaptive and time optimal control, and control of electrical drives. He was awarded a Fulbright scholarship for the year 1996/97 and worked as a researcher in the Automation & Robotics Research Institute, University of Texas, Arlington, U.S.A. with the research group of Professor Frank L. Lewis. He is a principal investigator and project leader of several projects funded by industry and government. He is a coauthor of the book *Fundamentals of Robotics* (in Croatian). He has been a coauthor of numerous papers published in journals and presented at the national and international conferences. He is a member of KoREMA, Croatian Robotics Society, IEEE, and Sigma Xi.

Contents

1 Introduction

The ability of a human being to find solutions for particular problematic situations is called *human intelligence*. It is founded on the ability of symbolic (exact and abstract) expression of thoughts and interpretation of sensory stimuli in the form of movement, speech, writing, or pictures. We know from our experience that humans have the ability to simultaneously process a large amount of information and make effective decisions, although neither input information nor consequent actions are precisely (firmly) defined. Our experience tells us that the level of knowledge and gained experience has a large impact on the actual success of human actions. Human thinking and decision making mechanisms represent a perfect model, which scientists and engineers attempt to imitate and transform into practical solutions of diverse technical and nontechnical problems. The results of striving for such development are numerous procedures called *artificial intelligence methods*. For example, artificial sight and hearing are based on the use and processing of information from cameras and microphones — that is, from technical devices whose functionality matches the human sensory organs — that is, eyes and ears. We can also include algorithms, which contain elements of the human way of thinking and problem solving, such as *artificial neural networks*, *fuzzy logic algorithms*, *evolutionary* or *genetic algorithms*, and *expert systems*, into the basic forms of artificial intelligence.

Fuzzy control emerged on the foundations of Zadeh's fuzzy set theory [1]. It is a methodology of intelligent control that mimics human thinking and reacting by using a multivalent fuzzy logic and elements of artificial intelligence (simplified deduction principles) [2]. The word "fuzzy" is used here to describe terms that are either not well-known or not clear enough, or their closer specification depends on subjectivity, estimation, and even the intuition of the person who is describing these terms. In everyday life there are a lot of situations characterized by a certain degree of ambiguity whose description includes terms and expressions such as *majority*, *many*, *several*, *not exactly*, or *quite possible*, all of which can be qualified as "fuzzy terms." On the other hand, terms like *false*, *true*, *possible*, *necessary*, *none*, or *all* reflect crisp meanings, and in such a context, represent "exact terms."

The fuzzy logic concept had very strong opponents in the beginning. They believed that any form of vagueness or imprecision could be equally well described with the theory of probability. Furthermore, opponents claimed that fuzzy logic theory was only a theory without real potential for practical applications. In the field of automatic control, the strongest opponents assumed that traditional control techniques were superior to fuzzy logic or at least equal in effect. In one of his interviews, Zadeh commented that the process of accepting fuzzy logic as a method of solving technical problems would require long-term education and a change in the basic approach to scientific analysis and engineering design [3].

These days quite a few engineers know very well that by replacing normally used numbers and sets with fuzzy numbers and sets, every theory can become fuzzy. In that sense, the classical theories of numbers and sets we know and use today are just a boundary form of theories based on fuzzy numbers and sets. For example, in control literature one may find a representation of a classical proportional-integral (PI) controller as a boundary form of a fuzzy controller [4–6], and some classical control methods are analyzed by means of fuzzy logic [7].

While attempting to describe some system, simple or not, one must face the fact that all possible events or phenomena in the system cannot be identified. Even if one would be able to do it, a problem could still remain: how often does a particular event occur in the system? Incomplete knowledge of events and unpredictable frequency of their occurrences impose the usage of approximate system models. In control system theory there are excellent tools for approximate modeling of systems and for design of analytically founded control algorithms. For the systems, which can be well-described with a linear second-order model there are a number of procedures for design of PI and PID controllers, while for the systems modeled with high-order linear models one can use, for example, pole-placement design methods or methods carried out in the frequency domain. Also, we can derive control algorithms by optimizing some criterion (e.g., integral criterion, minimum of variance, etc.). In general, the better the match-up of a process and a model, the better the response of a system controlled by control algorithms designed upon an approximate system model.

The problem arises when the model of a system is unknown or when it is known, but so complex that the design of a controller by using classic analytical methods would be totally impractical. There are also situations when the model of a system is highly nonlinear and where variations of parameters and rates of parameter changes may be extremely high. Some of these situations can be solved by using adaptive control methods [8–11], but their basic mathematical apparatus is rather complex and very often ends in a large number of computing iterations. Although adaptive control schemes using a reference model and signal adaptation act instantaneously (in the first iteration), only simplifications and modifications make their application possible in practice.

A special class of control problems is control of highly nonlinear processes that are exposed to strong influence of external disturbances. With such systems, the only remaining solution is actually carried out in practice: such systems are controlled by operators using their years-long experience and knowledge about static and dynamic characteristics of the system. The achieved quality of control is usually proportional to the operators' knowledge and experience. The operator's experience is connected to monitoring of relevant process variables, and depending on their states and deviations from reference values, operators decide where, how, and how much they need to act on the process to achieve a given control goal. In other words, they execute their "program" or "control algorithm" according to their experience and by applying the following typical pattern of decision making:

IF
such and such states of process variables (i.e., inputs) are
THEN
such and such control actions (i.e., outputs) are needed.

Understanding such a type of control is very easy since there are many examples from everyday life, such as driving a car, where such a control pattern is regularly applied. Let us suppose a situation where the driver of a car has just started overtaking the car in front of him, while another vehicle still far away is approaching from the opposite direction. Among many possible driver's actions, let us mention just a few:

Rule 1 (R^1)

IF
the car that is approaching is far away
AND
if the car in front of the driver's car is driving very slowly
THEN
speed up moderately and pass the car in front.

Rule 2 (R^2)

IF
the car that is approaching is far away
AND
if the car in front of the driver's car is driving at normal speed
THEN
speed up greatly and pass the car in front.

Rule 3 (R^3)

IF
the car that is approaching is far away
AND
the speed of the car that is approaching is high
AND
if the car in front of the driver's car is driving at normal speed
THEN
give up overtaking the car in front and continue driving at the same speed.

From this example we may conclude that a driver of a car must process a great amount of input information simultaneously and make effective decisions, although neither input information (distance, speed) nor decided actions are actually precisely (crisply) determined. Linguistic qualifications *far away*, *high speed*, *normal speed*, *speed up moderately*, *speed up greatly* met in rules R^1, R^2, and R^3 will vary in interpretation from driver to driver, but nevertheless, they are very effective in practice and car collisions, considering the number of cars driving at the same time, are very rare. This proves that our human actions are based on very effective action provocation mechanisms that depend exclusively on imprecise linguistic qualifications of causes and consequences expressed in the form of very simple to very complex action rules.

The main problem a control designer is confronted with is how to find a formal way to convert the knowledge and experience of a system operator into a well-designed control algorithm. By using multivalent fuzzy logic, linguistic expressions in antecedent and consequent parts of IF–THEN rules describing the operator's actions can be efficaciously converted into a fully-structured control algorithm suitable for microcomputer implementation or implementation with specially designed analog (digital) fuzzy processors [12–16]. IF–THEN rules themselves have already been used in bivalent logic control concepts, but such gradation of the truth (fulfilment) of antecedent and consequent parts of IF–THEN rules is a qualitatively new moment brought about by the introduction of fuzzy logic.

Due to the fact that a fuzzy algorithm has the characteristics of a universal approximator, a designer is able to model (identify) an unknown process with a set of IF–THEN fuzzy rules [17–20], which makes possible the introduction of feed-forward control elements for fuzzy model-based prediction of future system states [21]. There are also situations when operators are not able to express the rules of how they are conducting the system (usually they would say — by a "feeling"). In that case the way out is to identify the control actions and describe them by using fuzzy rules [22].

It must be noted that application of fuzzy logic is not limited only to systems difficult for modeling [23–29]. By application of fuzzy logic on systems with known, but complex mathematical models, the time needed for controller design and for practical application can be significantly shortened, sometimes up to ten times [30], although the improvement in achieved control quality may not always be so high.

The nonlinear character of a fuzzy controller may contribute to a higher robustness of systems, which contain nonlinear elements but otherwise have a simple structure from the control point of view.

Besides having the role of the principal controller in a control loop, fuzzy logic algorithms can be equally well used in adaptive control schemes, performing different tasks like tuning parameters of conventional controllers [31–35] or working in parallel with other methods of intelligent control like genetic algorithms [36] or artificial neural networks [37–42]. A very informative review of neuro-fuzzy control structures can be found in Reference 43. What makes fuzzy logic algorithms attractive for applications is also the fact that the level of knowledge needed for their design does not have to be as high as it usually must be for the design of a conventional controller controlling a very complex system.

REFERENCES

1. Zadeh, L.A., "Fuzzy sets," *Information Control*, 8, 94–102, 1965.
2. Chang, S.S.L. and Zadeh, L.A., "On fuzzy mapping and control," *IEEE Transactions on Systems, Man and Cybernetics*, SMC-2, 30–34, 1972.
3. Zadeh, L.A., "Making computers think like people," *IEEE Spectrum*, 26–32, August 1984.
4. Brehm, T. and Rattan, K.S., "The classical controller: a special case of the fuzzy logic controller," in *Proceedings of the 33rd Conference on Decision and Control*, Lake Buena Vista, pp. 4128–4129, 1994.
5. Ying, H., Siler, W., and Buckley, J.J., "Fuzzy control theory: a nonlinear case," *Automatica*, 26, 513–520, 1990.
6. Tang, K.L. and Mulholland, R.J., "Comparing fuzzy logic with classical controller design," *IEEE Transactions on Systems, Man and Cybernetics*, 17, 1085–1087, 1987.
7. Tanaka, H., Uejima, S., and Asai, K., "Linear regression analysis with fuzzy model," *IEEE Transactions on Systems, Man and Cybernetics*, 12, 903–906, 1982.
8. Butler, H., *Model Reference Adaptive Control — From Theory to Practice*, Prentice Hall, New York, 1992.
9. Chalam, V.V., *Adaptive Control Systems — Techniques and Applications*, Marcel Dekker, Inc., New York and Basel, 1987.
10. Landau, Y.D., *Adaptive Control — The Model Reference Approach*, Marcel Dekker Inc., New York, 1979.
11. Kaufman, H., Bar-Kana, I., and Sobel, K., *Direct Adaptive Control Algorithms: Theory and Applications*, Springer-Verlag, New York, 1994.
12. Omron, *Clearly Fuzzy*, Omron Corporation, Tokyo, Japan, 1991.
13. Patyra, M.J., Grantner, J.L., and Koster, K., "Digital fuzzy logic controller: design and implementation," *IEEE Transactions on Fuzzy Systems*, 4, 439–459, 1996.
14. Guo, S., Peters, L., and Surmann, H., "Design and application of an analog fuzzy logic controller," *IEEE Transaction on Fuzzy Systems*, 4, 429–438, 1996.
15. Hung, D.L., "Dedicated digital fuzzy hardware," *IEEE Micro*, 15, 31–39, August 1995.
16. Nakamura, K., Sakashita, N., Nitta, Y., Shimomura, K., and Tokuda, T., "Fuzzy inference and fuzzy inference processor," *IEEE Micro*, 13, 37–48, October 1993.
17. Yoshinari, Y., Pedrycz, W., and Hirota, K., "Construction of fuzzy models through clustering techniques," *Fuzzy Sets and Systems*, 54, 157–165, 1993.
18. Pham, T.D. and Valliappan, S., "Aleast squares model for fuzzy rules of inference," *Fuzzy Sets and Systems*, 64, 207–212, 1994.

19. Yi, S.Y. and Chung, M.J., "Identification of fuzzy relational model and its application to control," *Fuzzy Sets and Systems*, 59, 25–33, 1993.
20. Chen, J.Q., Lu, J.H., and Chen, L.J., "An on-line identification algorithm for fuzzy systems," *Fuzzy Sets and Systems*, 64, 63–72, 1994.
21. Moore, C.G. and Harris, C.J., "Aspects of fuzzy control and estimation," in Harris C.J. (ed.), *Advances in Intelligent Control*, pp. 201–242, 1994.
22. Takagi, T. and Sugeno, M., "Fuzzy identification of systems and its applications to modeling and control," *IEEE Transactions on Systems, Man and Cybernetics*, 15, 116–132, 1985.
23. Maiers, J. and Sherif, Y.S., "Application of fuzzy set theory," *IEEE Transactions on Systems, Man and Cybernetics*, 15, 175–189, 1985.
24. Dutta S., "Fuzzy logic applications: technological and strategic issues," *IEEE Transactions on Engineering Management*, 40, 237–254, 1993.
25. King, P.J. and Mamdani, E.H., "The application of fuzzy control systems to industrial processes," *Automatica*, 13, 235–242, 1977.
26. Kickert, W.J.M. and van Nauta Lemke, H.R., "Application of a fuzzy controller in a warm water plant," *Automatica*, 12, 301–308, 1976.
27. Meier, R., Nieuwland, J., Zbinden, A.M., and Hacisalihzade, S.S., "Fuzzy logic control of blood pressure during anesthesia," *IEEE Control System Magazine*, 12–17, December 1992.
28. Heckenthaler, T. and Engell, S., "Approximately time-optimal fuzzy control of a two-tank system," *IEEE Control Systems Magazine*, 24–30, June 1994.
29. Aliev, R.A., Aliev, F.T., and Babaev, M.D., "The synthesis of a fuzzy coordinate-parametric automatic control system for an oil-refinery unit," *Fuzzy Sets and Systems*, 47, 157–162, 1991.
30. Self, K., "Designing with fuzzy logic," *IEEE Spectrum*, 42–44, November 1990.
31. He, S.Z., Tan, S., Xu, F.L., and Wang, P.Z., "Fuzzy self-tuning of PID controller," *Fuzzy Sets and Systems*, 56, 37–46, 1993.
32. Zhao, Z.Y., Tomizuka, M., and Isaka, S., "Fuzzy gain scheduling of PID controller," *IEEE Transactions on Systems, Man and Cybernetics*, 23, 1392–1398, 1993.
33. Peng, X.T., Liu, S.M., Yamakava, T., Wang, P., and Liu, X., "Self-regulating PID controllers and its applications to a temperature controlling process," in M. Gupta and T. Yamakawa (eds), *Fuzzy Computing: Theory, Hardware, and Applications*, 355–364, 1988.
34. Tseng, H.C. and Hwang, V.H., "Servocontroller tuning with fuzzy logic," *IEEE Transactions on Control Systems Technology*, 1, 262–269, 1993.
35. Stipaničev, D., "Diskretno vođenje složenih sustava adaptivnim nelinearnim PID-regulatorima," *Elektrotehnika*, 34, 153–161, 1991 (in Croatian).
36. Varšek, A., Urbančić, T., and Filipič, B., "Genetic algorithms in controller design and tuning," *IEEE Transactions on Systems, Man and Cybernetics*, 23, 1330–1339, 1993.
37. Vidal-Verdu, F. and Rodriquez-Vazquez, A., "Using building blocks to design analog neuro-fuzzy controllers," *IEEE Micro*, 15, 49–57, August 1995.
38. Mitra, S. and Pal, S.K., "Fuzzy self-organization, inferencing and rule generation," *IEEE Transactions on Systems, Man and Cybernetics*, 26, 608–620, 1996.
39. Lin, C.T. and Lu, Y.C., "A neural fuzzy systems with fuzzy supervised learning," *IEEE Transactions on Systems, Man and Cybernetics*, 26, 744–763, 1996.
40. Pedrycz, W., Poskar, C.H., and Czezowski, P.J., "A reconfigurable fuzzy neural network with in-situ learning," *IEEE Micro*, 15, 19–29, August 1995.

41. Barenji, H.R. and Khedkar, P., "Learning and tuning fuzzy logic controllers through reinforcements," *IEEE Transactions on Neural Networks*, 3, 724–740, 1992.
42. Tönshoff, H.K. and Walter, A., "Self-tuning fuzzy-controller for process control in internal grinding," *Fuzzy Sets and Systems*, 63, 359–373, 1994.
43. Buckley, J.J. and Hayashi, Y., "Fuzzy neural networks: a survey," *Fuzzy Sets and Systems*, 66, 1–13, 1994.

2 Fuzzy Controller Design

In this chapter we describe the basic definitions of fuzzy sets and operators on fuzzy sets. As they are used throughout the book it is necessary to start by introducing basic definitions of terms such as linguistic variables, fuzzy propositions, relations, implications, and inference engines. An emphasis is given to the description of the fuzzy controller structure that is most commonly used in practice, as well as to the several ways of defuzzification, that is, calculation of the crisp controller output value.

2.1 Fuzzy Sets

We will use the following example to help us define a fuzzy set. Let *A* be a set of all integers *greater* than 10. We write

$$A = \{x\colon x \in \aleph, x > 10\} \tag{2.1}$$

Let *B* be a set of all integers *much greater* than 10. Mathematically, this statement can be written as

$$B = \{x\colon x \in \aleph, x \gg 10\} \tag{2.2}$$

The main difference between these two sets is that relation (2.1) completely defines set *A*, while relation (2.2) is not sufficient for a complete definition of set *B*. The reason is the vagueness of the term *much greater*. It is clear that 11, 12, 1178, and 2,075 are elements of set *A*. Most of the people will agree that 11,234 and 23^{10} undoubtedly belong to set *B*, but it is doubtful whether 15 or 50 are elements of *B*. The problem is how to determine the lowest integer which is *much greater* than 10.

This problem can be solved if one uses an alternative way of describing a set. According to traditional set theory, a set can be defined by its *characteristic function*. In other words, instead of individually declaring each element of a set we define a function that can take on values 1 or 0 depending on full membership or no membership of a particular element, respectively.

Definition 2.1 (Characteristic function, crisp set) Let S be a set from the domain X. A *characteristic function* of the set S attains value $\mu_S(x) = 1$ if $x \in S$, and $\mu_S(x) = 0$ if $x \notin S$, $\mu\colon X \to \{0, 1\}$. Set S with its characteristic function is called a *crisp set*.

Defined as it is, the characteristic function cannot describe set *B*, that is, it cannot cope with the vagueness in determining the lowest integer which would

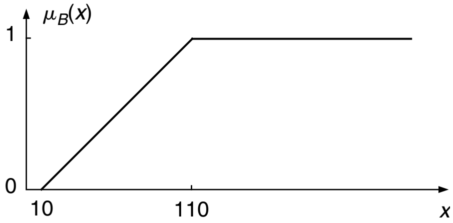


Figure 2.1 A graphical representation of a fuzzy set.

belong to set B. However, broadening the notion of a characteristic function offers an elegant way to define set B. Instead of determining the lowest integer belonging to set B, we may say that *all* integers greater than 10 belong to set B but with a different *membership degree*. The characteristic function, obtaining partial, or graded, values from the interval [0, 1], now becomes a *membership function*.

Definition 2.2 (Membership function and fuzzy set) Let F be a set from the domain X. A membership function $\mu_F(x)$ of set F is a function that assigns value, or *membership degree*, to every $x \in F$, μ: $X \to [0, 1]$. Then set F is called a *fuzzy set*.

Apparently, crisp sets may be treated as a special case of fuzzy sets since the characteristic function can assume only margin values from the interval [0, 1] on which membership function is defined.

Now we can completely define fuzzy set B as a set of pairs:

$$B = \{(\mu_B(x), x): x \in \aleph\}$$

$$\mu_B(x) = \begin{cases} 0, & \text{for } x < 10 \\ \dfrac{x - 10}{100}, & \text{for } 10 \leq x \leq 110 \\ 1, & \text{for } x > 110 \end{cases} \tag{2.3}$$

From the above definition we can see that numbers with membership degree 0 do not belong to fuzzy set B. Number 11 is an element of B with membership degree $\mu_B(11) = 0.01$, while membership degree of number 100 is $\mu_B(100) = 0.9$. Fuzzy set B is pictured in Figure 2.1.

From fuzzy set Definition 2.2, it follows that two fuzzy sets having the same elements will be equal only if their membership functions are equal. This means that each element belonging to one fuzzy set must belong to the other fuzzy set with the same membership degree.

Definition 2.3 (Fuzzy subset) Fuzzy set C is a *fuzzy subset* of fuzzy set B if

$$\forall x \in X: \mu_C(x) \leq \mu_B(x) \tag{2.4}$$

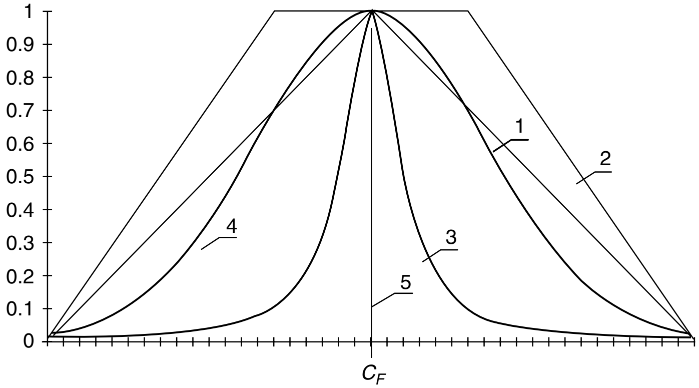


FIGURE 2.2 Typical shapes of membership functions: 1 — triangular, 2 — trapezoidal, 3 — Gaussian, 4 — bell-shaped, 5 — singleton.

In fuzzy sets theory, the range of possible quantitative values considered for fuzzy set members is called *universe of discourse*. Universe of discourse can be continuous or discrete. Discrete universe of discourse is normally bounded and contains a finite number of elements. A fuzzy set with discrete universe of discourse is called *a discrete fuzzy set*. The measure of fuzziness of each element is determined using a membership function spread either over a part or over the entire universe of discourse. As we have stated, the membership function converts the degree of fuzziness into the normalized interval [0, 1] where the boundary values 0 and 1 resemble the membership degrees of crisp set members. Membership functions can attain different forms. However, triangular, trapezoidal, Gaussian, and bell-shaped forms, shown in Figure 2.2, are used more than others:

$$\mu_F(x) = \begin{cases} 0, & \text{for } x < a \\ \dfrac{x-a}{b-a}, & \text{for } a \le x < b \\ \dfrac{c-x}{c-b}, & \text{for } b \le x \le c \\ 0, & \text{for } x > c \end{cases} \qquad \text{triangular}$$

$$\mu_F(x) = \begin{cases} 0, & \text{for } x < a \\ \dfrac{x-a}{b-a}, & \text{for } a \le x < b \\ 1, & \text{for } b \le x < c \\ \dfrac{d-x}{d-c}, & \text{for } c \le x \le d \\ 0, & \text{for } x > d \end{cases} \qquad \text{trapezoidal}$$

$$\mu_F(x) = \mathrm{e}^{-(x-c_F)^2/w} \quad \text{Gaussian,} \qquad \mu_F(x) = \frac{1}{1+(x-c_F)^2} \quad \text{bell-shaped}$$

These fuzzy sets are defined for variable x. Anticipating that different variables, for example, x and y, may have fuzzy sets with identical names (indices), it is convenient to introduce modified membership function notation $\mu_F^x = \mu_F(x)$. Such notation can, for example, discern notation for $\mu_F^y = \mu_F(y)$. This allows further generalizations, such as $\mu_i^x = \mu_i(x), i = 1, \ldots, l$.

Definition 2.4 (Center and nucleus of fuzzy set) A singular value $x = c_F = c_F^x \in F$ with the maximum degree of membership, $\mu_F(c_F) = 1$, is called the *center* of fuzzy set F. If there exists a set of values with the maximum degree of membership,

$$\text{nuc}(F) = \{x \in X\colon \mu_F(x) = 1\} \tag{2.5}$$

then nuc(F) is called the *nucleus* of fuzzy set F.

The center of fuzzy set F with a nucleus is defined as $c_F^x = (x_a + x_b)/2$; where x_a and x_b are the boundaries of the nucleus. Fuzzy set F, having the center c_F^x as its only element, is a *fuzzy singleton* (Figure 2.2). In case we would like to replace a typical fuzzy set with a singleton (which is often done in the case of output fuzzy sets of fuzzy controllers), we must decide which value is representative and unique for different shapes of fuzzy sets, in order to be able to make such a replacement. Due to the fact that fuzzy sets can be either uniform or not, symmetrical or not, bounded or not — a good measure of their geometrical shape is the *centroid*, the point within a fuzzy set designating its center of gravity (COG). For example, the centroid of a triangular fuzzy set is the concurrence point of its three medians.

For a symmetrical fuzzy set the projection of its centroid on universe of discourse is equal to the center of the fuzzy set. That is why the center c_F^x is often referred to as the centroid. Fuzzy singletons are frequently used in fuzzy controller design since they reduce calculation efforts and provide shorter control intervals, which are desirable features in real-time control applications. We shall return to this issue in the chapters that follow.

Definition 2.5 (Adjacent fuzzy sets) Let two fuzzy sets, F and T, with centers c_F^x and c_T^x, $c_F^x < c_T^x$, be defined from the same universe of discourse X. If there does not exist a fuzzy set S, defined from X, with the center c_S^x, such that $c_F^x < c_S^x < c_T^x$, then F and T are called *adjacent fuzzy sets*.

The importance of adjacency will become apparent in the next paragraph where we describe properties of a set of rules formed by fuzzy sets.

Operations *union*, *intersection*, and *complement* are strictly defined on crisp sets. They are unambiguous because statements in traditional set theory are formed by *and*, *or*, and *not* operators which have well-defined semantics. In traditional set theory statement "$x \in B$ and $x \in C$" is true only if both declarations are true. In other words, a new set can be formed from sets B and C and x will belong to this new set only in case it is an element of both set B and set C. In fuzzy set theory the interpretation of statement "($x \in B$: $\mu_B(x) = 0.1$) and ($x \in C$: $\mu_C(x) = 0.3$)"

is not that simple. Namely, it is not apparent how to determine the membership degree of x in the new fuzzy set formed by fuzzy sets B and C.

There are many different suggestions for determining the membership function of a fuzzy set that is the result of union, intersection, and complement of other fuzzy sets. Zadeh proposed the following definitions of these operations:

$$\begin{aligned} \mu_{B\cap C}(x) &= \min(\mu_B(x), \mu_C(x)) \\ \mu_{B\cup C}(x) &= \max(\mu_B(x), \mu_C(x)) \\ \mu_{\bar{B}}(x) &= 1 - \mu_B(x) \end{aligned} \tag{2.6}$$

According to (2.6), statement "($x \in B$: $\mu_B(x) = 0.1$) and ($x \in C$: $\mu_C(x) = 0.3$)" form a new fuzzy set $D = B \cap C$ with $\mu_D(x) = 0.1$.

It should be noted that the above definitions are also valid for crisp sets. If a membership function is replaced with a characteristic function that obtains only two values, 0 and 1, then Zadeh's operators will give the same results as standard *and*, *or*, and *not* operators.

In general, operators on fuzzy sets use *triangular norms* — a class of binary functions, which may be divided into *T-norms* (AND operators) and *S-norms* (OR operators) [1,2]. T-norms perform an *intersection* operation on fuzzy sets and have a particular importance in fuzzy logic control. T-norm is usually denoted as $T(a, b)$. S-norms represent a *union* operation denoted as $S(a, b)$.

Almost all T-norms used in fuzzy control applications can be derived from four basic T-norms listed below:

$$\begin{aligned} &1.\quad T(\mu_B, \mu_C) = \min(\mu_B, \mu_C) \\ &2.\quad T(\mu_B, \mu_C) = \mu_B \cdot \mu_C \\ &3.\quad T(\mu_B, \mu_C) = \max(0, \mu_B + \mu_C - 1) \\ &4.\quad T(\mu_B, \mu_C) = \begin{cases} \mu_B, & \text{if } \mu_C = 1 \\ \mu_C, & \text{if } \mu_B = 1 \\ 0, & \text{if } \mu_B, \mu_C < 1 \end{cases} \end{aligned} \tag{2.7}$$

The differences between these four T-norms are shown in the following example.

Example 2.1 Four basic T-norms.

We will now consider two fuzzy sets, B and C, which are defined as:

$$B = \{(\mu_B(x), x)\colon x \in \aleph\}, \qquad \mu_B(x) = \begin{cases} 0.1 \cdot x, & \text{for } x < 10 \\ \dfrac{20 - x}{10}, & \text{for } 10 \le x \le 19 \\ 0, & \text{for } x > 19 \end{cases}$$

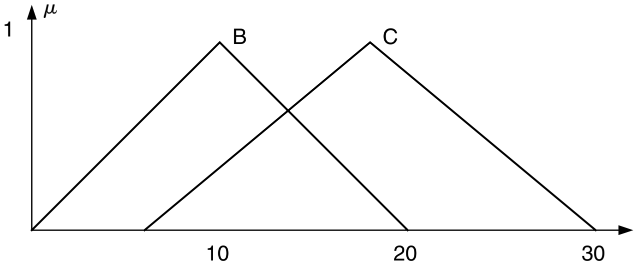


FIGURE 2.3 Fuzzy sets B and C considered in Example 2.1.

$$C = \{(\mu_C(x), x)\colon x \in \aleph\} \qquad \mu_C(x) = \begin{cases} 0, & \text{for } x < 7 \\ \dfrac{x-6}{12}, & \text{for } 7 \le x \le 18 \\ \dfrac{30-x}{12}, & \text{for } 19 \le x \le 29 \\ 0, & \text{for } x > 29 \end{cases}$$

Fuzzy sets B and C are pictured in Figure 2.3. Numerical values of the corresponding membership functions and results obtained after the application of four basic T-norms are given in Table 2.1.

Membership functions calculated in Table 2.1 for four different T-norms are represented graphically in Figure 2.4.

2.2 Linguistic Variables

In everyday communication we often use short sentences, which carry the same amount of information as their longer counterparts. When we say that "the car is far away" we actually mean that "the car's distance belongs to the *far away* (*very long*) category." Even if we knew that the distance was exactly 350 m, in everyday communication we would prefer saying that "the car is far away," as we would assume that there is a common understanding what a *very long distance* in traffic terms is. The term *distance* may attain two different values: numerical (350 m) and linguistic (far away). Variables, for which values are words or sentences, rather than numbers, are called *linguistic variables*. Continuing with our car analogy — the driver makes decisions about her/his actions based on imprecise linguistic qualifications of input information. Therefore, the strategy of driving a car is expressed in the form of IF–THEN rules, which contain linguistic variables: usually the names of inputs and outputs, rather than their concrete values (numbers). As Zadeh has stated, linguistic variables may assume different linguistic values over a specified universe of discourse. This means that linguistic values defined by an appropriate

TABLE 2.1
Numerical Values of Membership Functions Obtained after Application of Four Basic *T*-Norms from (Equation 2.7) on Fuzzy Sets *B* and *C*

x	$\mu_B(x)$	$\mu_C(x)$	$\min(\mu_B, \mu_C)$	$\mu_B \cdot \mu_C$	$\max(0, \mu_B + \mu_C - 1)$	T-norm 4
0	0	0	0	0	0	0
1	0.100	0	0	0	0	0
2	0.200	0	0	0	0	0
3	0.300	0	0	0	0	0
4	0.400	0	0	0	0	0
5	0.500	0	0	0	0	0
6	0.600	0	0	0	0	0
7	0.700	0.083	0.083	0.058	0	0
8	0.800	0.167	0.167	0.133	0	0
9	0.900	0.250	0.250	0.225	0.150	0
10	1.000	0.333	0.333	0.333	0.333	0.333
11	0.900	0.417	0.417	0.375	0.317	0
12	0.800	0.500	0.500	0.400	0.300	0
13	0.700	0.583	0.583	0.408	0.283	0
14	0.600	0.667	0.600	0.400	0.267	0
15	0.500	0.750	0.500	0.375	0.250	0
16	0.400	0.833	0.400	0.333	0.233	0
17	0.300	0.917	0.300	0.275	0.217	0
18	0.200	1.000	0.200	0.200	0.200	0.200
19	0.100	0.917	0.100	0.092	0.017	0
20	0	0.833	0	0	0	0
21	0	0.750	0	0	0	0
22	0	0.667	0	0	0	0
23	0	0.583	0	0	0	0
24	0	0.500	0	0	0	0
25	0	0.417	0	0	0	0
26	0	0.333	0	0	0	0
27	0	0.250	0	0	0	0
28	0	0.167	0	0	0	0
29	0	0.083	0	0	0	0
30	0	0	0	0	0	0

semantic rule represent nothing but informative attributes about the physical values in a part of a specified universe of discourse.

A linguistic variable can be noted in this way:

$$[x, T, X, M] \tag{2.8}$$

where x is the name of a linguistic variable, $T = \{T_i\}$ is the set of linguistic values which x may attain, $i = 1, \ldots, l, X$ is the quantitative universe of discourse

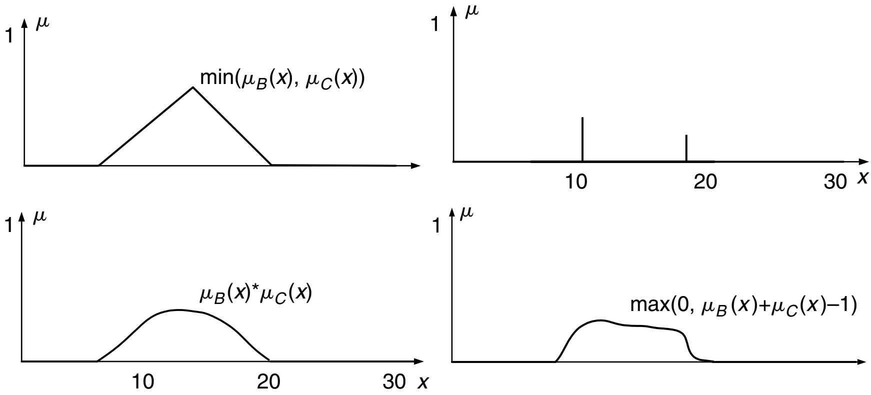


FIGURE 2.4 T-norms calculated in Example 2.1.

(continuous or discrete) of x, M is the semantic function which associates linguistic values in T with the universe of discourse X.

Semantic function M basically describes the distribution of fuzzy sets which represent linguistic values of the variable x over a range of numerical values that x may attain. In our example, the variable *distance* may have linguistic values such as *very long* (*far away*), *long*, *short*, *close*, and *very close*. These values may be associated with completely different physical (numerical) values. For example, the same linguistic value can be used for the distances of 5 km (e.g., in aircraft control), 10 m (e.g., in car control), or 3 cm (e.g., in servo control).

Definition 2.6 (Fuzzy proposition) Let $x \in X$ be a linguistic variable and $T_i(x)$ be a fuzzy set associated with a linguistic value T_i. Then the structure

$$P_i\colon x \text{ is } T_i \tag{2.9}$$

written in modified notation also as P_i^x: x is T_i, represents a *fuzzy proposition.*

A fuzzy proposition is interpreted by a process known as fuzzification.

Definition 2.7 (Fuzzification) Let $x \in X$ be a linguistic variable and $T_i(x)$ be a fuzzy set associated with a linguistic value T_i. The conversion of a physical (numerical) value of x into a corresponding linguistic value by associating a membership degree, $x \to \mu_{T_i}(x)$ is called *fuzzification.* The membership degree $\mu_{T_i}(x)$ represents the fuzzy equivalent of the value of x.

The definitions of a linguistic variable, as well as definitions of a fuzzy proposition and fuzzification are illustrated with the following example.

Example 2.2 Linguistic variable, fuzzy proposition, and fuzzification.

Suppose that a fuzzy logic controller input takes values from 0 to 100. Then a possible definition of a fuzzy controller input as a linguistic variable could be:

$$x\text{: controller input}, T\text{: \{large, medium, small, zero\}}, X\text{: } [0, 100], \text{ and } M\text{: } X \to T$$

defined as:

$$\text{large} = \{(\mu_L(x), x) \mid x \in X\}, \qquad \text{medium} = \{(\mu_M(x), x) \mid x \in X\}$$
$$\text{small} = \{(\mu_S(x), x) \mid x \in X\}, \qquad zero = \{(\mu_Z(x), x) \mid x \in X\}$$

$$\mu_L(x) = \begin{cases} 0, & \text{for } x < 70 \\ \dfrac{x-70}{20}, & \text{for } 70 \leq x \leq 90 \\ 1, & \text{for } x > 90 \end{cases} \qquad \mu_M(x) = \begin{cases} 0, & \text{for } x < 30 \\ \dfrac{x-30}{20}, & \text{for } 30 \leq x < 50 \\ \dfrac{75-x}{25}, & \text{for } 50 \leq x \leq 75 \\ 0, & \text{for } x > 75 \end{cases}$$

$$\mu_S(x) = \begin{cases} 0, & \text{for } x < 15 \\ \dfrac{x-15}{15}, & \text{for } 15 \leq x < 30 \\ \dfrac{45-x}{15}, & \text{for } 30 \leq x \leq 45 \\ 0, & \text{for } x > 45 \end{cases} \qquad \mu_Z(x) = \begin{cases} 1, & \text{for } x < 5 \\ \dfrac{20-x}{15}, & \text{for } 5 \leq x \leq 20 \\ 0, & \text{for } x > 20 \end{cases}$$

According to semantic function M, we can express the fact that the numerical value of the “controller input” is equal to 55 using a fuzzy proposition

controller input is medium

The fuzzy equivalent of the value 55 is obtained by fuzzification, that is, by inclusion of 55 in relation that describes the membership function of the fuzzy set *medium*(x):

$$\mu_M(55) = \frac{75-55}{25} \quad \text{for } 50 \leq x \leq 75 = 0.8$$

Fuzzy propositions are the building blocks of a fuzzy controller. They are elements used for description of someone’s experience or knowledge. Very often, two or more fuzzy propositions are put in relation (in case of multiple input-multiple output controller configurations) to describe more complex knowledge about process control.

Definition 2.8 (Fuzzy relation) Let $x \in X$ and $y \in Y$ be linguistic variables, and $T_i(x)$ and $F_j(y)$ be fuzzy sets corresponding to linguistic values T_i and F_j, respectively. Then the structure

$$\begin{aligned} &R_{ij}\colon x \text{ is } T_i \quad \wp \quad y \text{ is } F_j \\ &\text{denoted as} \\ &R_{ij}\colon P_i^x \quad \wp \quad P_j^y \end{aligned} \tag{2.10}$$

represents a two-dimensional *fuzzy relation* where $\wp$ represents an operator.

It should be noted that the selection of $\wp$ directly influences the structure of the designed fuzzy controller [3,4].

If $\wp$ were the classical AND operator and the propositions in (2.10) had crisp sets (with only *true* or *false* states), the relation would be true only if both propositions were true. On the other hand, the degree to which fuzzy relation (2.10) is true depends on the operator $\wp$ and the degree of each proposition is determined by the membership functions $\mu_{T_i}(x)$ and $\mu_{F_i}(x)$. This implies the existence of a fuzzy relation membership function. Accordingly, fuzzy relation (2.10) can be noted in the following way:

$$\begin{aligned} R_{ij} &= \{[\mu_{R_{ij}}(x, y), x, y] \mid x \in X, y \in Y\} \\ \mu_{R_{ij}}(x, y) &= \wp\{\mu_{T_i}(x), \mu_{F_j}(y)\} \end{aligned} \tag{2.11}$$

where $\mu_{R_{ij}}(x, y)$: $[0, 1] \times [0, 1] \rightarrow [0, 1]$.

Two-dimensional fuzzy relations are actually two-dimensional fuzzy sets which can be graphically depicted for $T_i(x) = T_1(x), F_j(y) = F_1(y)$, as shown in Figure 2.5, where a T-norm min is applied and triangular shapes of membership functions are selected. It may be noted that in this case the membership function $\mu_{R_{11}}(x, y)$ represents the surface, which creates a pyramid with the x–y plane.

2.3 Fuzzy Rules

We have already mentioned that the goal of fuzzy controllers is to mimic a human operator's actions or to make humanlike decisions by using the knowledge about controlling a target system (without knowing its model). This is achieved with *fuzzy rules* that constitute a *fuzzy rule base*. The fuzzy rule base is a central component of the fuzzy controller and it represents the "intelligence" in any fuzzy control algorithm [5–7]. This is the place where the designer's knowledge and experience must be correctly interpreted and organized into an appropriate set of rules.

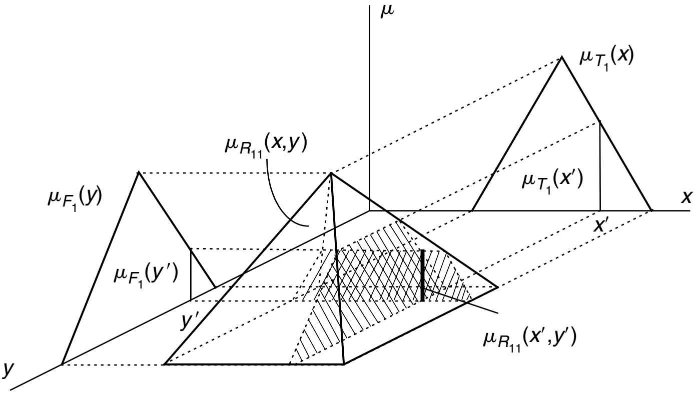


FIGURE 2.5 A graphical interpretation of a two-dimensional fuzzy relation.

Definition 2.9 (Fuzzy rule) Let A and B be either fuzzy relations or fuzzy propositions. Then the structure

$$\text{FR:} \quad \text{IF } A \text{ THEN } B \tag{2.12}$$

is called a *fuzzy rule.*

As we can see, every fuzzy rule can be divided into an antecedent part (IF . . .) and a consequent part (THEN . . .), with antecedent parts describing causes and consequent parts describing consequences relevant for control action [8]. Such a form of fuzzy rules enables nonlinear mapping of inputs and outputs and thus enables creation of versatile *static nonlinear control functions*. The nonlinear character of these functions allows the fuzzy logic controller to cope successfully with complex nonlinear control problems.

In case of dealing with the most frequently used two-input one-output fuzzy controller fuzzy rules that compose the fuzzy rule base have the form of "IF *relation* THEN *proposition*," which corresponds to the general form (2.12).

The organization of a fuzzy rule base is normally considered to be the most demanding step in the process of fuzzy controller design. When we consider the other parts of the fuzzy controller, we may say that they are only a service to the fuzzy rule base. Besides, the number of input fuzzy sets and the shape of their membership functions, the way how they are distributed along the universe of discourse and finally, the choice of a procedure for calculation of the fuzzy controller output have less influence on the fuzzy control algorithm than the rule base itself.

The size of the fuzzy rule base depends on the number of fuzzy rules, while the number of fuzzy rules depends on the number of input and output variables and on the number of linguistic values (fuzzy sets) associated with each of the variables. The number of fuzzy rules will decrease if the knowledge base about process control is incomplete and some fuzzy rules stay undefined.

In general, the formation of fuzzy rules must follow some common sense in order to preserve basic fuzzy rule base characteristics such as *consistency* (*contradiction*), *continuity*, and *completeness*.

Definition 2.10 (Consistency of fuzzy rule base) A fuzzy rule base is *consistent* if it does not include rules FR^i and FR^j such that

$$\mathrm{FR}^i\text{:}\quad \text{IF } R_{pq} \text{ THEN } P_m$$

$$\mathrm{FR}^j\text{:}\quad \text{IF } R_{pq} \text{ THEN } P_n$$

that is, there are no rules, which have equal antecedent parts and different consequent parts. A designer should take care that fuzzy rules do not become contradictory.

Prior to defining continuity of a fuzzy rule base, we need to explain the notion of *adjacent fuzzy rules*. Adjacency is related to antecedent parts of fuzzy rules and it will be explained for rules having the two-dimensional relation (2.10) as an antecedent part. The extension of this principle to rules with multidimensional relations is relatively straightforward.

Definition 2.11 (Adjacent fuzzy rules) Let fuzzy rule FR^i have an antecedent relation R_{pq}: P_p^x AND P_q^y and fuzzy sets associated with propositions P_p^x and P_q^y be $F_p(x)$ and $T_q(y)$, respectively. If there exists fuzzy rule FR^j with an antecedent relation R_{vw}: P_v^x AND P_w^y such that a fuzzy set associated with the proposition P_v^x is adjacent to fuzzy set $F_p(x)$ and fuzzy set associated with proposition P_w^y is adjacent to fuzzy set $T_q(y)$, then FR^i is adjacent to FR^j.

If we think of the rule base as a two-dimensional table, we may figure out from Definition 2.10 that a fuzzy rule with a two-dimensional antecedent relation can be surrounded in such a table with eight adjacent rules at most.

Definition 2.12 (Continuity of fuzzy rule base) For a fuzzy rule base to be continuous propositions in consequent parts of any two adjacent fuzzy rules must be (i) associated with adjacent fuzzy sets, or (ii) the same.

The continuity of fuzzy rules will provide the continuity of controller output, which is a desirable feature in all control applications.

Definition 2.13 (Completeness of fuzzy rule base) Fuzzy rule base is *complete* if for each relation, $R_{ijk..}$: $P_i^x \wp P_j^y \wp P_k^z \wp \cdots$, that can be created from input linguistic variables there exists a fuzzy rule with relation $R_{ijk..}$ as an antecedent part.

In practice, the completeness of a fuzzy rule base is rarely achieved [9–11]. For some control problems, only a few rules may be sufficient to provide good control quality, while in other cases certain combinations of linguistic values at controller inputs simply do not or cannot occur. Incompleteness may also reflect a lack of the operator's knowledge about process control.

Besides fuzzy propositions and relations, the consequent parts of fuzzy rules may also have functions and expressions, which explicitly determine the dependence of controller inputs and outputs. This type of controller is often referred to as a *Takagi–Sugeno fuzzy controller* (see References 23 and 12 in Chapters 1 and 2, respectively). The rules of such a controller have the form

$$\text{FR}^i: \quad \text{IF } R_{pq} \text{ THEN } u_i = \rho_i(x_1, x_2, \ldots, x_n) \tag{2.13}$$

where ρ_i is a function and $x_1, x_2, \ldots, x_n$ are numerical (quantitative) values of inputs.

If ρ_i is a linear function, $\rho_i = a_{0i} + a_{1i}x_1 + a_{2i}x_2 + \cdots + a_{ni}x_n$, and coefficients $a_{1i} = a_{2i} = \cdots = a_{ni} = 0$, then the rules of the Takagi–Sugeno controller become identical to the rules of a fuzzy controller containing singletons in the consequent part of the rule

$$\text{FR}^i: \quad \text{IF } R_{pq} \text{ THEN } u_i = a_{0i} = A_{pq} \tag{2.14}$$

where A_{pq} is a fuzzy singleton.

Most fuzzy controllers described in this book have this simple form, usually called zero-order Takagi–Sugeno controller. Due to its simplicity and efficiency, this form is prevalent in industrial applications.

Example 2.3 Fuzzy rule base properties.

Let us characterize the behavior of an air-conditioning system, which bases its actions on outdoor and indoor temperatures, by a fuzzy rule base. We can define the following linguistic variables:

outdoor temperature = {low, moderate, high}
indoor temperature = {cold, warm, hot}
air-conditioner action = {cool, none, heat}

Fuzzy relations, used for antecedent parts of fuzzy rules, relate two input variables, OT = outdoor temperature and IT = indoor temperature. The output variable OA = air-conditioner action will be used in the consequent part of fuzzy rules.

Since there are three linguistic values that each input may take, there are nine possible relations:

R_{11}: OT is *low* and IT is *cold*

R_{12}: OT is *low* and ITis *warm*

R_{13}: OT is *low* and IT is *hot*

R_{21}: OT is *moderate* and IT is *cold*

R_{22}: OT is *moderate* and IT is *warm*

R_{23}: OT is *moderate* and IT is *hot*

R_{31}: OT is *high* and IT is *cold*

R_{32}: OT is *high* and IT is *warm*

R_{33}: OT is *high* and IT is *hot*

Three linguistic values are defined over the universe of discourse of the output variable, which means that three propositions can be identified:

P_1: OA is *cool*

P_2: OA is *none*

P_3: OA is *heat*

Having relations and propositions that form antecedent and consequent parts of fuzzy rules we can now start defining the fuzzy rule base. Based on our experience we may say, for example, that "If *outdoor temperature* is *low* and *indoor temperature* is *cold* then the *air-conditioner* should heat the room," or in the form of fuzzy rule, FR^1: IF R_{11} THEN P_3. By following the same reasoning we can form a fuzzy rule base for air-conditioner functioning, which includes subsequent rules:

FR^2: IF R_{22} THEN P_2

FR^3: IF R_{13} THEN P_2

FR^4: IF R_{21} THEN P_3

FR^5: IF R_{33} THEN P_1

FR^6: IF R_{32} THEN P_1

FR^7: IF R_{12} THEN P_3

If we analyze properties of a rule base defined in such a way, it becomes clear that it is not complete since it does not include rules with relations R_{23} and R_{31}.

Meanwhile, the fuzzy rule base is consistent as there are no rules with the same antecedent parts and different consequent parts.

In order to examine continuity, we have to define fuzzy sets associated with linguistic values of inputs and the output. Let *cool*, *none*, and *heat* be represented with fuzzy sets $\mathrm{OA_c}$, $\mathrm{OA_n}$, and $\mathrm{OA_h}$ having centers at −10, 0, and 10 V, respectively. Fuzzy sets, defined over the universe of discourse of the linguistic variable OT, are $\mathrm{OT_l}$, $\mathrm{OT_m}$, and $\mathrm{OT_h}$, with centers at 5, 20, and 30°C, while values of IT correspond to fuzzy sets $\mathrm{IT_c}$, $\mathrm{IT_w}$, and $\mathrm{IT_h}$ having centers at 15, 20, and 24°C. Once fuzzy sets are defined, we are able to see that, for example, fuzzy rules FR^3 and FR^7 are adjacent. Since fuzzy sets $\mathrm{OT_m}$ and $\mathrm{OT_h}$, associated with their consequent part propositions, are adjacent, these two rules fulfil the continuity definition. By going over all adjacent rules we can verify that the fuzzy rule base, described above, is continuous.

2.3.1 Fuzzy Implication

Whatever form fuzzy rules may have, our main concern is how to interpret the meaning of each rule, that is, how to determine the influence produced by the antecedent part of the fuzzy rule on the consequent part of the rule. The procedure for assessing this influence is called *fuzzy implication.* As the connotations of fuzzy propositions and fuzzy relations are expressed by membership functions, it follows that fuzzy implications, related to rules formed of propositions and relations, also imply membership functions as a method of interpretation.

There are many possible ways to define a fuzzy implication [13], but in control applications two of them are preferred: a *product* (also called Larsen) implication, and a *min* or a *Mamdani* implication:

$$\begin{aligned}\mu_{\mathrm{FR}^i} &= \mu_{R_{pq}} \cdot \mu_{P_m} \\ \mu_{\mathrm{FR}^i} &= \min(\mu_{R_{pq}}, \mu_{P_m})\end{aligned} \tag{2.15}$$

Sometimes, index FR^i in Equation (2.15) is replaced with index $R_{pq} \to P_m$ designating more expressively a selected fuzzy rule, FR^i: IF R_{pq} THEN P_m.

The difference between the two implications will be shown in the example that follows.

Example 2.4 Fuzzy implication.

Let T_1 and F_1 be discrete fuzzy sets with triangular membership functions determined as: $T_1(x) = \{(0.25, 2); (0.5, 3); (0.75, 4); (1, 5); (0.75, 6); (0.5, 7); (0.25, 8)\}$ and $F_1(y) = \{(0.33, 10); (0.67, 11); (1, 12); (0.67, 13); (0.33, 14)\}$. Let a fuzzy rule have the form FR^1: IF R_{11} THEN P_1, and let relation R_{11} have a two-dimensional form, R_{11}: x is T_1 AND y is F_1. A min operator is used as a T-norm.

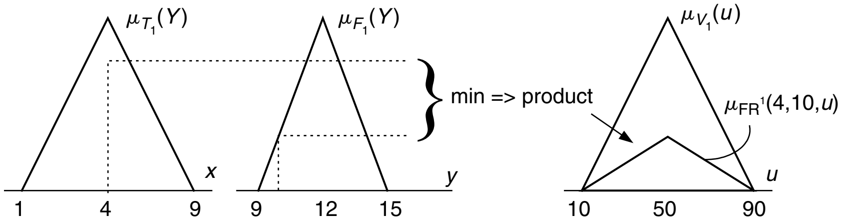


FIGURE 2.6 A fuzzy product implication applied to fuzzy rule FR^1.

For the values $x = 4$, $y = 10$, the relation R_{11} will have the following membership degree:

$$\mu_{R_{11}}(4, 10) = \min[\mu_{T_1}(4), \mu_{F_1}(10)] = \min[0.75, 0.33] = 0.33$$

We have already mentioned that the procedure which assigns degree of membership $\mu_{T_1}(4)$ to the numerical value 4 is called fuzzification. The degrees of membership $\mu_{T_1}(4) = 0.75$ and $\mu_{F_1}(10) = 0.33$ represent fuzzy equivalents of numbers 4 and 10, respectively.

Let proposition P_1 in rule FR^1 have a form, P_1: u is V_1, where $u \in U$ is a linguistic variable, and V_1 is a linguistic value associated with a fuzzy set $V_1(x) = \{(\mu_{V_1}(u), u)\colon u \in U\}$, $V_1(x) = \{(0.25, 20); (0.5, 30); (0.75, 40); (1, 50); (0.75, 60); (0.5, 70); (0.25, 80)\}$. Then the interpretation of fuzzy rule FR^1 will be given by membership function μ_{FR^1} obtained after applying a *product* implication:

$$\mu_{\mathrm{FR}^1}(4, 10, u) = \mu_{R_{11}}(4, 10) \cdot \mu_{V_1}(u) = 0.33 \cdot \mu_{V_1}(u)$$

The fuzzy *product* implication can be graphically presented as shown in Figure 2.6.

In a similar way, after applying a *min* implication, the membership function of fuzzy rule FR^1 assumes the form:

$$\mu_{\mathrm{FR}^1}(4, 10, u) = \min[\mu_{R_{11}}(4, 10), \mu_{V_1}(u)] = \min[0.33, \mu_{V_1}(u)]$$

The fuzzy *min* implication can be graphically presented as shown in Figure 2.7.

The difference between the results of two fuzzy implications is obvious. With the *product* implication, membership function $\mu_{\mathrm{FR}_1}(\cdot)$ is formed by scaling $\mu_{V_1}(u)$ and it retains a triangular form after the implication, while the *min* implication "clips" the original membership function $\mu_{V_1}(u)$ which results in its trapezoidal form. The difference between the results of implications suggests that the type of implication used in fuzzy controller design will have an influence on the structure of the fuzzy control algorithm.

If the THEN part of a fuzzy rule contains a singleton fuzzy set, then the type of the fuzzy implication (*product* or *min*) is insignificant for the result, that is,

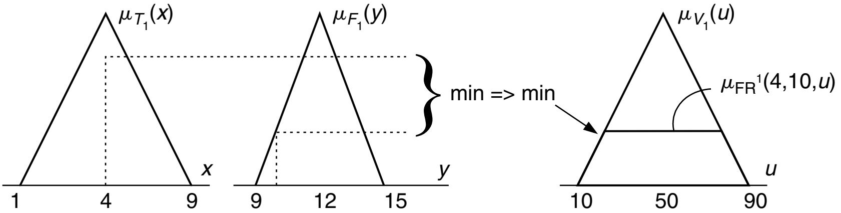


FIGURE 2.7 A fuzzy *min* implication applied to fuzzy rule FR^1.

the resulting fuzzy rule membership function will be the same. In our example,

$$\mu_{\mathrm{FR}^1}(4,10,u) = \min[\mu_{R_{11}}(4,10), \mu_{V_1}(u)] = \min[0.33, 1] = 0.33$$
$$\mu_{\mathrm{FR}^1}(4,10,u) = \mu_{R_{11}}(4,10)\cdot\mu_{V_1}(u) = 0.33\cdot 1 = 0.33$$

As shown in Example 2.9, a fuzzy implication yields a resultant output fuzzy set for each activated fuzzy rule, but it does not define how this fuzzy set really contributes to the crisp output value of a fuzzy controller. Namely, the crisp value of any input variable usually belongs to more than one fuzzy input set which in turn activates more than one fuzzy rule and, therefore, more than one output fuzzy set contributes to the output.

In general, there are two principal ways of computing the contribution of each activated rule: by using either an *individual rule-based* or a *composition-based inference engine*.

The first step of individual rule-based inference, which is predominantly used in fuzzy controller design, has been described in the previous example. For each activated rule we first calculate the membership function of the IF part of the rule (e.g., relation R_{pq}), and then we calculate the influence on the HEN part of the rule (e.g., proposition P_m). When this procedure is carried out for all activated fuzzy rules, a process called *aggregation* concludes individual rule-based inference with one output fuzzy set, which may be used thereafter for the computation of crisp output value [14].

Individual fuzzy rules can be composed in different ways, depending on which aggregation operator we use. There are different aggregation operators, but the *max* operator and the *sum* operator are the most frequently used operators. Their definitions follow.

Definition 2.14 (Max–min aggregation) Let r denote a number of fuzzy rules activated by x_k and y_k and $\mu_{\mathrm{FR}^i}(x_k, y_k, u)$, $i = 1,\ldots,r$, represent a fuzzy interpretation of the ith rule. If a *max* operator is used as an aggregation operator, then the meaning of all fuzzy rules is defined as

$$\mu_U(x_k, y_k, u) = \mu_{\bigcup_{i=1}^{r}\mathrm{FR}^i}(x_k, y_k, u) = \max\{\min_{i=1}^{r}[\mu_{R_{pq}}(x_k, y_k), \mu_{P_m}(u)]\} \tag{2.16}$$

According to Zadeh, such an aggregation is called a *max–min* aggregation. If a *min* implication is replaced with a *product* implication, then such an aggregation is called a *max–product* aggregation. The resultant output fuzzy set will have a different form of membership function in accordance with the differences between the two types of fuzzy implication.

Definition 2.15 (Sum–min aggregation) Let r denote the number of fuzzy rules activated by x_k and y_k, and $\mu_{\mathrm{FR}^i}(x_k, y_k, u)$, $i = 1, \ldots, r$, represent a fuzzy interpretation of the ith rule. If a *sum* operator is used as an aggregation operator, then the meaning of all fuzzy rules is defined as

$$\mu_U(x_k, y_k, u) = \mu_{\bigcup_{i=1}^{r} \mathrm{FR}^i}(x_k, y_k, u) = \sum \min_{i=1}^{r}[\mu_{R_{pq}}(x_k, y_k), \mu_{P_m}(u)] \tag{2.17}$$

This aggregation is called a *sum–min* aggregation. If a *min* implication is replaced with a *product* implication, then such an aggregation is called a *sum–product* aggregation. The membership function of the resultant output fuzzy set obtained with a *sum* operator has a special characteristic: its values may exceed a basic interval [0, 1].

The composition-based inference, rarely applied in control applications, uses membership functions $\mu_{\mathrm{FR}^i}(x, y, u)$, $i = 1, \ldots, n$, for the calculation of one compositional output membership function $\mu_{\mathrm{FR}}(x, y, u)$:

$$\mu_{\mathrm{FR}}(x, y, u) = \mu_{\bigcup_{i=1}^{n} \mathrm{FR}^i}(x, y, u) = \max\{\min_{i=1}^{n}[\mu_{R_{pq}}(x, y), \mu_{P_m}(u)]\} \tag{2.18}$$

The difference between membership functions (2.16) and (2.18) should be noted. While the first one is calculated only for rules that have been activated by a pair of crisp input values x_k, y_k and it is defined over domain U, the former represents a function defined over a three-dimensional domain $X \times Y \times U$.

Once a compositional membership function $\mu_{\mathrm{FR}}(x, y, u)$ is calculated, the second step in composition-based inference needs to be taken: the determination of membership functions of antecedent parts of rules activated by a pair of crisp input values x_k and y_k. The output fuzzy equivalent of these inputs is then calculated as

$$\mu_U(x_k, y_k, u) = \max\{\min_{i=1}^{r}[\mu_{R_{pq}}(x_k, y_k), \mu_{\mathrm{FR}}(x, y, u)]\}, \tag{2.19}$$

where r is the number of rules activated by inputs x_k and y_k, and $\mu_{Rpq}(x_k, y_k)$ are fuzzy equivalents of the antecedent parts of activated rules.

The difficulty with composition-based inference lies in the fact that we have to calculate and store a “massive” compositional membership function $\mu_{\mathrm{FR}}(x, y, u)$ in order to be able to interpret activated rules. The second drawback is the determination and computation of membership functions $\mu_{Rpq}(x, y)$ (in case of using a *min* T-norm and triangular fuzzy sets they are represented by prismatic surfaces — see Figure 2.5).

One can argue that computer memory required for storing $\mu_{\mathrm{FR}}(x, y, u)$ and computational power necessary for calculations of $\mu_{Rpq}(x, y)$ and $\mu_U(x', y', u)$ do not present a problem for today's powerful computers. That is true if we fail to consider the range of cheap and compact microcontroller-based solutions widely present in industrial applications.

Regarding the decision of which inference engine to use in fuzzy controller design we should not forget the way we as operators and designers process information. It is much easier to understand the combined influence of several rules by considering each of them first, than try to comprehend a combined action of all possible rules and then make conclusions about individual influences of several activated rules.

Example 2.5 Fuzzy inference.

Suppose that two fuzzy rules

$$\mathrm{FR}^1: \quad \text{IF } x \text{ is } T_0 \text{ AND } y \text{ is } F_1 \text{ THEN } u \text{ is } V_1$$

$$\mathrm{FR}^2: \quad \text{IF } x \text{ is } T_1 \text{ AND } y \text{ is } F_1 \text{ THEN } u \text{ is } V_2$$

are activated by crisp input values $x = 4$ and $y = 10$, where discrete fuzzy sets $T_0(x)$, $T_1(x)$, and $F_1(y)$ are defined as follows: $T_0(x) = \{(0.25, -2); (0.5, -1); (0.75, 0); (1, 1); (0.75, 2); (0.5, 3); (0.25, 4)\}$, $T_1(x) = \{(0.25, 2); (0.5, 3); (0.75, 4); (1, 5); (0.75, 6); (0.5, 7); (0.25, 8)\}$, $F_1(y) = \{(0.33, 10); (0.67, 11); (1, 12); (0.67, 13); (0.33, 14)\}$, while $V_1(u) = \{(0.25, 20); (0.5, 30); (0.75, 40); (1, 50); (0.75, 60); (0.5, 70); (0.25, 80)\}$ and $V_2(u) = \{(0.25, 60); (0.5, 70); (0.75, 80); (1, 90); (0.75, 100); (0.5, 110); (0.25, 120)\}$ are discrete fuzzy sets defined over the universe of discourse of output variable u.

Let also an output fuzzy set be calculated by using individual rule-base inference. The procedure can then be split into two consecutive steps. First, we determine the activation degrees of the antecedent parts of rules FR^1 and FR^2:

$$\mu_{R_{01}}(4, 10) = \min[\mu_{T_0}(4), \mu_{F_1}(10)] = \min[0.25, 0.33] = 0.25$$

$$\mu_{R_{11}}(4, 10) = \min[\mu_{T_1}(4), \mu_{F_1}(10)] = \min[0.75, 0.33] = 0.33$$

Then, after applying the Mamdani (*min*) implication, activation degrees of the consequent parts of rules FR^1 and FR^2 become:

$$\mu_{\mathrm{FR}^1}(4, 10, u) = \min[\mu_{R_{01}}(4, 10), \mu_{V_1}(u)] = \min[0.25, \mu_{V_1}(u)]$$

$$\mu_{\mathrm{FR}^2}(4, 10, u) = \min[\mu_{R_{11}}(4, 10), \mu_{V_2}(u)] = \min[0.33, \mu_{V_2}(u)]$$

Since fuzzy sets $V_1(u)$ and $V_2(u)$ have seven elements, fuzzy sets that represent rules FR^1 and FR^2 obtain the following form:

$$\begin{aligned}\mathrm{FR}^1(4, 10, u) =&\{(0.25, 20); (0.25, 30); (0.25, 40); (0.25, 50); (0.25, 60);\\ &(0.25, 70); (0.25, 80)\}\\ \mathrm{FR}^2(4, 10, u) =&\{(0.25, 60); (0.33, 70); (0.33, 80); (0.33, 90); (0.33, 100);\\ &(0.33, 110); (0.25, 120)\}\end{aligned}$$

Once individual contributions of both rules are determined, we may proceed with the second step — the aggregation of two fuzzy sets into one output fuzzy set. By using Equation (2.16) we obtain:

$$\mu_{U_{12}}(4, 10, u) = \mu_{\bigcup_{i=1}^{2} \mathrm{FR}^i}(4, 10, u) = \max\{\mu_{\mathrm{FR}^1}(4, 10, u), \mu_{\mathrm{FR}^2}(4, 10, u)\}$$

which yields the overall output fuzzy set $U_{12}(u)$:

$$\begin{aligned}U_{12}(u) = \{&(0.25, 20); (0.25, 30); (0.25, 40); (0.25, 50); (0.25, 60); (0.33, 70);\\ &(0.33, 80); (0.33, 90); (0.33, 100); (0.33, 110); (0.25, 120)\}\end{aligned}$$

By using Equation (2.17), where a *sum* aggregation is used, the fuzzy output set attains the form:

$$\begin{aligned}U_{12}(u) = \{&(0.25, 20); (0.25, 30); (0.25, 40); (0.25, 50); (0.5, 60); (0.58, 70);\\ &(0.58, 80); (0.33, 90); (0.33, 100); (0.33, 110); (0.25, 120)\}\end{aligned}$$

A graphical presentation of the above procedure is shown in Figure 2.8.

Even though only individual rule-base inference is used throughout the rest of the book, for the sake of comparison we will now show the first step in the mechanism of compositional rule-base inference. Let us suppose that a rule base contains only two rules already defined at the beginning of the example. According to the compositional inference engine defined by Equation (2.18), we need to calculate the membership functions of all triples (x, y, u) determined by the rules. As fuzzy sets $T_0(x)$, $T_1(x)$, $V_1(u)$, and $V_2(u)$ have seven elements, and set $F_1(y)$ has five elements, we have to evaluate $[(7 \times 5) \times 7] \times 2 = 490$ values for two rules. Table 2.2 shows only the first few inputs of the membership function for the first rule in the rule base.

After calculating 245 values for the first rule, we must calculate another 245 values for the second rule. Then, according to Equation (2.18), a *max* operation finally gives a compositional output membership function $\mu_{\mathrm{FR}}(x, y, u)$ with all 450 entries.

This first step in a compositional rule-base inference alone is enough to show how many calculations and memory this method requires. Obviously, the aggregation of individual rules is much simpler and easier to implement. Moreover,

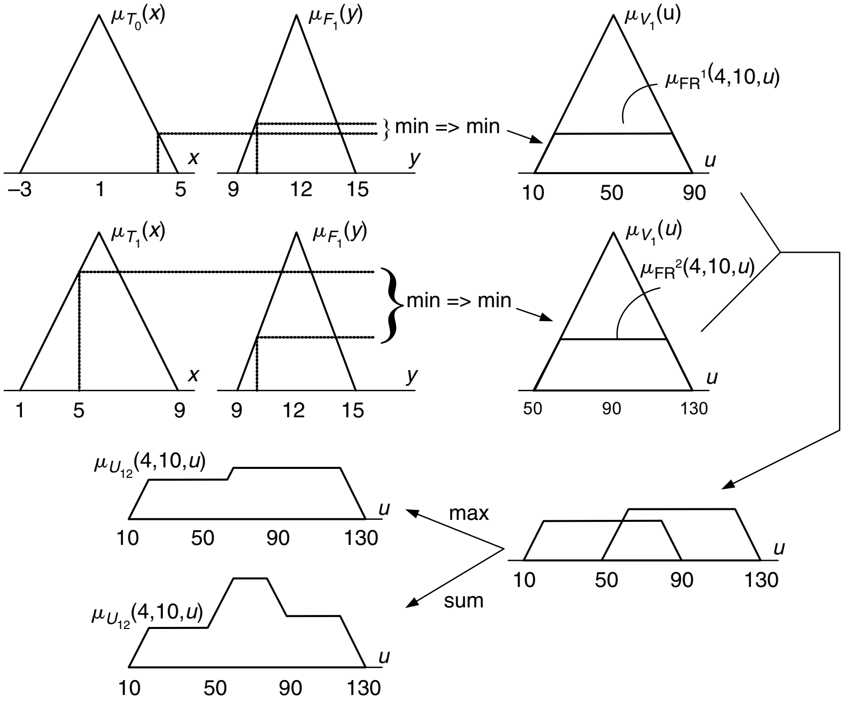


FIGURE 2.8 A graphical presentation of an individual rule-based inference procedure.

in case a *min* implication is used, the final result, an output fuzzy set, is the same for both inference methods. That is yet another reason that justifies our preference of using individual rule-based inference in fuzzy controller design.

2.3.2 Defuzzification

The result of fuzzy inference is a fuzzy output set. On the other hand, every control task will imply the existence of crisp value at the fuzzy controller output. The procedure which extracts crisp output value from a fuzzy output set is called *defuzzification*.

There are various types of defuzzification [15]. However, crisp output value is most frequently calculated according to the *center of area* (COA) principle:

$$u_{\mathrm{FC}}(x_k, y_k) = \frac{\sum_i u_i \cdot \mu_u(x_k, y_k, u_i)}{\sum_i \mu_u(x_k, y_k, u_i)} \tag{2.20}$$

where $u_{\mathrm{FC}}(x_k, y_k)$ represents the crisp value of the fuzzy controller output, $u_i \in U$ is a discrete element of an output fuzzy set, and $\mu_u(x_k, y_k, u_i)$ is its membership function.

Table 2.2
A Fraction of Membership Function Values for Compositional Rule-Base Inference

x	y	u	$\mu(x)$	$\mu(y)$	$\mu(u)$	min[μ]
−2	10	20	0.25	0.33	0.25	0.25
−2	10	30	0.25	0.33	0.5	0.25
−2	10	40	0.25	0.33	0.75	0.25
−2	10	50	0.25	0.33	1	0.25
−2	10	60	0.25	0.33	0.75	0.25
−2	10	70	0.25	0.33	0.5	0.25
−2	10	80	0.25	0.33	0.25	0.25
−2	11	20	0.25	0.67	0.25	0.25
−2	11	30	0.25	0.67	0.5	0.25
−2	11	40	0.25	0.67	0.75	0.25
−2	11	50	0.25	0.67	1	0.25
−2	11	60	0.25	0.67	0.75	0.25
−2	11	70	0.25	0.67	0.5	0.25
−2	11	80	0.25	0.67	0.25	0.25
−2	12	20	0.25	1	0.25	0.25
−2	12	30	0.25	1	0.5	0.25
−2	12	40	0.25	1	0.75	0.25
−2	12	50	0.25	1	1	0.25
−2	12	60	0.25	1	0.75	0.25
−2	12	70	0.25	1	0.5	0.25
−2	12	80	0.25	1	0.25	0.25
−2	13	20	0.25	0.67	0.25	0.25
−2	13	30	0.25	0.67	0.5	0.25
−2	13	40	0.25	0.67	0.75	0.25
−2	13	50	0.25	0.67	1	0.25
−2	13	60	0.25	0.67	0.75	0.25
−2	13	70	0.25	0.67	0.5	0.25
−2	13	80	0.25	0.67	0.25	0.25

Equation (2.20) is a discrete form of the COA method. In case of a continuous universe of discourse, sums in the equation should be replaced by integrals.

The other defuzzification method, which is very often used in control applications, is a COG method. For discrete universe of discourse, fuzzy controller output u_{FC} is calculated according to the COG principle in the following way:

$$u_{\text{FC}}(x_k, y_k) = \frac{\sum_i u_i \sum_{j=1}^{r} \mu_{\text{FR}^j}(x_k, y_k, u_i)}{\sum_i \sum_{j=1}^{r} \mu_{\text{FR}^j}(x_k, y_k, u_i)} \tag{2.21}$$

where r is the number of fuzzy rules activated by crisp inputs x_k and y_k.

From Equation (2.21) it may be seen that the COG method does not require aggregation since it already works with individual output fuzzy sets obtained after the processing of fuzzy rules. The COG method's distinguished features are marked simplicity and very low computing effort, which enables short control intervals necessary for the control of highly dynamic systems. This is the main reason why this method of defuzzification is used for fuzzy controller design throughout this book. The second reason is that when *max* aggregation is used and several activated rules have the same consequent part (or demand the same type of controller action), only the one with the highest membership function will contribute to crisp output value, negating others, if COA is used. Ignoring the rules with lower membership functions creates a situation when greater weight could be given to rules that are perhaps less important. The COG method takes into account such a situation and calculates contributions of all activated rules regardless of the fact that consequent parts may be the same.

Some designers often refer to the COG method as the *center of sums* defuzzification method. That is because the sums of membership functions appear in Equation (2.21). Also, the COG method (2.21) is very often equalled to the COA method (2.19), which stems from the fact that the COA of a membership function $\mu_u(x_k, y_k, u)$ created by the aggregation operator *sum* (Figure 2.8) is equal to the COG obtained from Equation (2.21) for individual fuzzy output sets. In the text that follows we refer to COA and COG as they are defined in Equations (2.20) and (2.21), respectively.

While we were describing the most frequently used shapes of membership functions, we stressed the importance of singletons — single-valued fuzzy sets — in practical control applications. Now we can show that the simplest way of calculating crisp output value is obtained by the substitution of usual (triangular, trapezoidal, etc.) fuzzy sets with singletons. Singletons have only one element u_c corresponding to the projection of a centroid of a likely fuzzy set on the controller output universe of discourse. Consequently, there is only one membership function inside the inner sum of Equation (2.21), while the number of elements in the outer sums of Equation (2.21) is equal to the number of activated fuzzy rules ($i = r$), which yields:

$$u_{\text{FC}}(x_k, y_k) = \frac{\sum_{i=1}^{r} u_{c_i} \cdot \mu_{\text{FR}^i}(x_k, y_k, u_{c_i})}{\sum_{i=1}^{r} \mu_{\text{FR}^i}(x_k, y_k, u_{c_i})} \tag{2.22}$$

By substituting $A_i = u_{c_i}$ and $\mu_i = \mu_{\text{FR}^i}(x_k, y_k, u_{c_i})$, Equation (2.22) gets a simpler form:

$$u_{\text{FC}}(x_k, y_k) = \frac{\sum_{i=1}^{r} A_i \mu_i}{\sum_{i=1}^{r} \mu_i} = \left| \varphi_i = \frac{\mu_i}{\sum_{i=1}^{r} \mu_i} \right| = \sum_{i=1}^{r} A_i \varphi_i \tag{2.23}$$

where φ_i is a *fuzzy basis function* describing how much each activated fuzzy rule contributes to the crisp value of the fuzzy controller output.

There is a special case when triangular input fuzzy sets are used and when only two adjacent fuzzy sets overlap with the intersection point value $\mu(x) = 0.5$. In this case, after the application of the *product* implication, expression (2.23) converts into a very simple form:

$$\sum_{j=1}^{r} \mu_j = 1, \quad \varphi_j = \mu_j, \quad u_{\mathrm{FC}} = \sum_{j=1}^{r} A_j \mu_j \tag{2.24}$$

Defuzzification which uses singletons corresponds to the method known in literature as *height defuzzification*. Having described the most frequently used defuzzification methods we have covered all the essential terms and elements necessary for the design and implementation of a fuzzy controller. Before we focus on the structure of the fuzzy controller itself, and start discussion about basic directions in controller design in order to achieve desired control quality, we give an example of the above mentioned defuzzification methods.

Example 2.6 Defuzzification.

Let fuzzy sets $\mathrm{FR}^1(4, 10, u) = \{(0.25, 20); (0.25, 30); (0.25, 40); (0.25, 50); (0.25, 60); (0.25, 70); (0.25, 80)\}$, and $\mathrm{FR}^2(4, 10, u) = \{(0.25, 60); (0.33, 70); (0.33, 80); (0.33, 90); (0.33, 100); (0.33, 110); (0.25, 120)\}$, from Example 2.5, interpret two activated rules. Having those sets, our task is to determine crisp output value.

Let us first use COA defuzzification. It requires the aggregation of FR1 and FR2 which has been already done in Example 2.5. For *max* aggregation we obtained

$$\begin{aligned} U_{12}(u) = \{&(0.25, 20); (0.25, 30); (0.25, 40); (0.25, 50); (0.25, 60); (0.33, 70); \\ &(0.33, 80); (0.33, 90); (0.33, 100); (0.33, 110); (0.25, 120)\} \end{aligned}$$

The inclusion of these values in Equation (2.19) gives

$$\begin{aligned} u_{\mathrm{FC}}(4, 10) &= [0.25 \cdot 20 + 0.25 \cdot 30 + 0.25 \cdot 40 + 0.25 \cdot 50 + 0.25 \cdot 60 \\ &\quad + 0.33 \cdot 70 + \cdots + 0.25 \cdot 120][0.25 + 0.25 + 0.25 + 0.25 + 0.25 \\ &\quad + 0.33 + \cdots + 0.25]^{-1} \\ &= \frac{228.5}{3.15} = 72.5397 \end{aligned}$$

In case *sum* aggregation is used, the fuzzy output set attains this form:

$$\begin{aligned} U_{12}(u) = \{&(0.25, 20); (0.25, 30); (0.25, 40); (0.25, 50); (0.5, 60); (0.58, 70); \\ &(0.58, 80); (0.33, 90); (0.33, 100); (0.33, 110); (0.25, 120)\} \end{aligned}$$

which yields

$$
\begin{aligned}
u_{\mathrm{FC}}(4,10) &= [0.25\cdot 20 + 0.25\cdot 30 + 0.25\cdot 40 + 0.25\cdot 50 + 0.5\cdot 60 + 0.58\cdot 70\\
&\quad + \cdots + 0.25\cdot 120][0.25 + 0.25 + 0.25 + 0.25 + 0.5 + 0.58\\
&\quad + \cdots + 0.25]^{-1}\\
&= \frac{281}{3.9} = 72.0513
\end{aligned}
$$

Now we can calculate output according to the COG principle. By including FR^1 and FR^2 in Equation (2.21), the following crisp value is obtained:

$$
\begin{aligned}
u_{\mathrm{FC}}(4,10) &= [0.25\cdot 20 + 0.25\cdot 30 + 0.25\cdot 40 + 0.25\cdot 50\\
&\quad + (0.25+0.25)\cdot 60 + (0.25+0.33)\cdot 70 + \cdots + 0.25\cdot 120]\\
&\quad \times [0.25 + 0.25 + 0.25 + 0.25 + (0.25+0.25) + (0.25+0.33)\\
&\quad + \cdots + 0.25]^{-1}\\
&= \frac{281}{3.9} = 72.0513
\end{aligned}
$$

As we have mentioned earlier, this result confirms that COA and COG methods yield the same crisp value in case *sum* aggregation is used.

The difference between calculations of COG and COA when *max* aggregation is used is graphically presented in Figure 2.9. We can see that the shaded surface at the cross section of two fuzzy sets will appear twice in the COG method and only once in the COA method. From Equation (2.20) we see that the COA is calculated by multiplying each discrete value $u_i \in U$ with a membership function $\mu_u(x_k, y_k, u_i)$ obtained either by individual rule-based or compositional inference engine, while in Equation (2.21) the COG is calculated by multiplying each discrete value $u_i \in U$ with the sum of membership functions of the activated fuzzy rules. This will cause a difference between two crisp values of controller output, as shown in the example and depicted in Figure 2.9.

Let us now assume that instead of fuzzy output sets with triangular membership functions, fuzzy output singletons are used; specifically, $V_1(u) = \{(1, 50)\}$ and $V_2(u) = \{(1, 90)\}$. By insertion of these singletons into the equations from

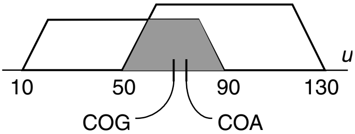


FIGURE 2.9 Agraphical presentation of results obtained by COG and COA defuzzification methods.

Example 2.5 we have

$$\mu_{\mathrm{FR}^1}(4, 10, u) = \min[\mu_{R_{01}}(4, 10), \mu_{V_1}(u)] = \min[0.25, 1] = 0.25$$

$$\mu_{\mathrm{FR}^2}(4, 10, u) = \min[\mu_{R_{11}}(4, 10), \mu_{V_2}(u)] = \min[0.33, 1] = 0.33$$

Relation (2.22) gives crisp output value

$$u_{\mathrm{FC}}(4, 10) = \frac{0.25 \cdot 50 + 0.33 \cdot 90}{0.25 + 0.33} = \frac{42.2}{0.58} = 72.7586$$

It is apparent that the use of singletons enormously simplifies the defuzzification process. Crisp output value can be calculated with only a few additions and multiplications.

We need to point out one more detail. The very small difference between crisp output numerical results acquired by different methods in this example may lead to the wrong conclusion that there is not a large difference between COA and COG. We leave it to the reader to perform an exercise in order to find out that the activation of two additional rules with the same consequent parts

$$\mathrm{FR}^3\text{:}\quad \text{IF } x \text{ is } T_0 \text{ AND } y \text{ is } F_0 \text{ THEN } u \text{ is } V_1$$

$$\mathrm{FR}^4\text{:}\quad \text{IF } x \text{ is } T_1 \text{ AND } y \text{ is } F_0 \text{ THEN } u \text{ is } V_1$$

where $F_0(y) = \{(0.33, 7); (0.67, 8); (1, 9); (0.67, 10); (0.33, 11)\}$, shall give COA- and COG-based crisp output values which differ more than the values in our example with only two rules.

2.4 Fuzzy Controller Structure

The kind of a structure a fuzzy controller will have will primarily depend on the controlled process and the demanded quality of control. Since the application area for fuzzy control is really wide, there are many possible controller structures, some differing significantly from each other by the number of inputs and outputs, or less significantly by the number of input and output fuzzy sets and their membership functions forms, or by the form of control rules, the type of inference engine, and the method of defuzzification. All that variety is at the designer's disposal, and it is up to the designer to decide which controller structure would be optimal for a particular control problem [16–19].

For example, if the controlled process exhibits integral behavior (we say it is astatic), then a so-called *non-integral* or *PD-type* fuzzy controller whose crisp output value represents absolute control input value could provide the required quality of control. On the other hand, a so-called *integral* or *PI-type* fuzzy controller whose crisp output value represents an increment of control input value could be a satisfactory solution for the control of static systems [20,21].

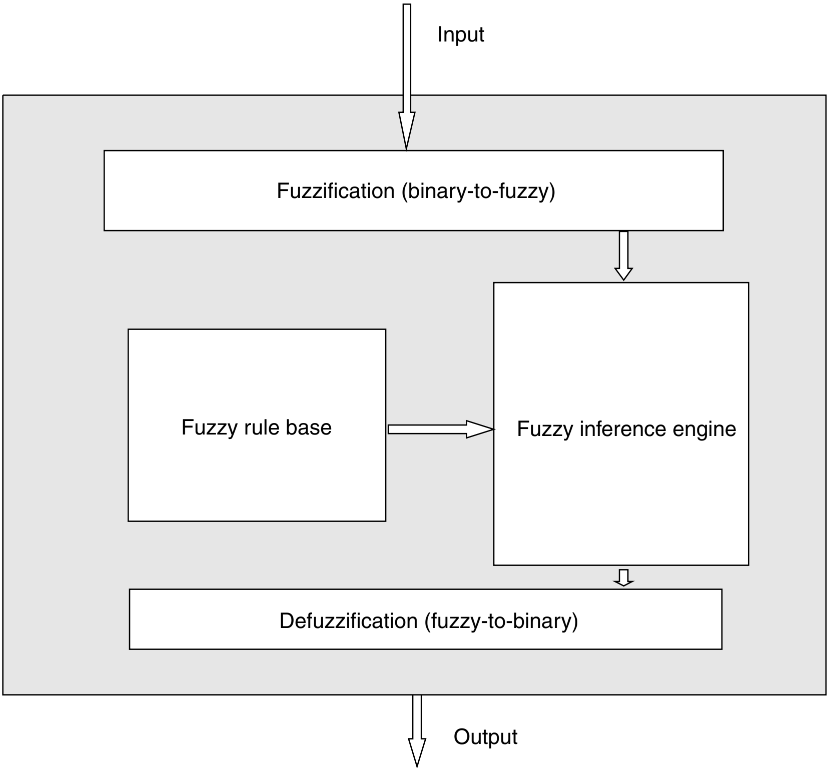

Input
Fuzzification (binary-to-fuzzy)
Fuzzy rule base
Fuzzy inference engine
Defuzzification (fuzzy-to-binary)
Output

FIGURE 2.10 The structure of a fuzzy logic controller.

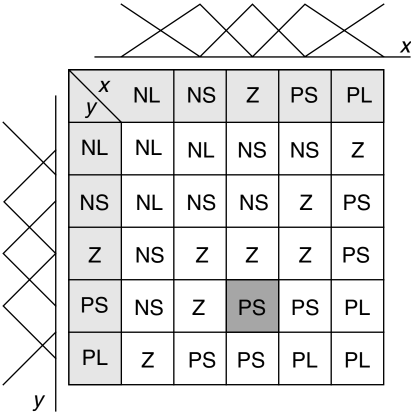
The usage of fuzzy algorithms is not limited to fuzzy logic controllers. Fuzzy algorithms can be used equally well as nonlinear adaptation mechanisms, universal approximators or as auxiliary units added to some conventional control solutions [12,22–25]. We should not fail to mention that fuzzy controllers are very convenient as supervisory controllers [26]. Sometimes, fuzzy logic algorithms are also used as modal or fuzzy state controllers [27].

Despite the variety of possible fuzzy controller structures, the basic form of all common types of controllers consists of:

- Input fuzzification (binary-to-fuzzy [B/F] conversion)
- Fuzzy rule base
- Inference engine
- Output defuzzification (fuzzy-to-binary [F/B] conversion)

The basic structure of a fuzzy controller is shown in Figure 2.10.

Although there are many analog fuzzy controllers on the market, most of today's fuzzy controllers are implemented in digital form (the fuzzy controllers



y \ x	NL	NS	Z	PS	PL
NL	NL	NL	NS	NS	Z
NS	NL	NS	NS	Z	PS
Z	NS	Z	Z	Z	PS
PS	NS	Z	PS	PS	PL
PL	Z	PS	PS	PL	PL

FIGURE 2.11 A fuzzy rule base displayed as a fuzzy rule table.

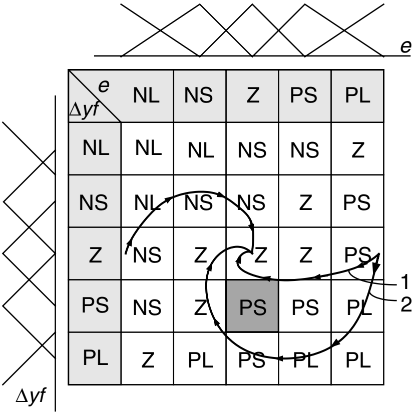
described in this book belong to that group as well). This is the reason why the term B/F conversion is introduced herein, as inputs of a digital fuzzy controller are defined over discrete universes of discourse with the finite number of elements (integers) obtained after quantization of sensor signals (A/D or f/D conversion). Also, the output of such a controller has discrete universe of discourse, while F/B conversion represents deffuzification, which results in digital output value.

The fuzzy rule base, which reflects the collected knowledge about how a particular control problem must be treated, is the heart of a fuzzy controller. The other parts of the controller perform service tasks necessary for the controller to be fully functional.

2.4.1 Fuzzy Rule Table

The most frequently used structure of a fuzzy controller is the double input–single output (DISO) structure. In case of designing such a controller, a very convenient form of displaying the complete fuzzy rule base is a *fuzzy rule table* (Figure 2.11). Every rule in the fuzzy rule table is represented by an output fuzzy set engaged in the THEN part of the rule. The rule position within the fuzzy rule table is determined by coordinates of inputs fuzzy sets engaged in the IF part of the rule. Thus the fuzzy rule table provides straight insight into the essence of the fuzzy rule base and automatically eliminates the creation of contradictory fuzzy rules. The tabular format also makes an elegant entry of new fuzzy rules possible.

Figure 2.11 shows the fuzzy rule-table of a DISO fuzzy controller with $l = 5$ triangular fuzzy sets defined for both inputs x and y, and output u as following: negative large (NL); negative small (NS); around zero (Z); positive small (PS); and positive large (PL). For the studied $l \times l = 5 \times 5$ table a number of fuzzy rules may increase up to $l^2 = 25$ rules.



e / Δyf	NL	NS	Z	PS	PL
NL	NL	NL	NS	NS	Z
NS	NL	NS	NS	Z	PS
Z	NS	Z	Z	Z	PS
PS	NS	Z	PS	PS	PL
PL	Z	PL	PS	PL	PL

FIGURE 2.12 Phase trajectories drawn in a fuzzy rule table.

The shaded rule in Figure 2.11 can be read as follows:

IF x is Z AND y is PS THEN u is PS

A short glance at the table confirms the completeness (all 25 rules are there) and the continuity of the displayed fuzzy rule-base (consistency is automatically provided). A fuzzy rule table can also be viewed as the state space of two process variables x and y (e.g., let $x = e(k)$ — control error, $y = \Delta y_f(k)$ — change of a process output, where k is the substitute for kT_d, and T_d is a sampling interval). By using a fuzzy rule table, we get the chance to see the corresponding phase trajectories resulting from consecutive switching of fuzzy rules (Figure 2.12).

We have already mentioned that the design of a fuzzy controller is actually a heuristic search for the best fitted static nonlinear mapping function between controller inputs and outputs. As a result of mapping, every discrete trajectory $[e(k), \Delta y_f(k)]$ has a matching controller output series $u_{\mathrm{FC}}(k), k = 0, 1, \ldots, \infty$ (Figure 2.13). This allows us to interpret the fuzzy rule table as a partitioned state space composed of a phase plane and a corresponding fuzzy control surface lying above the plane. Every controller output sequence $u_{\mathrm{FC}}(k)$ belongs to this fuzzy control surface. Any changes made in the fuzzy rule table during the design process will change the path of phase trajectories. Therefore, these trajectories are very useful for getting a better insight into the progress of an ongoing controller design. By following the trajectory during a transient response one can easily find which fuzzy control rules are activated and how they contribute to crisp output value.

A fuzzy rule table viewed as a phase plane is frequently used for heuristic assessment of closed-loop system stability, as it offers an elegant way to investigate the influence of individual control rules (their THEN parts) on the shape of phase trajectories [28,29].

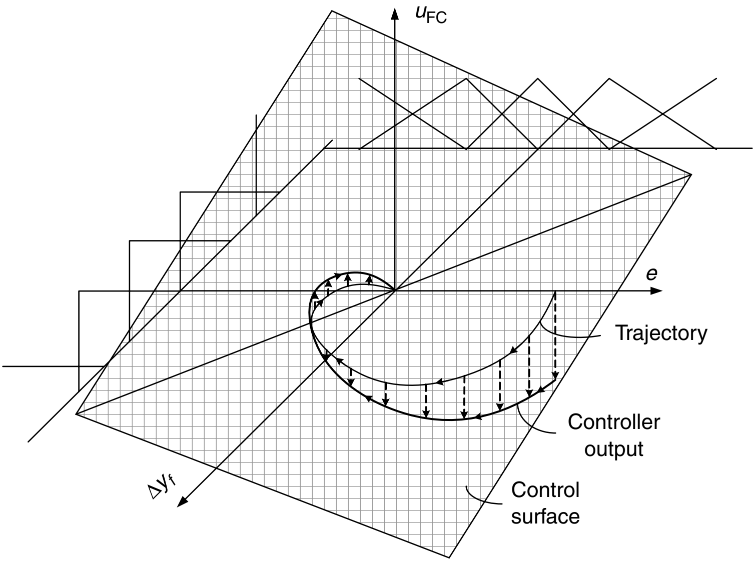


FIGURE 2.13 Phase trajectory with matching fuzzy controller output.

In order to bring more generality into the process of controller design, we would advise normalization of controller input and output domains. The universe of discourse of fuzzy controller inputs and outputs varies from one application to another. To avoid having to make adjustments for each application, inputs and outputs can be scaled to fit the normalized universes of discourse. When we use the term normalized fuzzy controller, we have in mind a controller whose fuzzification, fuzzy rule base and defuzzification parts operate with normalized values usually lying in the interval $[-1, 1]$.

The normalization of inputs should be performed before proceeding with fuzzification:

$$x_{\mathrm{N}}(k) = K_x x(k) \tag{2.25}$$

where x is controller input, x_{N} is normalized controller input, and K_x is the scaling factor.

Thus, for example, control error $e(k)$ and change in process output $\Delta y_f(k)$ after normalization become

$$e_{\mathrm{N}}(k) = K_e e(k)$$

$$\Delta y_{\mathrm{N}}(k) = K_{\Delta y} \Delta y_f(k)$$

where K_e and $K_{\Delta y}$ are scaling factors.

TABLE 2.3
Some Commonly Used Fuzzy Controller Inputs

Name of linguistic variable	Continuous fuzzy controller	Discrete fuzzy controller
System error, control error	$e(t)$	$e(k)$
Control error derivative: control error differential	$\mathrm{d}e(t)/\mathrm{d}t$	$\Delta e(k),\ \mathrm{d}e(k)$
Control error integral; control error sum	$\int e(t)\,\mathrm{d}t$	$\sum e(k)$
System output derivative; system output differential	$\mathrm{d}y_f(t)/\mathrm{d}t$	$\Delta y_f(k),\ \mathrm{d}y_f(k)$
State variable vector	$x(t)$	$x(k)$

The normalized variable $x_{\mathrm{N}}(k)$ is thereafter converted into its fuzzy equivalent. The impact of input scaling factors on phase trajectories can be quite significant. We show in Figure 2.12 two trajectories, the more shortened trajectory 1, obtained without scaling, and the more stretched trajectory 2, obtained by the scaling of inputs. Normalized trajectory 2 activates nine fuzzy rules, while trajectory 1 only three. Apparently, different input values trigger different fuzzy control rules, which eventually result in completely different controller output values. In general, it is much easier to achieve the desired control quality with a larger number of activated fuzzy rules. That is why the scaling of inputs should be done carefully so that we can use the full fuzzy rule base. Because of inadequate scaling, the fuzzy rule table may be imperfectly partitioned, which may cause many of the rules to remain inactive even though the rule base is complete.

It is worth mentioning that scaling factors, including controller output scaling factor K_u, have such a strong impact on the dynamic behavior of a control system because they directly influence the value of the open-loop gain coefficient.

2.4.2 Choice of Shape, Number, and Distribution of Fuzzy Sets

Although its rule base is the core of a fuzzy controller, an important issue in controller design still remains the choice of linguistic values and their membership functions: their shape, number and distribution [30–33]. Before we analyze the influence of these parameters on fuzzy controller behavior, let us introduce some commonly used fuzzy controller inputs, displayed in Table 2.3. Variations in the selection of fuzzy controller inputs arise from the character of the controlled process and the controller itself, whether it is continuous or discrete.

Control error $e(k)$ is used as an input in almost all fuzzy controllers intended to replace standard controllers in single input–single output (SISO) control systems. As the second input of such controllers designers usually choose a differential or change of control error $de(k) = \Delta e(k) = e(k) - e(k-1)$. On the other hand, the sum of the control error is rarely used since a summed/up horizon tends to become infinite, which creates problems with calculation making it impractical. Instead of a summation, an integral term, if needed for a particular application, is then converted into a more suitable form for calculation. In control systems where reference input may change extensively, change of a system output, $\Delta y_f(k)$, may be used instead of $\Delta e(k)$. The sudden change of the reference input causes a considerable change of $e(k)$ and $\Delta e(k)$, which in turn yields a significant change of the controller output. In case $\Delta y_f(k)$ is used, the controller response to an abrupt change of the reference input would be stress-free. This option is often used in implementations of standard PID controllers where an input to a derivative term becomes the system output instead of the control error.

If a fuzzy controller is used as an adaptation mechanism then control error $e(k)$ is replaced with the reference model tracking error $e_{\mathrm{M}}(k)$, which denotes the difference between reference model and system outputs, $e_{\mathrm{M}}(k) = y_{\mathrm{M}}(k) - y_f(k)$. This type of a fuzzy controller will be discussed in the chapters that follow.

Regarding fuzzy controller outputs, either an absolute controller output $u_{\mathrm{FC}}(t), u_{\mathrm{FC}}(k)$, or controller output increment $du_{\mathrm{FC}}(t)$, $\Delta u_{\mathrm{FC}}(k)$ are usually generated. Which type of output will be generated depends on the type of the fuzzy controller. The output is then fed to the controlled process directly or it is used, for example, for adaptation.

The number of universes of discourse is equal to the number of fuzzy controller inputs and outputs. Each input has a given distribution of fuzzy sets on its universe of discourse. A general rule is that for a given distribution of fuzzy sets, the number of fuzzy control rules increases geometrically with the number of inputs. The geometrical progression of the number of rules becomes an obstacle for practical applications of multi-input fuzzy controllers [34]. Therefore, fuzzy controllers with two or at most three inputs prevail over others. Although intensive effort has been put into the development of fuzzy control structures that would solve the problem of rules explosion, effective solutions that work in engineering practice are still not available [35].

Another reason why fuzzy controllers with only two or three inputs are used lies in the fact that human perception is limited. In an everyday decision process we usually do not take into account more than two or three propositions, very rarely four, and almost never five or more at the same time. Since the main task of a fuzzy controller is the interpretation of heuristic knowledge provided by the operator, two or three inputs are usually enough to summarize the operator's comprehension. One good example is driving a car, where due to a lot of information to be processed the extended driver's training is required. The number of input variables affecting driver's actions may vary depending on the driving conditions. While simple cruise control is mostly based on the driver's assessment of the distance from the leading vehicle, during parking or passing actions the driver must also consider additional input variables. An on-line adjustment of the fuzzy controller inputs, which would be responsible for successful decision making in a particular situation, is the property of a *variable structure fuzzy controller.*

Whether we follow an operator's description of control actions or design a controller from our experience, our perception of a control problem will govern the controller's number of inputs and outputs.

In further examples we shall use inputs designated with the following linguistic variables: "control error," noted e, "control error change," noted $\Delta e(\mathrm{d}e)$, "system output change," noted Δy_f, "tracking error," noted e_{M}, and "tracking error change," noted $\Delta e_{\mathrm{M}}(\mathrm{d}e_{\mathrm{M}})$. Accordingly, the accompanying universes of discourse E, DE, DYF, EM, and DEM, will be defined if they satisfy conditions $e \in \mathrm{E}$, $\Delta e \in \mathrm{DE}$, $\Delta y_f \in \mathrm{DYF}$, $e_{\mathrm{M}} \in \mathrm{EM}$, and $\Delta e_{\mathrm{M}} \in \mathrm{DEM}$.

Usually, but not necessarily, the number of fuzzy sets is set to be equal for all fuzzy controller inputs. In fuzzy control literature, one will find 5, 7, 9, and sometimes 11 or more fuzzy sets defined for each input variable. Fuzzy controllers with seven fuzzy sets are predominant, as they perfectly match a standard human's perception of linguistic values: large (L), medium (M), and small (S). If we extend these qualifications into negative and positive directions and add zero (Z), we get seven most frequently used linguistic values: NL, negative medium (NM), NS, zero (Z), PS, positive medium (PM), and PL.

Although it may seem that a larger number of fuzzy sets will result in a better designed controller, practical experience has proven that the number of fuzzy sets involved is not so important. Quite to the contrary: every fuzzy controller design should tend to solve a control problem with a minimal number of fuzzy sets. For example, by succeeding to solve a problem with a 5 × 5 fuzzy rule table rather than a 7 × 7 fuzzy rule table, the processing of 25 instead of 49 rules will save a lot of computing time.

Every controller input is represented by a linguistic variable whose semantic function (describing the distribution of fuzzy sets over an input domain) and the shape of fuzzy set membership functions must be defined. Various combinations of shapes and distributions of fuzzy sets result in a variety of possible controller structures. Most designers use triangular membership functions with a linear distribution of fuzzy sets [36]. Very often, a polynomial or an exponential law of distribution is adopted, providing a higher density of fuzzy sets near the origin of the domain (Figure 2.14). With such a distribution, the smaller the control error is, the finer the changes of the controller output will be.

Likewise, some fuzzy controller structures have two groups of fuzzy sets with different distributions of fuzzy sets over the same universe of discourse. For an input variable larger than the predefined threshold value, one group is used, while for the input within the threshold bounded area another group takes over. Usually, these two disjunctive control regions define areas of coarse and fine control, respectively.

When defining fuzzy membership functions and semantic functions we should take care to ensure that every quantitative input value is an element of at least one input fuzzy set defined on the input domain. This is not a problem if fuzzy sets with Gaussian or bell-shaped membership functions are used, although, in that case another issue becomes important. Namely, each fuzzy set with a Gaussian or bell-shaped membership function is defined on the whole universe of discourse,

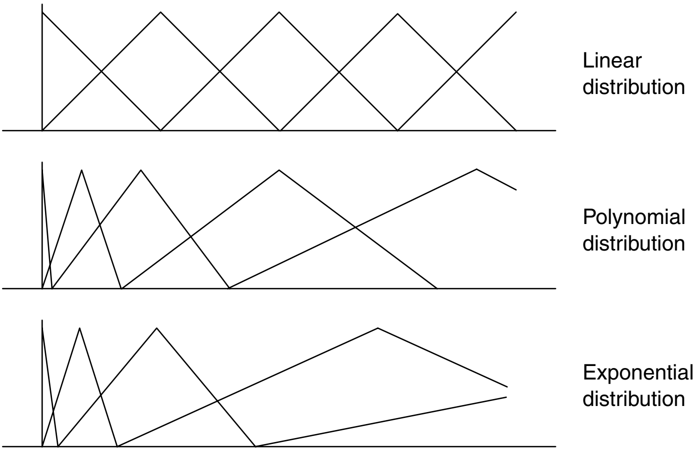


FIGURE 2.14 Various distributions of membership functions.

and thus for any crisp input all fuzzy rules become active and contribute to the crisp controller output, but only a few do so to a considerable degree. To deal with this situation, we can modify the definition of Gaussian and bell-shaped fuzzy sets as follows:

$$\mu_i^x = \mu_i(x) = \begin{cases} \mathrm{e}^{(x-c_i^x)^2/w_i^x}, & \text{for } L_i^x \le x \le R_i^x \\ 0, & \text{for } x < L_i^x,\ \ x > R_i^x \end{cases}$$
$$\mu_i^x = \mu_i(x) = \begin{cases} \dfrac{1}{1-(x-c_i^x)^2}, & \text{for } L_i^x \le x \le R_i^x \\ 0, & \text{for } x < L_i^x,\ \ x > R_i^x \end{cases} \tag{2.26}$$

where L_i^x and R_i^x denote left and right margins of the fuzzy set equal to quantitative values of the variable x which satisfy condition $\mu_i(L_i^x) = \mu_i(R_i^x) = \alpha$; α is a parameter with small arbitrary value taken usually from the range (0.01 to 0.1), and i denotes a fuzzy set index.

The described operation on Gaussian and bell-shaped fuzzy sets is called α-cut. It cuts off values with negligible degrees of membership, achieving, for example, that only two adjacent fuzzy sets overlap, resulting in the reduction of rules that contribute to controller output. This simplifies the calculation of crisp controller output and reduces computing time. Although it is possible, the α-cut operation on triangular and trapezoidal membership functions does not make sense because their domain is already strictly bounded.

The requirement that each crisp value of a linguistic variable must be matched with at least one fuzzy set defined over its universe of discourse brings about conditions that adjacent fuzzy sets should satisfy.

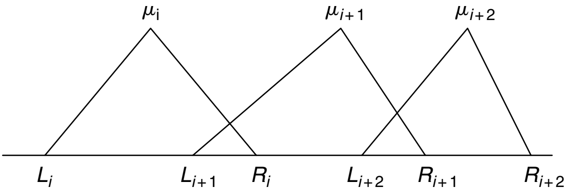


FIGURE 2.15 An arrangement of fuzzy sets that fulfills conditions (2.27) and (2.28).

Referring to (2.26) this condition will always be fulfilled if

$$R_i^x > L_{i+1}^x \tag{2.27}$$

where L_{i+1}^x and R_i^x are left and right margins of adjacent fuzzy sets, that is, intersection point x, $\mu_i(x) = \mu_{i+1}(x)$ must exist. Fuzzy controllers with two fuzzy sets having more than one intersection point are rare and we do not consider this case here. Furthermore, fuzzy sets are usually defined so that every quantitative input value never belongs to more than two fuzzy sets defined on the input domain. In other words, there should be

$$R_i^x < L_{i+2}^x \tag{2.28}$$

In that way the number of rules that contribute to crisp output value of a DISO fuzzy controller can be one, two, or maximally four.

Conditions (2.27) and (2.28) are graphically depicted in Figure 2.15.

As already mentioned, a fuzzy controller will normally have a linear distribution of input fuzzy sets with triangular or trapezoidal membership functions and intersection points at $\mu(x) = 0.5$. Usually, the membership functions are symmetrical, $R_i^x - c_i^x = c_i^x - L_i^x$, and the center and left/right margins of adjacent sets are equal, $c_i^x = R_{i-1}^x = L_{i+1}^x$.

In some instances, the density of fuzzy sets will need to be increased near the origins of related domains in order to give a controller the ability for fine and coarse control.

In case we decide to experiment with other shapes and distributions of fuzzy sets, we shall find out that keeping the same distribution and changing the shape of fuzzy sets from one (e.g., triangular) to another (e.g., trapezoidal, bell-shaped, Gaussian) form will barely affect controller performance. This is true for input fuzzy sets, but it is even truer for output fuzzy sets. Only if we go to extremes, for example, by changing input sets from a triangular shape to a trapezoidal shape with a wide nucleus, we shall induce noticeably different controller behavior. This can be used for increasing controller robustness, which is discussed in more details in Chapter 4. The type of fuzzy implication (e.g., product, minimum or some other T-norm) does not have a significant influence on fuzzy controller performance. Nevertheless, variations of the above mentioned elements contribute to a variety of possible fuzzy controller structures that a designer has at her/his disposal.

2.5 Fuzzy Controller Stability

Although the rigorous mathematical framework of control systems stability theory in some way opposes the vagueness of fuzzy controller properties, stability remains a key issue in fuzzy controller design. The main criticism of fuzzy control is related to its lack of precise stability analysis. That is why efforts have been put into the investigation of various techniques that have a potential to solve the stability issue in fuzzy controlled systems [37,38]. The problem is that an operator, whose experience is the base for a fuzzy control algorithm, can guide a system into the desired state according to some criterion without knowing why the actions taken cause the stable behavior of the system. At the same time the operator is aware that there is a set of manoeuvres which could be a source of instability. From the operator's point of view, the stability region is not strictly defined since actions that lead to the unstable controlled system are described by linguistic values. Being in the position of a fuzzy controller designer, we should take these descriptive justifications of forbidden (unstable) actions into account in the final structure of the controller. In this section we describe techniques for stability analysis of systems controlled by a fuzzy controller.

Some of techniques have roots in stability analyses of nonlinear control systems described with their nonlinear mathematical models. Some of the methods being developed are applicable only to special problematic cases or to strictly determined structures of fuzzy controllers [39]. For example, in Reference 40 it is shown that for a class of fuzzy controllers, which can be described as multilevel nonlinear relay elements, a Nyquist stability criterion can be used for determining the stability region for a fuzzy controlled system. The procedure is also applicable to MIMO control systems. This method's drawback is that it requires knowing the process transfer function, which represents a problem if the main postulate in fuzzy control is imposed: knowledge about system models is a priori incomplete or it does not exist at all.

Stability can be assessed through the analysis of a so-called sliding-mode operation of a fuzzy controller. So, in Reference 41 a fuzzy sliding mode controller is proposed and proof of stability of the controlled system is shown. The fuzzy controller inputs are linear function $s = c \cdot e + e'$ and its first derivative s' (e denotes the difference between reference input and system output). Since function s is defined with only two variables, the hyperplane is represented with line $s = 0$, which yields $c{\cdot}e =-e\prime$. A system with such a fuzzy controller will best able if fuzzy controller outputs are determined in a way that condition $s \cdot s' < 0$ is permanently fulfilled. The validity and practical value of this method was demonstrated on a nonlinear pendulum control problem. The usage of this method does not require the knowledge of a system model, but it does require a qualitative relation of how the control signal acts with regards to function $s \cdot s'$.

On a similar nonlinear system controlled by a variable structure fuzzy controller of the Takagi–Sugeno type with standard inputs e and $e\prime$, in Reference 42 it has been shown that system stability can also be assessed by using the sliding mode control principle. A phase plane has been partitioned into nine regions and the linear function in the consequent part of fuzzy rules depends on the region the phase trajectory is in.

A similar partitioning of the phase plane has been made in Reference 43. The plane was first divided into eight regions and by using the Lyapunov function we first determined a nonlinear function that guarantees asymptotic stability for every region, and then put in the consequent part of the corresponding fuzzy rule.

In Reference 44 the fuzzy sensitivity concept has been used to solve a problem of stability assessment enabling the designer to find out the range of parameter variations for which a given fuzzy controller will maintain stable system performance.

Since fuzzy logic can also be applied for the determination of fuzzy system models, in References 45 and 46 methods are given that prove the stability of fuzzy process models. In both cases a Takagi–Sugeno controller is used and it is shown that by using a Lyapunov function, a fuzzy model will be stable only if linear models in the consequent parts of the rules are stable.

As methods in fuzzy control literature are dominantly based on Lyapunov's theory of stability, here we give the classification of stability according to Lyapunov:

1. Equilibrium point x_e is said to be *stable* if small changes in the initial conditions cause small changes in state trajectory $x(t)$, that is,

$$\forall t_0, \forall \varepsilon > 0; \exists \delta > 0\colon \|x(t_0) - x_e\| < \delta \Rightarrow \|x(t) - x_e\| < \varepsilon, t \geq t_0$$

2. Equilibrium point x_e is said to be *asymptotically stable* if it is stable and if it attracts trajectory $x(t)$, that is,

$$\forall t_0; \exists \delta^* > 0\colon \|x(t_0) - x_e\| < \delta^* \Rightarrow \lim_{t\to\infty} \|x(t) - x_e\| = 0$$

3. Equilibrium point x_e is said to be *globally asymptotically stable* if it is asymptotically stable and δ^* can be arbitrarily large.

Lyapunov's formulation of system stability is based on the observation of energy balance in the system. According to Lyapunov, the system with continuous dissipation of energy will eventually settle into an equilibrium state. Hence, the assessment of system stability is regularly made on the basis of some system energy function, usually called the *Lyapunov function* or the Lyapunov candidate. The point here is that for the system under examination we may construct more such functions (candidates) and examine them before we find the right one which proves system (in)stability. A Lyapunov function, $V(\boldsymbol{x})$, should be continuous and positive definite, $V(\boldsymbol{x}) > 0$ for $\boldsymbol{x} \neq 0$, and its energy level should vanish in the state space origin, $V(\boldsymbol{x}) = 0$ for $\boldsymbol{x} = 0$. When such a candidate function can be defined so that it satisfies the condition $\mathrm{d}V(\boldsymbol{x})/\mathrm{d}t \leq 0$, then according to Lyapunov, such a system is stable. If $\mathrm{d}V(\boldsymbol{x})/\mathrm{d}t < 0$, the system is asymptotically stable. It should be

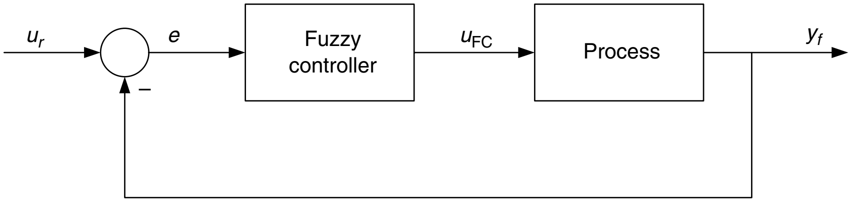


FIGURE 2.16 The SISO system with a fuzzy controller.

noted that the impossibility to define a Lyapunov function for a particular system does not imply that this system is unstable.

The most commonly used Lyapunov function is that of generalized quadratic form $\boldsymbol{x}^{\mathrm{T}}\mathrm{P}x$, where $\boldsymbol{x}$ is a state vector and **P** is a positive definite matrix (if we choose $\mathbf{P} = \mathbf{P}^{\mathrm{T}}$, then **P** will be positive definite if all of its eigenvalues are positive). Having defined the Lyapunov function as a quadratic form, the issue of system stability becomes an issue of finding an appropriate matrix **P**. Since methods for the calculation of **P** are well-described in literature [47], we shall concentrate on a simple case to present some basic ideas, but will show in detail the procedure for testing fuzzy control system stability.

Besides system variables, a Lyapunov function may also contain additional quadratic forms related to fuzzy controller parameters. In that case, stability analysis may become the basis for the synthesis of a tuning algorithm, that is, expressions that are found to guarantee system stability may thereafter be used for tuning controller parameters. Generally very complex, these algorithms are demanding when it comes to practical implementation. It should be noted that even though global asymptotic stability regarding the synthesis of a tuning algorithm is frequently discussed in literature, from a practical point of view, bounded-input–bounded-output stability is much more important.

Let us observe a SISO system with a DISO fuzzy controller, shown in Figure 2.16. The reduction of process models to lower-order differential (difference) equations is regularly made in practice wherever appropriate, as this allows for faster and simpler analysis. This is the reason why we will consider a simple case with a second-order process model. This is also the reason why throughout this book fuzzy controller design is mainly based on second-order process approximation. The results obtained with second-order process can be graphically interpreted (and thus better understood), which makes the choice of such an approach even more logical.

Second-order process is described as

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= f(x_1, x_2) + b \cdot u_{\mathrm{FC}} \\ y_f &= x_1 \end{aligned} \tag{2.29}$$

where x_1 and x_2 are state variables, $f(x_1, x_2)$ is a nonlinear continuous function, $b > 0$ is process gain, and u_{FC} is fuzzy controller output. In general, function f and gain b change with time and therefore only partially describe the real process. For now let us assume that they are time invariant.

Let a fuzzy controller be defined as

$$u_{\mathrm{FC}} = \psi(e, \dot{e}) \tag{2.30}$$

where $\psi(\cdot)$ is a nonlinear fuzzy mapping function. From Figure 2.16 we see that $e = u_r - y_f$. If an autonomous case is considered, $u_r = 0$, then we have $e = -y_f = -x_1$. Let the Lyapunov function be defined as

$$V = x^{\mathrm{T}}\mathbf{P}x = \frac{1}{2}\left(p_{11}x_1^2 + p_{22}x_2^2\right) \tag{2.31}$$

where $\mathbf{P}$ is a diagonal matrix. Then, according to the Lyapunov theory, for the autonomous case of (2.29) to be asymptotically stable we must have

$$\begin{aligned}\dot{V} &= p_{11}x_1\dot{x}_1 + p_{22}x_2\dot{x}_2 = x_2 \cdot (p_{11}x_1 + p_{22}\dot{x}_2)\\ &= x_2 \cdot [p_{11}x_1 + p_{22}(f(x_1, x_2) + b \cdot \psi(-x_1, -x_2))] < 0\end{aligned} \tag{2.32}$$

If we look at Equation (2.32), it becomes clearer how to fulfil the Lyapunov asymptotic stability condition. In order to keep $\mathrm{d}V(x)/\mathrm{d}t \leq 0$, we must somehow provide that the input of variables having the sign opposite to the sign of their derivatives is bigger than the input of variables having the same sign as their derivatives. It is apparent from Equation (2.32) that this goal can be accomplished with a suitable definition of matrix $\mathbf{P}$ (in our case, of p_{11} and p_{22}).

Relation (2.32) can be expressed in the form of two conditions that ensure the asymptotic stability of the system (2.29) when controller (2.30) is used:

$$\begin{aligned} x_2 < 0 \quad &\Rightarrow \quad p_{11}x_1 > -p_{22}[f(x_1, x_2) + b \cdot \psi(-x_1, -x_2)]\\ x_2 > 0 \quad &\Rightarrow \quad p_{11}x_1 < -p_{22}[f(x_1, x_2) + b \cdot \psi(-x_1, -x_2)]\end{aligned} \tag{2.33}$$

These two conditions are graphically presented in Figure 2.17. Nonlinear function $f(\cdot)$ and controller $\psi(\cdot)$ form a surface over a two-dimensional state space. This surface must be in a particular relation with surface $g(x_1, x_2) = p_{11}x_1$, for system (2.29) to be stable. If function $f(\cdot)$ is known, once the fuzzy controller is designed, parameters p_{11} and p_{22} should be determined in order to find out whether conditions (2.33) are fulfilled. For linear function $f(\cdot)$, the calculation of $\mathbf{P}$ can be done by solving linear matrix inequality, which is straightforward for a second-order system. Actually, matrix $\mathbf{P}$ "rotates" the surface shown in Figure 2.17 around the equilibrium. In case there exists a positive definite matrix $\mathbf{P}$ that adjusts the surface according to the Lyapunov criterion (2.33), the system is asymptotically stable.

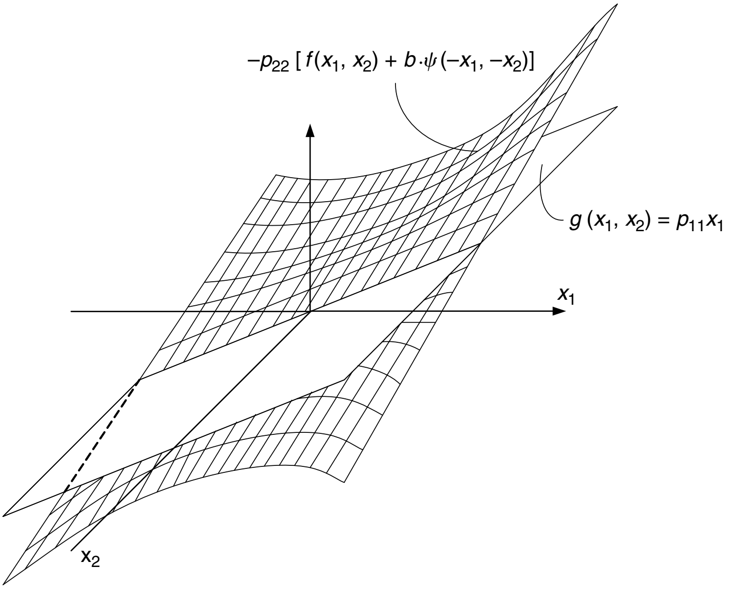

$-p_{22}\,[f(x_1, x_2) + b\cdot\psi(-x_1, -x_2)]$
$g\,(x_1, x_2) = p_{11}x_1$
x_1
x_2

FIGURE 2.17 A graphical representation of stability conditions for a second-order process.

A problem could arise when $f(\cdot)$ is nonlinear (which is more or less always the case). In some situations, depending on the type of nonlinearities in $f(\cdot)$, it is very difficult, if not impossible, to find matrix **P**. In that case, a generalized quadratic form is not suitable and a different Lyapunov function candidate should be chosen. The alternative is partitioning the state space in several regions. Then, linear process approximations can be used in Equation (2.33) for examining stability in the region where a particular linear model is valid (indirect Lyapunov method). Since system inputs and system outputs are bounded in practice, universes of discourse of x_1 and x_2 are also bounded. This means that a number of linearized regions can be restricted to a reasonable number.

Conditions (2.33) can be used not only for the analysis of system stability, but also for the fuzzy controller design. If we rewrite (2.33) in the form

$$\begin{aligned} x_2 < 0 \;&\Rightarrow\; \frac{(p_{11}/p_{22})x_1 + f(x_1, x_2)}{b} > -[\psi(-x_1, -x_2)] \\ x_2 > 0 \;&\Rightarrow\; \frac{(p_{11}/p_{22})x_1 + f(x_1, x_2)}{b} < -[\psi(-x_1, -x_2)] \end{aligned} \tag{2.34}$$

then, having defined the structure of the fuzzy controller and knowing function $f(\cdot)$, we can determine fuzzy rules.

Example 2.7 Fuzzy controller stability — the Lyapunov approach.

We are investigating a nonlinear system described by the following equations:

$$\begin{aligned}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= -0.3x_1(1+x_1^2) - 3.9x_2(1+x_2^2) + b \cdot u_{\mathrm{FC}} \\
y &= x_1
\end{aligned}$$

The equations represent a mechanical system with damper and spring reactive forces that nonlinearly depend on a speed, x_2, and a position, x_1, respectively. The system equilibrium point is $\boldsymbol{x}_{\mathrm{e}} = [0\ 0]^{\mathrm{T}}$.

Let us first examine open-loop stability, that is, $u_{\mathrm{FC}} = 0$. According to Equation (2.33)

$$\begin{aligned}
x_2 < 0 \quad &\Rightarrow \quad p_{11}x_1 > p_{22}[0.3x_1(1+x_1^2) + 3.9x_2(1+x_2^2)] \\
x_2 > 0 \quad &\Rightarrow \quad p_{11}x_1 < p_{22}[0.3x_1(1+x_1^2) + 3.9x_2(1+x_2^2)]
\end{aligned}$$

A graphical representation of the nonlinear function $f(x_1, x_2)$ with an enlarged area around the equilibrium (phase plane origin) is given in Figure 2.18.

Due to the character of nonlinearities, a generalized quadratic form of the Lyapunov function cannot be used, so we must linearize the system around the equilibrium. Linearization gives

$$\begin{aligned}
x_2 < 0 \quad &\Rightarrow \quad p_{11}x_1 > p_{22}[0.3x_1 + 3.9x_2] \\
x_2 > 0 \quad &\Rightarrow \quad p_{11}x_1 < p_{22}[0.3x_1 + 3.9x_2]
\end{aligned}$$

For $p_{11} = 1$ and $p_{22} = 1/0.3$ we have

$$\begin{aligned}
x_2 < 0 \quad &\Rightarrow \quad x_1 > [x_1 + 13x_2] \\
x_2 > 0 \quad &\Rightarrow \quad x_1 < [x_1 + 13x_2]
\end{aligned}$$

Hence, the system is asymptotically stable around the equilibrium. The system response for initial conditions $x_0 = [1\ 0]^{\mathrm{T}}$ is shown in Figure 2.19.

Let us now define fuzzy controller structure. Each input is partitioned into five linguistic values that correspond to five linearly distributed fuzzy sets: NL, NS, Z, PS, and PL. All fuzzy sets have triangular membership functions. COG defuzzification method is used.

We will design a fuzzy controller based on a linear model that is valid around the equilibrium (operating point (0,0)); the universe of discourse for x_1 is $[-1, 1]$ while for x_2 we have set a region $[-0.5, 0.5]$. Input scaling factors are calculated to map inputs into range $[-1, 1]$, thus, $K_{x1} = 1$ and $K_{x2} = 2$. Using inequalities (2.34)

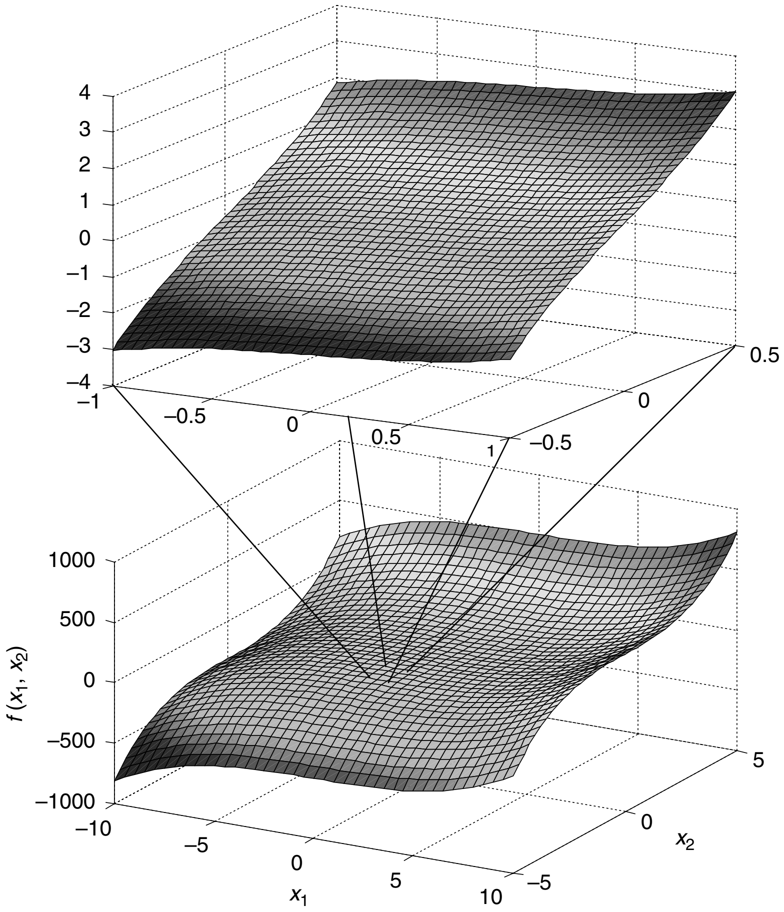


FIGURE 2.18 A graphical representation of the nonlinear function from Example 2.7.

(assuming $b = 1$) for control surface design gives

$$x_2 < 0 \Rightarrow \frac{p_{11}}{p_{22}}x_1 - 0.3x_1 - 3.9x_2 > -[\psi(-x_1, -x_2)]$$

$$x_2 > 0 \Rightarrow \frac{p_{11}}{p_{22}}x_1 - 0.3x_1 - 3.9x_2 < -[\psi(-x_1, -x_2)]$$

Although each pair (x_1, x_2) should satisfy the above conditions, we will calculate control surface values only for the centers of corresponding input fuzzy sets. As the first approximation of fuzzy control surface we use a linear function

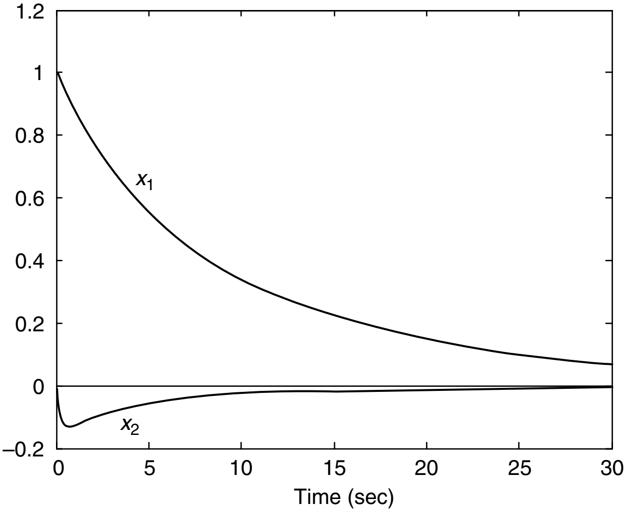


Figure 2.19 The response of the system from Example 2.7 for $\boldsymbol{x}_0 = [1\ 0]^T$.

$\psi(-x_1, -x_2) \approx k_1x_1 + k_2x_2$, which yields

$$x_2 < 0 \;\Rightarrow\; \frac{p_{11}}{p_{22}}x_1 > (0.3 - k_1)x_1 + (3.9 - k_2)x_2$$

$$x_2 > 0 \;\Rightarrow\; \frac{p_{11}}{p_{22}}x_1 < (0.3 - k_1)x_1 + (3.9 - k_2)x_2$$

We can see from the above equations that as far as $k_1 < 0.3$ and $k_2 < 3.9$, the system is stable according to Lyapunov criteria. In order to move further from the unstable region, we choose $k_1 = -1$ and $k_2 = 1$, hence, $\psi(-x_1, -x_2) \approx -x_1 + x_2$.

In Tables 2.4 and 2.5, the numerical values of points lying on the linear control surface, corresponding to each pair of centers of input membership functions (values shown in parenthesis) are given. As there are 13 different values, in order to obtain a good mimic we define 13 linguistic values for controller output. In the next step of fuzzy controller design we associate each value from the table with the center of one of the output fuzzy sets that have triangular membership functions: PLL, PL, PMM, PM, PSS, PS, Z, NS, NSS, NM, NMM, NL, and NLL (Figure 2.20).

System responses with linear control surface and with a fuzzy controller are shown in Figure 2.21. It may be seen that the system controlled by the fuzzy controller is stable.

However, we have yet to address a problem related to the stability of the system in case we move further from the equilibrium. From Figure 2.18 we see that position can take value from the interval [−10 cm, 10 cm] while the speed domain is [−5 cm/sec, 5 cm/sec]. Since the nonlinear function $f(\cdot)$ is known,

TABLE 2.4
The Numerical Values of Points Lying on the Linear Control Surface

x_2	x_1 NL (−1)	NS (−0.5)	Z (0)	PS (0.5)	PL (1)
NL (−0.5)	0.5	0	−0.5	−1	−1.5
NM (−0.25)	0.75	0.25	−0.25	−0.75	−1.25
Z (0)	1	0.5	0	−0.5	−1
PM (0.25)	1.25	0.75	0.25	−0.25	−0.75
PL (0.5)	1.5	1	0.5	0	−0.5

TABLE 2.5
The Corresponding Fuzzy Sets to the Values Presented in Table 2.4

x_2	x_1 NL	NS	Z	PS	PL
NL	PSS	Z	NSS	NMM	NLL
NM	PM	PS	NS	NM	NL
Z	PMM	PSS	Z	NSS	NMM
PM	PL	PM	PS	NS	NM
PL	PLL	PMM	PSS	Z	NSS

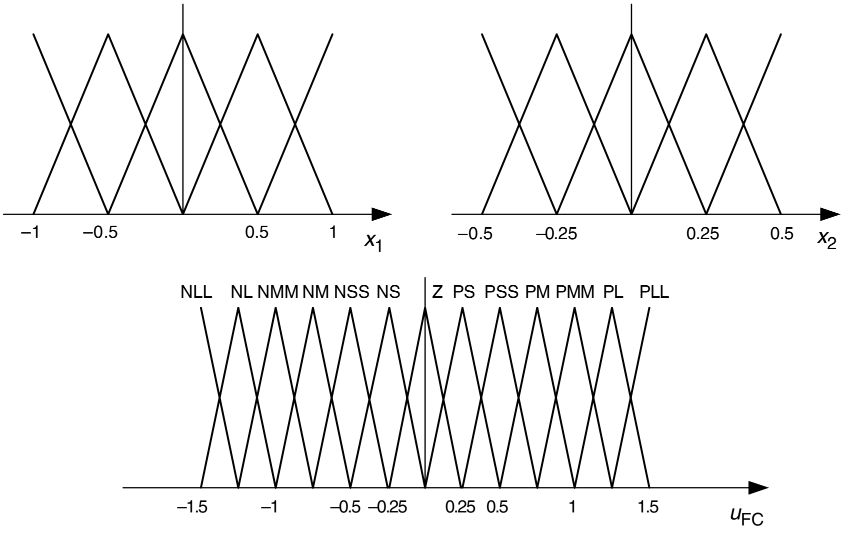


FIGURE 2.20 Fuzzy membership functions for the controller from Example 2.7.

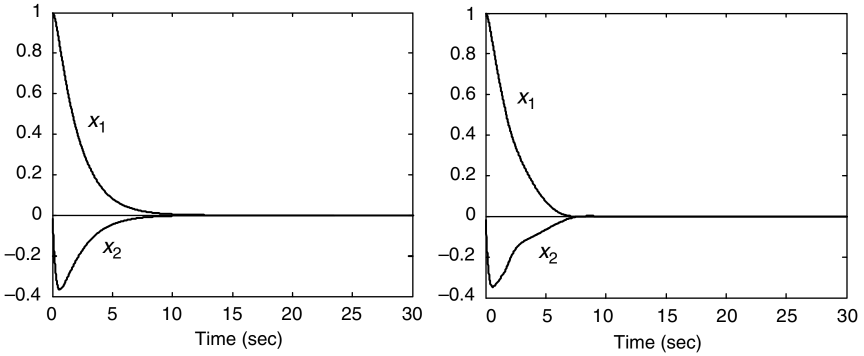


FIGURE 2.21 Response of the system from Example 2.7 for $x_0 = [1\ 0]^T$ with (a) a linear controller and (b) a fuzzy controller.

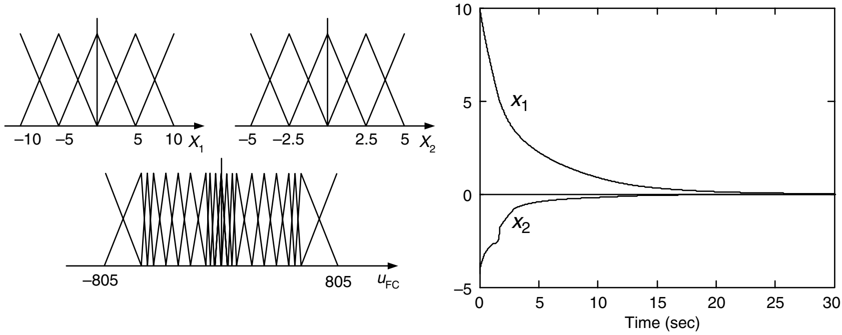


FIGURE 2.22 Fuzzy membership functions and the system response for the fuzzy controller from Example 2.7.

we can set

$$\psi(-x_1, -x_2) = 0.3x_1(1 + x_1^2) + 3.9x_2(1 + x_2^2) + k_1x_1 + k_2x_2$$

In that case the entire phase plane is covered by the control surface and the stability of the system depends on k_1 and k_2. Using the same principle as for the linearized case, we calculate centers of output fuzzy sets. The membership functions obtained for $k_1 = -1$ and $k_2 = 1$ are shown in Figure 2.22.

Due to the nonlinearities in the system, we can see that the distribution of output membership functions is nonlinear. The response of the system for $x_0 = [1\ 0\ 0]^T$ is shown in Figure 2.22.

We will now go on to describe a procedure that exploits geometric properties of state space in the investigation of system stability. Although this method is cumbersome, its practical value becomes clear in the situation when state space is reduced to a phase plane, which is the case in a second-order system. Then

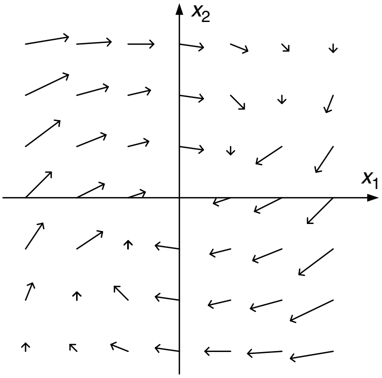


FIGURE 2.23 Phase plane velocity vectors.

phase plane analysis offers well-known procedures (especially in case $f(\cdot)$ is linear) for the determination of system stability.

Equations (2.29) describe the motion of a point along the phase trajectory in the phase plane x_1–x_2. The vector of the magnitude

$$v = \sqrt{\dot{x}_1^2 + \dot{x}_2^2} = \sqrt{x_2^2 + (f(x_1, x_2) + b \cdot u_{\text{FC}})^2} \tag{2.35}$$

and the direction

$$\vartheta = \arctan \frac{\dot{x}_2}{\dot{x}_1} = \arctan \frac{f(x_1, x_2) + b \cdot u_{\text{FC}}}{x_2} \tag{2.36}$$

can represent the velocity of this motion.

Velocity vectors are usually represented by arrows on the phase plane (Figure 2.23).

The examination of the magnitude and the direction of the velocity vectors can tell us whether the motion of the point on the phase plane is stable. The analysis gives characteristic objects, *points*, and *curves*. The inclusion of $\dot{x}_1 = 0$ in Equation (2.36) gives a so-called *isocline* (a set of points where ϑ is constant) that coincides with the abscissa axis of the phase plane, that is, with the straight line $x_2 = 0$. In that case

$$v = f(x_1, 0) + b \cdot u_{\text{FC}} \tag{2.37}$$

Another isocline is obtained for $\dot{x}_2 = 0$, which yields

$$v = x_2 \tag{2.38}$$

This isocline can be found as the solution of equation

$$f(x_1, x_2) + b \cdot u_{\text{FC}} = 0 \tag{2.39}$$

According to Equation (2.36), all velocity vectors in Equation (2.37) have the same direction $\vartheta = \pi/2$, that changes to $\vartheta = -\pi/2$; while the direction for the second isocline is $\vartheta = 0$, which changes to $\vartheta = \pi$. The change of direction takes place in the intersection points of two isoclines

$$f(x_1, 0) + b \cdot u_{\mathrm{FC}} = 0 \tag{2.40}$$

Intersection points are called *singular points* as both velocity components become equal to 0. This means that the state of the system cannot change without an external disturbance — singular points represent the *equilibrium* of the system. For the systems described by the second-order linear model, only one such point exists (there is a special degenerated case when such systems have an infinite number of equilibrium points). If the system is moved from the equilibrium, phase velocity vectors describe the motion of the point along the trajectory. This motion may be considered to be driven by the energy conservation law. In that picture, variables x_1 and x_2 correspond to potential and kinetic energy, respectively. As energy may reside in one of these two forms, for $x_1 = 0$ we have potential energy that vanishes and kinetic energy is at its maximum, while for $x_2 = 0$ all energy takes the potential form. While moving in the phase plane, the total energy of the point can (a) dissipate — which guides the point toward equilibrium, (b) remain constant — which produces cyclic motion of the point around equilibrium, or (c) grow — which forces the point to move away from the equilibrium. Since the system is nonlinear, the motion may not be so smooth, that is, energy can grow in some parts of the phase plane, while in others it dissipates or remains constant.

The characteristic positions of isoclines $\dot{x}_1 = 0$ and $\dot{x}_2 = 0$ for the second-order system (2.29) with a smooth function $f(\cdot)$ and one equilibrium, as well as for an autonomous case, $u_{\mathrm{FC}} = 0$, are shown in Figure 2.24. The arrows represent the directions of the velocity vectors.

In Equation (2.29) function $f(\cdot)$ can be considered as the measure of the change in the system's kinetic energy. Kinetic energy reaches extreme values on the isocline. Hence, by knowing the gradient of function $f(\cdot)$ on the isocline, we are able to say whether a particular extreme represents the maximum or the minimum of kinetic energy. The attainment of kinetic energy maximum actually means that kinetic energy is bounded, which further entails the limitation of the rise in the system's potential energy, thus making the overall system stable. It must be noted, however, that bounded kinetic energy does not necessarily mean that the system is stable, for example, for $f(x_1, x_2) = 0$ kinetic energy is constant and thus bounded, yet the system is unstable.

Isoclines shown in Figures 2.24([a]–[c]) are associated with stable systems, while other isoclines represent systems with unstable behavior. Let us consider case (a), which represents the degenerated situation mentioned earlier. Characteristic isoclines coincide with axis x_1. Hence, phase plane point $A(x_{1\mathrm{A}}, 0)$, that lies on axis x_1, belongs to both isoclines. As both $\dot{x}_1$ and $\dot{x}_2$ do not depend on x_1, the system, once positioned in point A, will not move — point A represents the equilibrium of the system. Thus, the system has an infinite number of equilibrium points.

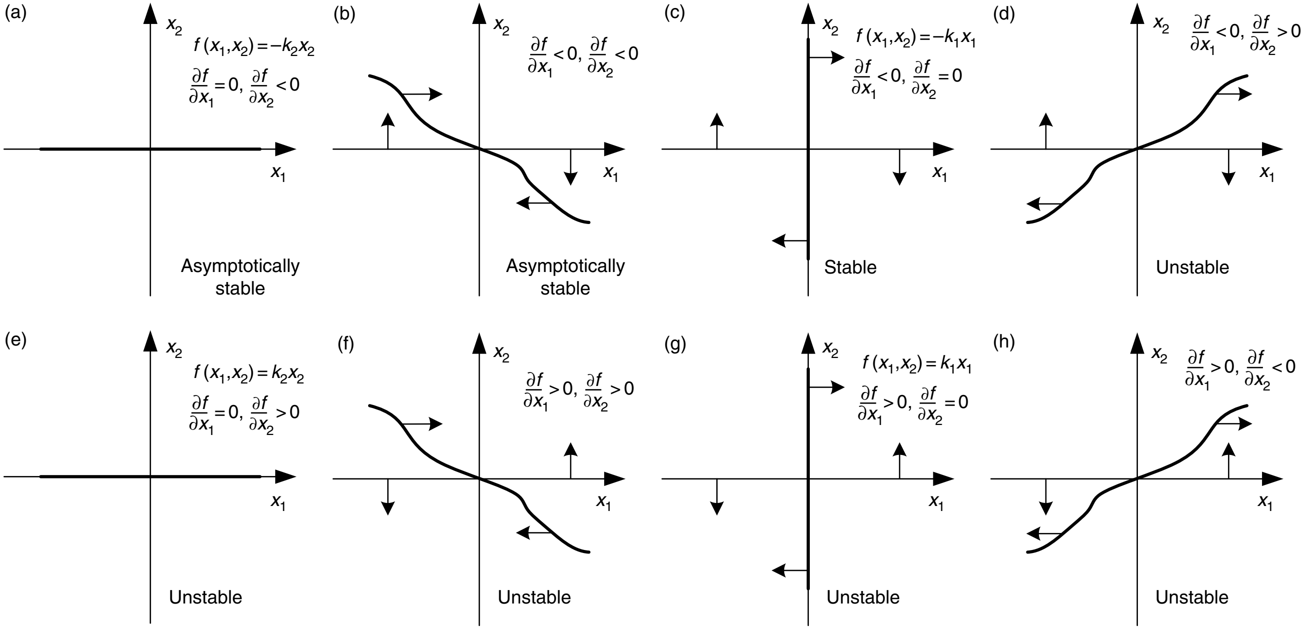

(a)
x_2
$f(x_1,x_2) = -k_2x_2$
$\frac{\partial f}{\partial x_1} = 0, \frac{\partial f}{\partial x_2} < 0$
x_1
Asymptotically stable
(b)
x_2
$\frac{\partial f}{\partial x_1} < 0, \frac{\partial f}{\partial x_2} < 0$
x_1
Asymptotically stable
(c)
x_2
$f(x_1,x_2) = -k_1x_1$
$\frac{\partial f}{\partial x_1} < 0, \frac{\partial f}{\partial x_2} = 0$
x_1
Stable
(d)
x_2
$\frac{\partial f}{\partial x_1} < 0, \frac{\partial f}{\partial x_2} > 0$
x_1
Unstable
(e)
x_2
$f(x_1,x_2) = k_2x_2$
$\frac{\partial f}{\partial x_1} = 0, \frac{\partial f}{\partial x_2} > 0$
x_1
Unstable
(f)
x_2
$\frac{\partial f}{\partial x_1} > 0, \frac{\partial f}{\partial x_2} > 0$
x_1
Unstable
(g)
x_2
$f(x_1,x_2) = k_1x_1$
$\frac{\partial f}{\partial x_1} > 0, \frac{\partial f}{\partial x_2} = 0$
x_1
Unstable
(h)
x_2
$\frac{\partial f}{\partial x_1} > 0, \frac{\partial f}{\partial x_2} < 0$
x_1
Unstable

FIGURE 2.24 The characteristic positions of isoclines $\dot{x}_1 = 0$ and $\dot{x}_2 = 0$ for a second-order system.

As long as the gradient of function $f(\cdot)$ is negative, the system moves from any point on the phase plane toward the equilibrium on axis x_1.

In case (b), isocline $\dot{x}_2 = 0$ is represented by the curve extended through the second and the fourth quadrants. Let us again begin with phase plane point $A(x_{1A}, 0)$, that lies on axis x_1. The velocity vector in this point is defined as

$$v_A^{\mathrm{T}} = [\dot{x}_{1A} \quad \dot{x}_{2A}] = [0 \quad f(x_{1A}, 0)] \tag{2.41}$$

$$\vartheta_A = \frac{\pi}{2} \tag{2.42}$$

which means that point A initially has the velocity of magnitude $v_A = f(x_{1A}, 0)$ and direction $\vartheta_A = \pi/2$ which coincides with the direction of the positive axis x_2.

As the point starts to move in the direction of ϑ_A, x_2 starts to increase. Since we are investigating model (2.29), where $\dot{x}_1 = x_2$, the velocity vector component also increases in the direction of x_1. Due to the negative gradient of $f(\cdot)$ with respect to x_2, the velocity vector component starts to decrease in the direction of x_2. According to (2.36), these changes of velocity vector components cause ϑ_A to decline. Eventually, the point arrives and intersects axis x_2 in point $B(0, x_{2B})$ at speed $v_B^{\mathrm{T}} = [\dot{x}_{1B} \;\; \dot{x}_{2B}] = [x_{2B} \;\; f(0, x_{2B})]$ and direction ϑ_B. Since the function $f(\cdot)$ changes the sign on the isocline situated in the second and the fourth quadrants, we have $-(\pi/2) < \vartheta_B < 0$. This means that the next time the system intersects x_1 axis (the state with zero kinetic energy), the remaining potential energy will be lower than it was at the beginning of movement, that is, in point A. Decreasing energy level indicates that case (b) system is stable.

Other characteristic cases can be studied in the same manner. The analysis of systems that have isoclines which pass through more than two quadrants or have more than one equilibrium can be rather complex. However, the above discussion and Equation (2.39) show that magnitudes and directions of velocity vectors as well as positions of equilibrium points and isoclines can be changed by using a fuzzy controller. In the following example we shall investigate the stability of a system with an unstable equilibrium and afterwards we shall describe the design of a fuzzy controller that can stabilize such a system.

Example 2.8 Fuzzy controller stability — phase plane approach.

Let us consider a system described by the following equations:

$$\begin{aligned}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= -x_1(x_1^2 - 1) - x_2 + b \cdot u_{\mathrm{FC}} \\
y &= x_1
\end{aligned}$$

First we will investigate the stability of the autonomous system. Isocline $\dot{x}_2 = 0$ and velocity vectors are shown in Figure 2.25. As may be seen, the isocline is part

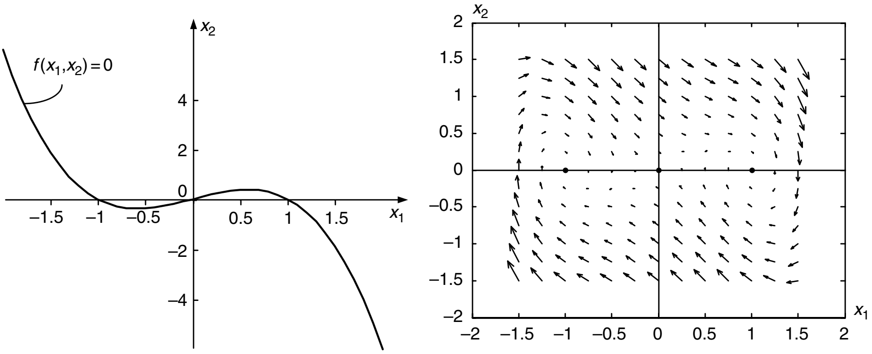


FIGURE 2.25 Isocline $\dot{x}_2 = 0$ and velocity vectors for the system from Example 2.8.

of four quadrants and intersects axis x_1 (coinciding with isocline $\dot{x}_1 = 0$) in three points. Thus, the system has three equilibriums, (1,0), $(-1, 0)$, and (0,0). According to the discussion related to isocline position and its gradients, the last equilibrium is unstable since for that point the gradient of function $f(x_1, x_2)$ with respect to x_1 is positive. The other two equilibriums are stable as both gradients of $f(x_1, x_2)$ are negative.

System response for the initial condition $x_0 = [2\ 0]^{\mathrm{T}}$ is shown in Figure 2.26. After the completion of transition, the system resides in equilibrium $x_{\mathrm{e}} = [1\ 0]^{\mathrm{T}}$. Our goal is to design a fuzzy controller which would move the system toward a single stable equilibrium in the origin of the phase plane and keep it there. According to Equation (2.39), if we set

$$u_{\mathrm{FC}} = \psi(x_1, x_2) = \frac{1}{b}[-f(x_1, x_2) + k_1 x_1 + k_2 x_2]$$

then the isocline $\dot{x}_2 = 0$ becomes a straight line

$$x_2 = -\frac{k_1}{k_2} x_1$$

A system with such an isocline has the equilibrium in the phase plane origin. By properly setting coefficients k_1 and k_2, we can position the isocline in the second and the fourth quadrant and at the same time provide negative gradients over the whole phase plane, making the system stable. For this purpose, we choose $k_1 = -1$ and $k_2 = -1$.

The same procedure as the one in Example 2.7 is used for fuzzy controller design. Domains of input variables, $x_1 = [-2, 2]$ and $x_2 = [-2, 2]$, are partitioned in seven fuzzy sets each: NL, NM, NS, Z, PS, PM, and PL. Numerical values of points at the control surface for each pair of centers (values denoted in parentheses) of input membership functions are given in Tables 2.6 and 2.7.

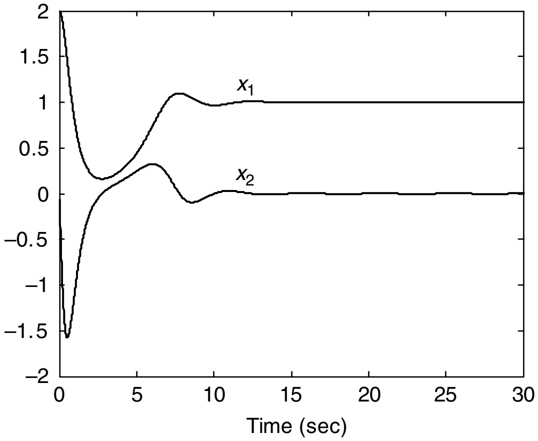


FIGURE 2.26 The response of the system from Example 2.8 for initial condition $x_0 = [2\,0]^T$.

TABLE 2.6
Numerical Values for a Nonlinear Function Shown in Example 2.8

	x_1						
x_2	NL (−2)	NM (−1.33)	NS (−0.67)	Z (0)	PS (0.67)	PM (1.33)	PL (2)
NL (−2)	12	6.37	4.29	4	3.7	1.63	−4
NM (−1.33)	10.67	5.04	2.96	2.67	2.37	0.3	−5.33
NS (−0.67)	9.33	3.7	1.63	1.33	1.04	−1.04	−6.67
Z (0)	8	2.37	0.3	0	−0.3	−2.37	−8
PS (0.67)	6.67	1.04	−1.04	−1.33	−1.63	−3.7	−9.33
PM (1.33)	5.33	−0.3	−2.4	−2.67	−2.96	−5.04	−10.67
PL (2)	4	−1.63	−3.7	−4	−4.29	−6.37	−12

Although Tables 2.6 and 2.7 contain 49 different numerical values, only nine fuzzy sets are used for partitioning the output universe of discourse. The system response attained with a fuzzy controller and its membership functions are shown in Figure 2.27. The system response is stable and the equilibrium is located in the origin. The nonlinear character of the fuzzy controller may be easily recognized from the rule table.

At the beginning of this section we have mentioned that by using linguistic values the operator can define not only stabilizing (allowed) actions, but also destabilizing (forbidden) control actions. The question is: if we replace the crisp mathematical definition of Lyapunov stability conditions (2.33) with linguistic terms, can we still treat these conditions as a valid test of stability? The answer to this query was proposed in Reference 48. Instead of using numbers for calculating

TABLE 2.7
A Rule Table of a Fuzzy Controller Designed in Example 2.8

	x_1						
x_2	**NL**	**NM**	**NS**	**Z**	**PS**	**PM**	**PL**
NL	PLL	PLL	PM	PM	PM	PS	NS
NM	PLL	PM	PS	PS	PS	Z	NM
NS	PL	PM	PS	PS	PS	NS	NL
Z	PL	PS	Z	Z	Z	NS	NL
PS	PM	PS	NS	NS	NS	NM	NL
PM	PM	Z	NS	NS	NS	NM	NLL
PL	PS	NS	NM	NM	NM	NLL	NLL

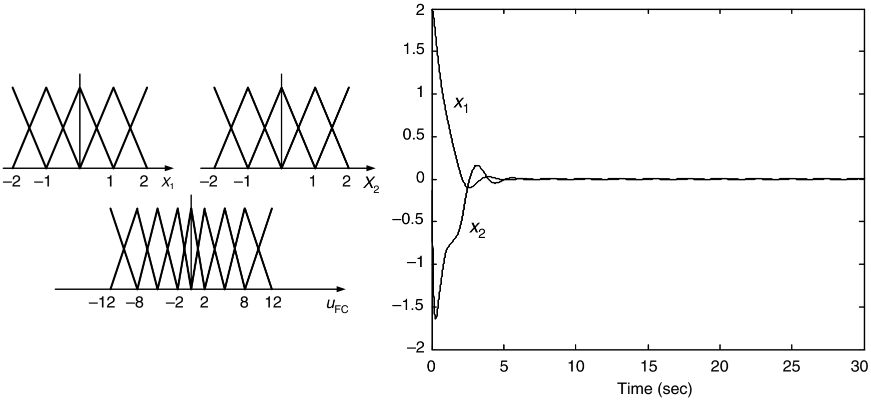


FIGURE 2.27 Fuzzy membership functions and the system response for the fuzzy controller from Example 2.8.

the derivative of a Lyapunov function, the authors adopted Zadeh's *computing with words* (CW) [49]. It has been shown that only partial knowledge of the system was enough for the design of a simple fuzzy controller that stabilizes a process (an inverted pendulum was used as an example). Let us recall the Lyapunov stability criterion for a second order system

$$\dot{V} = p_{11}x_1\dot{x}_1 + p_{22}x_2\dot{x}_2 = x_2 \cdot (p_{11}x_1 + p_{22}\dot{x}_2) < 0 \tag{2.43}$$

If a *pendulum angle* and a *change in pendulum angle* are chosen as process variables x_1 and x_2, then from the inverted pendulum model we know that $\dot{x}_2$ is proportional to controller output u_{FC}, which stands for the force applied to the cart. By setting $p_{11} = p_{22} = 1$, authors in Reference 48 examined signs of x_1 and x_2 and proposed a fuzzy controller, which has only four rules (Table 2.8) and which fulfills inequality (2.43).

TABLE 2.8
Control Rules

x_1	x_2	u_{FC} ($\sim\dot{x}_2$)	$\dot{V}$
Positive	Positive	Negative big	Negative
Positive	Negative	Zero	Negative
Negative	Positive	Zero	Negative
Negative	Negative	Positive big	Negative

From Bogdan, S., Kovačić, Z., and Punčec, M., *IEEE 4th Int. Conf. Intell. Syst. Design Appl.*, 271–276, 2004. With permission.

Linguistic values for state variables are *negative* and *positive* while controller output is partitioned in *negative big*, *zero*, and *positive big*. Since the proposed fuzzy controller does not take into account magnitudes of linguistic variables, it cannot provide fine changes in controller output. Furthermore, by using only the sign, the number of rules is decreased from the number we would have were we to use the available information about the process. As the domains of state variables and controller output are known in advance, there is no reason why we should not partition these domains in more linguistic values, thus providing more rules and finer controller output. The extension of a CW design in this direction was proposed in Reference 50. Instead of using only signs of state variables, the authors integrated their magnitudes in the form of *fuzzy numbers*. Before we proceed with the description of the proposed method, few definitions related to fuzzy numbers and their arithmetic are given.

A fuzzy number is a special interpretation of a fuzzy set that represents a set of "numbers close to ς" where ς is the number being fuzzified. We denote a fuzzy number as $\tilde{\varsigma}$.

Definition 2.16 (A fuzzy number) A fuzzy number $\tilde{\varsigma}$ is a fuzzy set that has a bounded domain and a convex and normal membership function $\mu_\varsigma(x)$, that is,

$$\mu_\varsigma[\lambda x_1 + (1 - \lambda x_2)] \geq \min[\mu_\varsigma(x_1), \mu_\varsigma(x_2)], \quad \forall x_1, x_2 \in X, \lambda \in [0, 1]$$
$$\sup[\mu_\varsigma(x)] = 1$$

The most commonly used form of a fuzzy number is the *triangular fuzzy number (L − R fuzzy number).* As its name says, the $L - R$ fuzzy number has a triangular membership function and is written as $\tilde{\varsigma} = \langle L, c, R\rangle$, where L is the left margin, c is the center, and R is the right margin of the number. A general procedure that provides an extension of crisp mathematical expressions to fuzzy domains is called the *extension principle* [51]. It states that having a function $y = f(x)$ and a fuzzy number $\tilde{a} = \{(\mu_a(x), x) \colon x \in X\}$, then

$$\tilde{b} = f(\tilde{a}) = \{(\mu_a(y), y) \colon y \in X\} \tag{2.44}$$

In other words, the outcome of a mathematical expression (2.44) is a fuzzy number obtained by computation of the image of the interval while the membership function is carried through. In case $f(x)$ is a many-to-one mapping, $\mu_a(y)$ is calculated as the *maximum* of multiple entries. For example, if $\tilde{a} = \{(0.3, -1), (0.7, 0), (1, 1), (0.6, 2), (0.2, 3)\}$ and $f(x) = x^2$, since $f(x)$ is a many-to-one mapping (notice that the first and the third element squared get the same value 1 with different membership degrees), then $\tilde{b} = \{(\max[0.3, 1], 1), (0.7, 0), (0.6, 4), (0.2, 9)\} = \{(0.7, 0), (1, 1), (0.6, 4), (0.2, 9)\}$.

The implementation of the extension principle to arithmetical operations gives the following definition.

Definition 2.17 (The arithmetic of fuzzy numbers) Let $\tilde{a}$ and $\tilde{b}$ be two fuzzy numbers and let "$\circ$" denote any of four basic arithmetic operations $(+, -, \times, /)$. Then

$$\mu_{a\,\circ\,b}(c) = \sup_{c=a\,\circ\,b} \min\{\mu_a(x), \mu_b(y)\} \tag{2.45}$$

In α-cuts notation (2.45) gives

$$\begin{aligned} a_\alpha + b_\alpha &= [\underline{a}_\alpha + \underline{b}_\alpha, \overline{a}_\alpha + \overline{b}_\alpha] \\ a_\alpha - b_\alpha &= [\underline{a}_\alpha - \overline{b}_\alpha, \overline{a}_\alpha - \underline{b}_\alpha] \\ a_\alpha \times b_\alpha &= [\min(\overline{a}_\alpha\overline{b}_\alpha, \overline{a}_\alpha\underline{b}_\alpha, \underline{a}_\alpha\overline{b}_\alpha, \underline{a}_\alpha\underline{b}_\alpha), \\ &\qquad \times \max(\overline{a}_\alpha\overline{b}_\alpha, \overline{a}_\alpha\underline{b}_\alpha, \underline{a}_\alpha\overline{b}_\alpha, \underline{a}_\alpha\underline{b}_\alpha)] \\ a_\alpha / b_\alpha &= [\underline{a}_\alpha, \overline{a}_\alpha] \times \left[\frac{1}{\overline{b}_\alpha}, \frac{1}{\underline{b}_\alpha}\right] \end{aligned} \tag{2.46}$$

where $a_\alpha = \{x\colon \mu_a(x) \geq \alpha\}$ and $b_\alpha = \{x\colon \mu_b(x) \geq \alpha\}$ are α-cuts of fuzzy numbers $\tilde{a}$ and $\tilde{b}$, and $\overline{a}_\alpha = \sup(a_\alpha), \underline{a}_\alpha = \inf(a_\alpha), \overline{b}_\alpha = \sup(b_\alpha), \underline{b}_\alpha = \inf(b_\alpha)$. The result of an arithmetic operation is obtained as the union of all α-cuts.

Let $\tilde{a} = \{(0.5, 1), (1, 2), (0.5, 3)\}$ and $\tilde{b} = \{(0.25, 8), (1, 9), (0.25, 10)\}$. Then their sum is calculated as

$$\begin{aligned} \tilde{a} + \tilde{b} = \{&(0.25, 9), (0.25, 10), (0.25, 11), (0.5, 10), (1, 11), (0.5, 12), (0.25, 11), \\ &(0.25, 12), (0.25, 13)\} \\ = \{&(0.25, 9), (0.5, 10), (1, 11), (0.5, 12), (0.25, 13)\} \end{aligned}$$

Division is undefined for 0 as an element of fuzzy set $\tilde{b}$. If $L - R$ fuzzy numbers are used in operations, the results of addition and subtraction are also $L - R$

numbers. Moreover, in that case if $\tilde{a} = \langle L_a, c_a, R_a \rangle$ and $\tilde{b} = \langle L_b, c_b, R_b \rangle$ then

$$
\begin{aligned}
\tilde{c} &= \tilde{a} + \tilde{b} = \langle L_a + L_b, c_a + c_b, R_a + R_b \rangle \\
\tilde{c} &= \tilde{a} - \tilde{b} = \langle L_a - R_b, c_a - c_b, R_a - L_b \rangle
\end{aligned} \tag{2.47}
$$

Multiplication and division of $L - R$ fuzzy numbers result in a fuzzy number that is not a $L - R$ number. However, for engineering purposes, multiplication and division can be approximated with relations defined in Reference 52

$$
\begin{aligned}
\tilde{c} &= \tilde{a}\tilde{b} = \langle \min\left(L_aL_b, L_aR_b, R_aL_b, R_aR_b\right), c_ac_b, \max\left(L_aL_b, L_aR_b, R_aL_b, R_aR_b\right) \rangle \\
\tilde{c} &= \tilde{a}/\tilde{b} = \left\langle \min\left(\frac{L_a}{L_b}, \frac{L_a}{R_b}, \frac{R_a}{L_b}, \frac{R_a}{R_b}\right), \frac{c_a}{c_b}, \max\left(\frac{L_a}{L_b}, \frac{L_a}{R_b}, \frac{R_a}{L_b}, \frac{R_a}{R_b}\right) \right\rangle
\end{aligned} \tag{2.48}
$$

Having defined arithmetic, another subject we need to address is the comparison of two fuzzy numbers. Since the Lyapunov condition for system stability is represented by an inequality, if we want to be able to determine whether a system is stable, we have to define *ordering* of fuzzy numbers. In other words, we need to introduce some sort of metrics into the set of fuzzy numbers.

Due to their nature, it is clear that the order of fuzzy numbers can be ascertained in various ways. While relations "greater than" and "less than" exclude each other for crisp numbers, these two relations may concur for fuzzy numbers, depending on the ordering function. Generally, we discern two classes of ordering methods. Methods in the first class are based on an ordering relation proposed in Reference 53.

Definition 2.18 (The ordering of fuzzy numbers) Let $\tilde{a}$ and $\tilde{b}$ be two fuzzy numbers and let $\tilde{\succ}$ denote the ordering function *greater than or equal to*. Then $\tilde{a} \mathrel{\tilde{\succ}} \tilde{b}$ if and only if $a_\alpha \geq b_\alpha, \forall\alpha \in (0, 1]; a_\alpha \geq b_\alpha$ if and only if $\overline{a}_\alpha \geq \overline{b}_\alpha$ and $a_\alpha \geq b_\alpha$.

Unfortunately, the above definition may be inconsistent, that is, for two overlapping fuzzy numbers we may get different orderings for different values of α. Nevertheless, in case of fuzzy numbers that fulfil conditions (2.27) and (2.28), graphically depicted in Figure 2.15, Definition 2.17 gives exclusive ordering.

The ordering methods that belong to the second class can overcome the inconsistency problem. They are based on a crisp representation of fuzzy numbers [54]. First, the fuzzy numbers' counterparts (indices) in the set of real numbers are determined and the obtained values are compared. Fuzzy numbers are usually represented by an area or COG (COA) in these methods. Since a weighted area calculation has many different procedures, the ordering of the set of fuzzy numbers

TABLE 2.9
Control Rules

	x_1				
x_2	**NM**	**NS**	**Z**	**PS**	**PM**
NM	PL	PL	PM	PS	Z
NS	PL	PM	PS	Z	NS
Z	PM	PS	Z	NS	NM
PS	PS	Z	NS	NM	NL
PM	Z	NS	NM	NL	NL

From Bogdan, S., Kovačić, Z., and Punčec, M., *IEEE 4th Int. Conf. Intell. Syst. Design Appl.*, 271–276, 2004. With permission.

attained by one method may be different from results obtained by a method that calculates the area by another principle.

In the text that follows we use ordering according to Definition 2.17 since it does not require the calculation of a fuzzy number index. Furthermore, fuzzy sets used in the rest of the book fulfill conditions (2.27) and (2.28) and provide consistent ordering.

Let us now return to fuzzy controller stability described in Reference 50. Instead of using only signs of state variables, input domains are partitioned in five linearly distributed fuzzy sets: NM, NS, Z, PS, and PM. The rules shown in Table 2.9 are obtained by including these linguistic values into the Lyapunov stability condition (2.43) and by using fuzzy arithmetic (2.45).

In case $\tilde{x}_1 = \text{NS}$ and $\tilde{x}_2 = \text{NM}$ then $\tilde{x}_2 \cdot (\tilde{x}_1 + \tilde{u}_{\text{FC}}) = \text{NM} \cdot (\text{NS} + \tilde{u}_{\text{FC}}) \prec \tilde{0}$ (a set $\tilde{0}$ is a fuzzy singleton having 0 as its only element). It is clear that the fulfilment of this inequality, that is, stability, depends on domains of the fuzzy numbers in question. Since $\underline{0} = \overline{0} = 0$, that is, both infimum and supremum, of fuzzy singleton $\tilde{0}$ are equal to 0, the domain of $\tilde{x}_2 \cdot (\tilde{x}_1 + \tilde{u}_{\text{FC}})$ should be $(-\infty, 0)$. In Reference 50 this fact is stated in the form of a theorem which states that a fuzzy control system is asymptotically stable if domain of $\tilde{\dot{V}}$ is $(-\infty, 0)$, where $\tilde{\dot{V}}$ is a linguistic value of the Lyapunov function derivative.

It should be noted that the theorem expresses only a sufficient condition for stability which can be easily checked by consulting the rules in Table 2.9. If inputs and output are described by linearly distributed triangular fuzzy numbers, then, for example, a fuzzy Lyapunov criterion in case $\tilde{x}_2 \cdot (\tilde{x}_1 + \tilde{u}_{\text{FC}}) = \text{PS} \cdot (\text{NS} + \text{Z})$ is not satisfied. However, the proposed controller in Table 2.9 is stable. This situation is caused by the fact that fuzzy arithmetic does not utilize all available information, meaning that the imprecision of the obtained results is greater than or equal to the imprecision of the fuzzy numbers used. In standard fuzzy arithmetic, for example, $\tilde{a} - \tilde{a} \neq 0$ or $\tilde{a}/\tilde{a} \neq 1$, which contradicts our intuition (new approaches to fuzzy arithmetic try to resolve this issue by redefining basic fuzzy arithmetic operations,

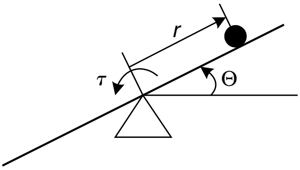


FIGURE 2.28 The ball and beam system. (From Bogdan, S., Kovačić, Z., and Punčec, M., *IEEE 4th Int. Conf. Intell. Syst. Design Appl.*, 271–276, 2004. With permission.)

but that subject requires more space than we have, so an interested reader can advise [55].

In order to overcome problems caused by imprecision, the output of the controller in Table 2.9 may be represented by singletons instead of triangular fuzzy numbers. In that case, our example, $\tilde{x}_2 \cdot (\tilde{x}_1 + \tilde{u}_{\mathrm{FC}}) = \mathrm{PS} \cdot (\mathrm{NS} + \mathrm{Z})$ becomes $\tilde{x}_2 \cdot (\tilde{x}_1 + \tilde{u}_{\mathrm{FC}}) = \mathrm{PS} \cdot (\mathrm{NS} + \tilde{0})$, which is less than $\tilde{0}$ and the Lyapunov stability condition is satisfied.

Fuzzy Lyapunov stability, based on fuzzy numbers and fuzzy arithmetic presented herein, is still being researched and there are many unresolved issues. However, due to its simplicity this approach may be exploited as the first elementary step in fuzzy controller design and fuzzy controller stability analysis, especially when only rudimentary information regarding the controlled process is available. We close this section with an example of fuzzy controller design based on the described method.

Example 2.9 Fuzzy controller stability — the fuzzy arithmetic approach.

The system we have chosen to demonstrate fuzzy arithmetic in fuzzy controller stability analysis is the well-known ball and beam control problem. The system, shown in Figure 2.28, consists of a ball that is free to roll on a beam.

The system is challenging from the control point of view as it is unstable and highly nonlinear. Our goal is to obtain a fuzzy controller that will stabilize the system around a set point r_{ref}. There are many solutions to the problem, from standard PID algorithms to neural networks [56,57]. Here we will solve the problem assuming that only basic knowledge about the system is available in the form of linguistic statements.

By using the Lagrange equation we may obtain a mathematical description of the system. Although the mathematical model is not used either in stability analysis or in controller design, it is given here:

$$\left(\frac{J_{\mathrm{B}}}{R^2} + m\right)\ddot{r} = mr\dot{\theta}^2 - mg\sin\theta$$

$$(mr^2 + J_{\mathrm{b}} + J_{\mathrm{B}})\ddot{\theta} = \tau - 2mr\dot{r}\dot{\theta} - mgr\cos\theta$$

where m is ball mass, R is ball radius, J_{b} is ball moment of inertia, J_{B} is beam moment of inertia around the center, g is the gravitational constant, τ is torque applied to the beam center, r is ball position, and θ is beam angle.

We know the following facts about the system:

- The range of beam angle θ is $\pm\pi/4$.
- The ball should be held within ± 0.1 m from the center of the beam.
- The ball position and the beam angle are measured.

Although we are assuming that the exact physical law of motion is unknown, we can, because of experience, distinguish that the ball's acceleration increases as the beam angle increases, so $\ddot{r} \approx \theta$ (note that according to Figure 2.28 a positive angle causes movement in the negative direction). Also, we know that the angular acceleration of the beam is somehow proportional to the applied torque, $\ddot{\theta} \approx \tau$. Since the ball position and the beam angle are measured we choose r and θ as process variables.

Now, let us define $L - R$ fuzzy numbers that will represent the linguistic values of deviations of process variables from the set points, $e_r = r_{\text{ref}} - r$ and $e_\theta = \theta_{\text{ref}} - \theta$. We define three linguistic values; *negative*, *zero*, and *positive*. The ranges of r and θ are known, thus

$$\tilde{e}_{r\text{N}} = \langle -0.2, -0.1, 0\rangle, \quad \tilde{e}_{r\text{Z}} = \langle -0.1, 0, 0.1\rangle, \quad \tilde{e}_{r\text{P}} = \langle 0, 0.1, 0.2\rangle$$
$$\tilde{e}_{\theta\text{N}} = \langle -\pi/2, -\pi/4, 0\rangle, \quad \tilde{e}_{\theta\text{Z}} = \langle -\pi/4, 0, \pi/4\rangle, \quad \tilde{e}_{\theta\text{P}} = \langle 0, \pi/4, \pi/2\rangle$$

Their derivatives, $\dot{e}_r$ and $\dot{e}_\theta$, are approximated by using the difference between two consecutive measurements (sampling time $T_\text{d} = 10$ msec), $\dot{e}_r \approx e_{dr}(k) = [e_r(k) - e_r(k-1)]/T_\text{d}$ and $\dot{e}_{r\theta} \approx e_{d\theta}(k) = [e_\theta(k) - e_\theta(k-1)]/T_\text{d}$. Since we assume that system dynamics are unknown, one of the ways to determine fuzzy numbers for these two variables is to require that ball velocity and beam angular velocity should remain inside predefined values. We bound $|\dot{r}| \leq 0.03$ m/sec and $|\dot{\theta}| \leq \pi/2$ rad/sec, which gives

$$\tilde{e}_{dr\text{N}} = \langle -1, -0.03, 0\rangle, \quad \tilde{e}_{dr\text{Z}} = \langle -0.03, 0, 0.03\rangle, \quad \tilde{e}_{dr\text{P}} = \langle 0, 0.03, 1\rangle$$
$$\tilde{e}_{d\theta\text{N}} = \langle -\pi, -\pi/2, 0\rangle, \quad \tilde{e}_{d\theta\text{Z}} = \langle -\pi/2, 0, \pi/2\rangle, \quad \tilde{e}_{d\theta\text{P}} = \langle 0, \pi/2, \pi\rangle$$

Although centers of proposed fuzzy numbers correspond with predefined boundaries it should be noted that we leave wide margins since actual values of velocities are unknown. Having defined the deviations of process variables and their linguistic values we may proceed with the fuzzy Lyapunov stability test. The Lyapunov function has the following form:

$$V = \tfrac{1}{2}(e_r^2 + \dot{e}_r^2 + e_\theta^2 + \dot{e}_\theta^2)$$

Its derivative gives (recall that $\ddot{r} \approx \theta$ and $\ddot{\theta} \approx \tau$)

$$\dot{V} = e_r\dot{e}_r + \dot{e}_r\ddot{e}_r + e_\theta\dot{e}_\theta + \dot{e}_\theta\ddot{e}_\theta = e_r\dot{e}_r + \dot{e}_r\theta + e_\theta\dot{e}_\theta - \dot{e}_\theta\tau$$

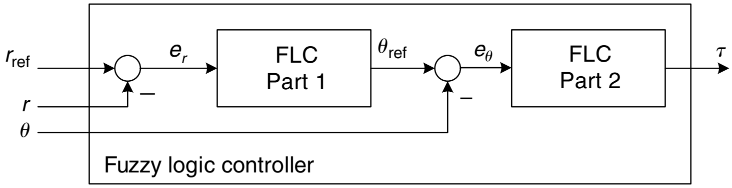


FIGURE 2.29 A cascade fuzzy controller used for ball and beam system stabilization. (From Bogdan, S., Kovačić, Z., and Punčec, M., *IEEE 4th Int. Conf. Intell. Syst. Design Appl.*, 271–276, 2004. With permission.)

For the system to be asymptotically stable we require $\dot{V} < 0$. By using the extension principle, the inclusion of linguistic values of the variables in the form of fuzzy numbers in the above equation will give a fuzzy Lyapunov stability criterion that will eventually define the rules of a fuzzy controller. Since each variable has three linguistic values, there are 81 possible combinations that should be tested. Hence, a final fuzzy controller will have 81 rules. We can take a different approach in order to reduce the number of rules. Let us study each of the two terms in the derivative of the Lyapunov function separately. First we determine stability conditions for $e_r\dot{e}_r + \dot{e}_r\theta < 0$ and then for $e_\theta\dot{e}_\theta - \dot{e}_\theta\tau < 0$. In that way the fuzzy controller is split into two parts; the first part, created by the first term, should generate a set point (commanded beam angle θ_{ref}) for the second part, whose design is based on the second term. The output from the second part of the fuzzy controller is torque τ applied to the beam. Such a fuzzy controller forms a cascade control scheme shown in Figure 2.29 [58]. The advantage of this approach is a significant reduction in the number of rules. While a standard controller contains 81 rules, a cascade fuzzy controller has only $9 + 9 = 18$ rules.

The insertion of fuzzy numbers that represent linguistic values of variables involved in the first term $e_r\dot{e}_r + \dot{e}_r\theta$, results in the following:

$$\begin{aligned}
&\tilde{e}_{dr\text{N}}(\tilde{e}_{r\text{N}} + \tilde{\theta}_{\text{ref}}) \prec \tilde{0}, \quad \tilde{e}_{dr\text{N}}(\tilde{e}_{r\text{Z}} + \tilde{\theta}_{\text{ref}}) \prec \tilde{0}, \quad \tilde{e}_{dr\text{N}}(\tilde{e}_{r\text{P}} + \tilde{\theta}_{\text{ref}}) \prec \tilde{0},\\
&\tilde{e}_{dr\text{Z}}(\tilde{e}_{r\text{N}} + \tilde{\theta}_{\text{ref}}) \prec \tilde{0}, \quad \tilde{e}_{dr\text{Z}}(\tilde{e}_{r\text{Z}} + \tilde{\theta}_{\text{ref}}) \prec \tilde{0}, \quad \tilde{e}_{dr\text{Z}}(\tilde{e}_{r\text{P}} + \tilde{\theta}_{\text{ref}}) \prec \tilde{0},\\
&\tilde{e}_{dr\text{P}}(\tilde{e}_{r\text{N}} + \tilde{\theta}_{\text{ref}}) \prec \tilde{0}, \quad \tilde{e}_{dr\text{P}}(\tilde{e}_{r\text{Z}} + \tilde{\theta}_{\text{ref}}) \prec \tilde{0}, \quad \tilde{e}_{dr\text{P}}(\tilde{e}_{r\text{P}} + \tilde{\theta}_{\text{ref}}) \prec \tilde{0}
\end{aligned}$$

The range of θ_{ref} should be the same as the range of θ. In order to get a smooth control surface, the domain of θ_{ref} is represented by five fuzzy numbers, NL, *negative*, *zero*, *positive*, and PL defined as:

$$\begin{aligned}
&\tilde{\theta}_{\text{refNL}} = \langle -1.2, -0.8, -0.4 \rangle, \quad \tilde{\theta}_{\text{refN}} = \langle -0.8, -0.4, 0 \rangle, \quad \tilde{\theta}_{\text{refZ}} = \langle -0.4, 0, 0.4 \rangle,\\
&\tilde{\theta}_{\text{refP}} = \langle 0, 0.4, 0.8 \rangle, \quad \tilde{\theta}_{\text{refPL}} = \langle 0.4, 0.8, 1.2 \rangle
\end{aligned}$$

We need to determine a linguistic value of beam angle set point θ_{ref} for each of the inequalities so that all of them are fulfilled. In case we choose $\tilde{\theta}_{\text{refPL}}$ as the first

TABLE 2.10
Rule Table for the First Part of a Cascade Fuzzy Controller

	e_r		
e_{dr}	**N**	**Z**	**P**
N	PL	P	Z
Z	P	Z	N
P	Z	N	NL

Adapted from Bogdan, S., Kovačić, Z., and Punčec, M., *IEEE 4th Int. Conf. Intell. Syst. Design Appl.*, 271–276, 2004. With permission.

inequality we get

$$\begin{aligned}\tilde{e}_{dr\mathrm{N}}(\tilde{e}_{r\mathrm{N}} + \tilde{\theta}_{\mathrm{refPL}}) &= \langle -1, -0.03, 0\rangle(\langle -0.2, -0.1, 0\rangle + \langle 0.4, 0.8, 1.2\rangle)\\ &= \langle -1, -0.03, 0\rangle\langle 0.2, 0.7, 1.2\rangle\\ &= \langle -1.2, -0.021, 0\rangle \tilde{\prec} \tilde{0}\end{aligned}$$

thus, the inequality is satisfied and the first rule becomes: "IF e_r is *negative* AND e_{dr} is *negative* THEN θ_{ref} is *positive large*." Other rules can be obtained in the same manner. The final fuzzy rule table determined according to the first part of the Lyapunov function derivative is shown in Table 2.10.

Let us now analyze the second part of the Lyapunov function derivative, $e_\theta \dot{e}_\theta - \dot{e}_\theta \tau < 0$. As in the previous case, we attain nine inequalities that have to be fulfilled in order to get stable behavior of the closed loop system

$$\begin{array}{lll}\tilde{e}_{d\theta\mathrm{N}}(\tilde{e}_{\theta\mathrm{N}} - \tilde{\tau}) \prec \tilde{0}, & \tilde{e}_{d\theta\mathrm{N}}(\tilde{e}_{\theta\mathrm{Z}} - \tilde{\tau}) \prec \tilde{0}, & \tilde{e}_{d\theta\mathrm{N}}(\tilde{e}_{\theta\mathrm{P}} - \tilde{\tau}) \prec \tilde{0},\\ \tilde{e}_{d\theta\mathrm{Z}}(\tilde{e}_{\theta\mathrm{N}} - \tilde{\tau}) \prec \tilde{0}, & \tilde{e}_{d\theta\mathrm{Z}}(\tilde{e}_{\theta\mathrm{Z}} - \tilde{\tau}) \prec \tilde{0} & \tilde{e}_{d\theta\mathrm{Z}}(\tilde{e}_{\theta\mathrm{P}} - \tilde{\tau}) \prec \tilde{0},\\ \tilde{e}_{d\theta\mathrm{P}}(\tilde{e}_{\theta\mathrm{N}} - \tilde{\tau}) \prec \tilde{0}, & \tilde{e}_{d\theta\mathrm{P}}(\tilde{e}_{\theta\mathrm{Z}} - \tilde{\tau}) \prec \tilde{0}, & \tilde{e}_{d\theta\mathrm{P}}(\tilde{e}_{\theta\mathrm{P}} - \tilde{\tau}) \prec \tilde{0}\end{array}$$

The first inequality gives

$$\tilde{e}_{d\theta\mathrm{N}}(\tilde{e}_{\theta\mathrm{N}} - \tilde{\tau}) = \langle -\pi, -\pi/2, 0\rangle(\langle -\pi/2, -\pi/4, 0\rangle - \tilde{\tau}) \tilde{\prec} \tilde{0}$$

which yields

$$\tilde{\tau} \tilde{\prec} \langle -\pi/2, -\pi/4, 0\rangle$$

The other inequalities make it clear that the value of applied torque for the first inequality should be the most negative one, that is, we should assign linguistic value NL with $\tilde{\tau}_{\mathrm{NL}} = \langle -3\pi/4, -\pi/2, -\pi/4\rangle$. The corresponding fuzzy rule is

TABLE 2.11
Rule Table for the Second Part of a Cascade Fuzzy Controller

	e_θ		
$e_{d\theta}$	**N**	**Z**	**P**
N	NL	N	Z
Z	N	Z	P
P	Z	P	PL

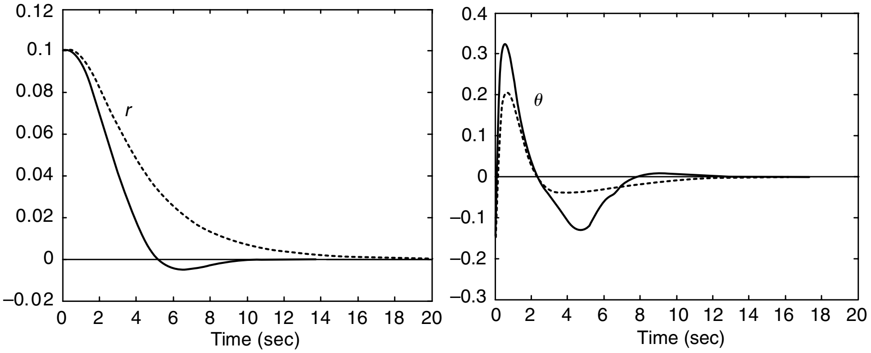


FIGURE 2.30 The response of a ball and beam system controlled with a cascade fuzzy controller; initial conditions $r = 0.1$ m and $\theta = -0.3$ rad. (From Bogdan, S., Kovačić, Z., and Punčec, M., *IEEE 4th Int. Conf. Intell. Syst. Design Appl.*, 271–276, 2004. With permission.)

"IF e_θ is *negative* AND $e_{d\theta}$ is *negative* THEN τ is *negative large*." The obtained fuzzy rule table for the second part of the fuzzy controller is shown in Table 2.11.

The problem with calculated torque values is that they are based on nothing but the elementary knowledge of the system. It is clear that torque $\tilde{\tau}_{\mathrm{NL}} = \langle -3\pi/4, -\pi/2, -\pi/4 \rangle$ may not be enough to move the beam in the right direction if, for example, the ball's mass is significant. Nevertheless, the obtained fuzzy controller is a solid first step in stability analysis and design. Once the basic structure of the controller is known, it is simple to extend the rule table, readjust fuzzy numbers, or tune input and output scaling factors.

The response of autonomous system ($r_{\mathrm{ref}} = 0$) controlled with a cascade fuzzy controller with initial conditions $r = 0.1$ m and $\theta = -0.3$ rad are shown in Figure 2.30 (dotted lines). One may see that the system is stable, but rather slow. Since the determination of fuzzy numbers representing changes in errors was based on assessments without knowledge about actual boundaries, we can readjust these values in order to make system dynamics faster. Division by factor 2 gives the results shown in Figure 2.30 (solid line). The system remains stable with a faster response containing a slight overshoot.

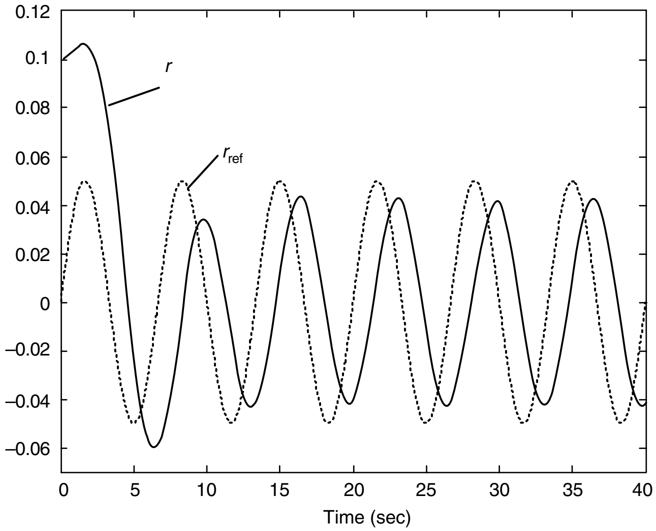


FIGURE 2.31 Tracking performance of a ball and beam system controlled with a cascade fuzzy controller; initial conditions $r = 0.1$ m and $\theta = -0.3$ rad, $r_{\text{ref}} = 0.05\sin(0.94t)$. (From Bogdan, S., Kovačić, Z., and Punčec, M., *IEEE 4th Int. Conf. Intell. Syst. Design Appl.*, 271–276, 2004. With permission.)

Tracking performance of the system is tested with signal $r_{\text{ref}} = 0.05\ \sin(0.94t)$ (Figure 2.31), indicating very good control quality and stable system behavior.

References

1. Gupta, M.M. and Qi, J., "Theory of T-norms and fuzzy inference methods," *Fuzzy Sets and Systems*, 40, 431–450, 1991.
2. Gupta, M.M. and Qi, J., "Design of fuzzy logic controllers based on generalized T-operators," *Fuzzy Sets and Systems*, 40, 473–489, 1991.
3. Pedrycz, W., "Processing in relational structures: fuzzy relational equations," *Fuzzy Sets and Systems*, 40, 77–106, 1991.
4. Filev, D.P. and Yager, R.R., "On the analysis of fuzzy controllers," *Fuzzy Sets and Systems*, 68, 39–66, 1994.
5. Braae, M. and Rutherford, D.A., "Theoretical and linguistic aspects of the fuzzy logic controller," *Automatica*, 15, 553–577, 1979.
6. Xiangchu, T. and Chengyuan, T., "A new approach to fuzzy control," in M. Gupta and T. YamaKawa (eds), *Fuzzy Logic in Knowledge Based Systems, Decision and Control*, 307–315, 1988.
7. Hohle, U. and Stout, L.N., "Foundations of fuzzy sets," *Fuzzy Sets and Systems*, 40, 257–296, 1991.
8. Bouslama, F. and Ichikawa, A., "Fuzzy control rules and their natural control laws," *Fuzzy Sets and Systems*, 48, 65–86, 1992.
9. Kickert, W.J.M. and van Nauta Lemke, H.R., "Application of a fuzzy controller in a warm water plant," *Automatica*, 12, 301–308, 1976.

10. Meier, R., Nieuwland, J., Zbinden, A.M., and Hacisalihzade, S.S., "Fuzzy logic control of blood preasure during anesthesia," *IEEE Control Systems Magazine*, December, 12–17, 1992.
11. Heckenthaler, T. and Engell, S., "Approximately time-optimal fuzzy control of a two-tank system," *IEEE Control Systems Magazine*, June, 24–30, 1994.
12. Backley, J.J., "Sugeno type controllers are universal controllers," *Fuzzy Sets and Systems*, 53, 299–303, 1992.
13. Mizumoto, M., "Fuzzy controls under various fuzzy reasoning methods," *Information Sciences*, 45, 129–151, 1988.
14. Hellendoorn, H., "Closure properties of the compositional rule of inference," *Fuzzy Sets and Systems*, 35, 163–183, 1990.
15. Runkler, T.A., "Selection of appropriate defuzzification methods using application specific properties," *IEEE Transactions on Fuzzy Systems*, 5, 72–79, 1997.
16. Kickert, W.J.M. and Mamdani, E.H., "Analysis of a fuzzy logic controller," *Fuzzy Sets and Systems*, 1, 29–44, 1978.
17. Matia, F., Jimenez, A., Galan, R., and Sanz, R., "Fuzzy controllers: lifting the linear–nonlinear frontier," *Fuzzy Sets and Systems*, 52, 113–128, 1992.
18. Backley, J.J., "Theory of the fuzzy controller: an introduction," *Fuzzy Sets and Systems,* 51, 249–258, 1992.
19. Hajjaji, A.E. and Rachid, A., "Explicit formulas for fuzzy controller," *Fuzzy Sets and Systems*, 62, 135–141, 1994.
20. Kosko, B., *Fuzzy Engineering*, Prentice Hall, New Jersey, 1996.
21. Driankov, D., Hellendoorn, H., and Reinfrank, M., *An Introduction to Fuzzy Control*, Springer-Verlag, Berlin, 1993.
22. Wang, L.X., *Adaptive Fuzzy Systems and Control: Design and Stability Analysis*, Prentice Hall, New Jersey, 1994.
23. Zhao, Z.Y., Tomizuka, M., and Sagara, S., "A fuzzy tuner for fuzzy logic controllers," in *Proceedings of the American Control Conference*, Chicago, pp. 2268–2272, 1992.
24. Ling, C. and Edgar, T.F., "A new fuzzy gain scheduling algorithm for process control," in *Proceedings of the American Control Conference*, pp. 2284–2290, 1992.
25. Kovačić, Z. and Bogdan, S., "Model reference adaptive fuzzy control of high-order systems," *Engineering Applications of Artificial Intelligence*, 7, 501–511, 1994.
26. Ollero, A. and Garcia-Cerezo, A.J., "Direct digital control, auto-tuning and supervision using fuzzy logic," *Fuzzy Sets and Systems*, 30, 135–153, 1989.
27. Smith, S.M. and Comer, D.J., "Automated calibration of a fuzzy logic controller using a cell state space algorithm," *IEEE Control Systems Magazine*, August, 18–28, 1991.
28. Tong, R.M., "A control engineering review of fuzzy systems," *Automatica*, 13, 559–569, 1977.
29. Bouslama, F. and Ichikawa, A., "Application of limit fuzzy controllers to stability analysis," *Fuzzy Sets and Systems*, 49, 103–120, 1992.
30. Mon, D.L. and Cheng, C.H., "Fuzzy systems reliability analysis for components with different membership functions," *Fuzzy Sets and Systems*, 64, 145–157, 1994.
31. Dombi, J., "Membership function as an evaluation," *Fuzzy Sets and Systems*, 35, 1–21, 1990.
32. Chang, T.C., Hasegawa, K., and Ibbs, C.W., "The effects of membership function on fuzzy reasoning," *Fuzzy Sets and Systems*, 41, 169–186, 1991.

33. Turksen, I.B., “Measurement of membership functions and their acquisition,” *Fuzzy Sets and Systems*, 40, 5–38, 1991.
34. Mamdani, E.H., “Application of fuzzy algorithms for control of simple dynamic plant,” *Proceedings of the IEE*, 121, 1585–1588, 1974.
35. Novaković, B., Kasać, J., Majetić, D., and Brezak, D., “A new analytic adaptive fuzzy robot control,” *Transactions of FAMENA*, XXVI, 21–34, 2002.
36. Pedrycz, W., “Why triangular membership functions,” *Fuzzy Sets and Systems*, 64, 21–30, 1994.
37. Kandel, Y.L. and Zhang, Y.-Q., “Stability analysis of fuzzy control systems,” *Fuzzy Sets and Systems*, 105, 33–48, 1999.
38. Bandemer, H. and Hartmann, S., “A fuzzy approach to stability of fuzzy controllers,” *Fuzzy Sets and Systems*, 96, 161–172, 1998.
39. Chen, C.-S., “Design of stable fuzzy control systems using Lyapunov’s method in fuzzy hypercubes,” *Fuzzy Sets and Systems*, 139, 95–110, 2003.
40. Ray, K.S. and Majumder, D., “Application of circle criteria for stability analysis of linear SISO and MIMO systems associated with fuzzy logic controller,” *IEEE Transactions on Systems, Man and Cybernetics*, 14, 345–349, 1984.
41. Hwang, G.C. and Lin, S.C., “A stability approach to fuzzy control design for nonlinear systems,” *Fuzzy Sets and Systems*, 48, 279–287, 1992.
42. Wu, J.C. and Liu, T.S., “Fuzzy control stabilization with applications to motorcycle control,” *IEEE Transactions on Systems, Man and Cybernetics*, 26, 836–847, 1996.
43. Thathachar, M.A.L. and Viswanath, P., “On the stability of fuzzy systems,” *IEEE Transactions on Fuzzy Systems*, 5, 145–151, 1997.
44. Farinwata, S.S. and Vachtsevanos, G., “Robust stability of fuzzy logic control systems,” in *Proceedings of the American Control Conference*, Seattle, pp. 2267–2271, 1995.
45. Tanaka, K. and Sugeno, M., “Stability analysis and design of fuzzy control systems,” *Fuzzy Sets and Systems*, 45, 135–156, 1992.
46. Luoh, L., “New stability analysis of T–S fuzzy system with robust approach,” *Mathematics and Computers in Simulation*, 59, 335–340, 2002.
47. Lyapunov, A.M., *General Problem of the Stability of Motion*, Taylor & Francis Books Ltd, London, 1992.
48. Margaliot, M. and Langholz, G., “Fuzzy Lyapunov based approach to the design of fuzzy controllers,” *Fuzzy Sets and Systems*, 106, 49–59, 1999.
49. Zadeh, L.A., “From computing with numbers to computing with words — from manipulation of measurements to manipulation of perceptions,” *International Journal of Applied Mathematics and Computer Science*, 12, 307–324, 2002.
50. Zhou, C., “Fuzzy-arithmetic-based Lyapunov synthesis in the design of stable fuzzy controllers: a computing-with-words approach,” *International Journal of Applied Mathematics and Computer Science*, 12, 411–421, 2002.
51. Dubois, D. and Prade, H., “Fuzzy numbers: an overview,” in J.C. Bezdek (ed.), *Analysis of Fuzzy Information*, Vol. 2, CRC-Press, Boca Raton, FL, pp. 3–39, 1988.
52. Oussalah, M. and De Schutter, J., “Approximated fuzzy LR computation,” *Information Sciences*, 153, 155–175, 2003.
53. Klir, G.J. and Yuan, B., *Fuzzy Sets and Fuzzy Logic: Theory and Applications*, Prentice Hall, Upper Saddle River, NJ, 1995.
54. Yager, R.R., “A procedure for ordering fuzzy subsets of the unit interval,” *Information Sciences*, 24, 143–161, 1981.

55. Klir, G.J., “Fuzzy arithmetic with requisite constraints,” *Fuzzy Sets and Systems*, 91, 165–175, 1997.
56. Eaton, P.H., Prokhorov, D.V., and Wunsch, II D.C., “Neurocontroller alternatives for ‘fuzzy’ ball-and-beam systems with nonuniform nonlinear friction,” *IEEE Transactions on Neural Networks*, 11, 423–435, 2000.
57. Hauser, J., Sastry, S., and Kokotovic, P., “Nonlinear control via approximate input–output linearization: the ball and beam example,” *IEEE Transactions on Automatic Control*, 37, 392–398, 1992.
58. Guanghui, W., Yantao, T., Wei, H., and Huimin, J., “Stabilization and equilibrium control of super articulated ball and beam system,” in *Proceedings of the Third World Congress on Intelligent Control and Automation*, Hefei, China, June 28–July 2, 2000.
59. Bogdan S., Kovačić Z.; “A Cascade Fuzzy Controller Design Based on Fuzzy Lyapunov Stability,” In *Proceedings of the 4th IEEE International Conference on Intelligent Systems Design and Applications ISDA’04*, Budapest, 271–276, 2004.

3 Initial Setting of Fuzzy Controllers

The initial structure of a fuzzy controller depends on the specifics of the controlled process, desired control quality, and the information obtained from an expert. Heuristic design and tuning of fuzzy controllers can be a rather demanding and time consuming job even when using specialized development tools. Very often, engineers who want to apply fuzzy controllers in industry must first go through a long negotiation process before their customer accepts a new controller. This is partly because of insufficient education of field personnel and partly because of a general suspicion of the new controller's reliability.

This can be overcome by fuzzy controller design methods, which are closely related to the synthesis of well-known control concepts and existing controllers. For example, in Reference 1 a gradient descent method is proposed for tuning a Takagi–Sugeno set of fuzzy rules while in Reference 2 the same method is applied to a fuzzy rule base with output singletons. The tuning of fuzzy controller parameters can be based on the Hooke–Jeeves pattern search algorithm, as explained in Reference 3. The implementation of the proposed algorithm shows that this method is able to tune a fuzzy controller with 9 and 25 rules in order to catch up to the behavior of a proportional-derivative (PD) controller. A fuzzy version of a well-known neural network model, a Kohonen's self-organizing map, is introduced in Reference 4, where Kohonen's learning laws are used for tuning the centers of fuzzy sets and for initialization of fuzzy rules. Dead-beat control philosophy has been applied in Reference 5 in order to implement a fuzzy logic gain scheduling algorithm for predicting the next proportional-integral-derivative (PID) controller output value. The concept of model predictive control may be used for setting fuzzy PID controllers, which control processes with delays and chaotic behavior [6]. Proportional-integral (PI) predictive fuzzy controllers may be tuned according to a so called symmetrical optimum in order to guarantee the desired domain for the "phase margin" of fuzzy controlled astatic control processes [7].

Although very successful in practice, such fuzzy controller tuning methods are not simple enough in cases when the tuning of fuzzy controllers must be done by less well-educated and less experienced field engineers.

We describe three approaches to initial fuzzy controller setting, which result in easy-to-implement algorithms: design of P-I-D-like fuzzy control algorithms, model reference-based design, and design using phase plane trajectories. These methods can be used for automated initial setting of fuzzy controllers used in nonlinear, inherently stable, time-varying single-input single-output (SISO) high-order systems, which can be linearized in a selected operating point. Such systems are often found in the industrial processes (e.g., control of temperature, pressure, flow, level, angular speed, and position). We also describe the implementation of three initial setting methods obtained from the above approaches and give experimental results of controlling a laboratory process.

3.1 Fuzzy Emulation of P-I-D Control Algorithms

No matter how complicated the control of a plant may seem, the majority of control loops in industrial control systems utilize standard P, PI, PD, or PID control algorithms (here denoted as P-I-D) with fixed parameter values set during the commissioning. The synthesis of P-I-D controller parameters based on well-known design methods normally requires a mathematical model, which can precisely describe the dynamical behavior of a control object. Values of P-I-D controller parameters obtained in such a way describe a linear control law adequate for a selected operating point. If such a controller is applied to a nonlinear control system, the performance of the system will vary depending on the variations of control object parameters. Also, the usage of a linear control law will cause different responses of a nonlinear system for the same magnitude of positive and negative reference input changes.

Different design strategies have been developed with the purpose to overcome the disadvantages of linear P-I-D controllers. Such strategies transform a linear P-I-D controller into P-I-D-like structures of fuzzy controllers such as PI, PD, PI+D, PD+I, and PI+PD [8–12]. An informative review of various fuzzy P-I-D-like controllers can be found in Reference 13. When designing a fuzzy controller by emulating of a linear P-I-D controller, we assume that the fuzzy controller should inherit the linear character of its model. In order to evaluate the quality of such a transformation, different measures of achieved linearity have been introduced [13–15]. For example, in Reference 13 it has been shown that nonoverlapping of adjacent output fuzzy sets generally produces higher nonlinearity in fuzzy P-I-D controller than in the overlapping case.

In terms of the influence that different fuzzy reasoning methods (fuzzy implications) have on the achieved linearity of PID-like fuzzy controllers, theoretical results show that the vast majority of fuzzy PI and PID controllers are actually nonlinear PI (PID) controllers [16–18]. In Reference 17 it has been mathematically proven that Takagi–Sugeno type of PI (PD) controllers are nonlinear PI (PID) controllers with P-gain, I-gain, and D-gain changing with the output of the controlled system, providing that they have at least three trapezoidal or triangular input fuzzy sets for each input variable, fuzzy rules with a singleton in the consequent part, Zadeh's AND operator and the centroid defuzzifier. By analyzing and comparing different fuzzy reasoning methods used for the implementation of fuzzy PI controllers, it has been found in Reference 18 that fuzzy PI controller gains do not change if product T-norm is used to assess the antecedent parts of fuzzy rules, and if Zadeh's AND operator is used in the process of fuzzy implication. Moreover, only fuzzy implications that use Mamdani and product T-norms in combination with a Zadeh's AND (i.e., min) operator give a sensible control effect (other methods either generate an incorrect sign of the fuzzy rule contribution or drastically change PID controller gains with the changes of the system output). Another important result presented in Reference 18 is that even in controller configurations acceptable from the control point of view, nonlinear gain increases as fuzzy controller inputs take larger values, so the control effort produced by a fuzzy controller becomes stronger than that of the corresponding linear PI controller.

Although the making a perfect fuzzy copy of a linear P-I-D controller could be an interesting design goal, it is more important to use a linear P-I-D controller as a starting point for the initial setting of a fuzzy controller, because its prime role is not to mimic the original, but to use all of the original's intrinsic nonlinear control potential. This can be achieved through various adaptive and self-organizing (self-learning) design concepts. When a more general solution is wanted, then phase space [15] and phase plane are utilized [19]. So in Reference 19 a minimal number (only 2) of fuzzy sets has been used to describe the current state vector $[e(k), \Delta e(k)]$ in its polar coordinates, its magnitude and its argument. In order to increase the performance of such a PID-like fuzzy controller an auxiliary fuzzy controller has been used.

When it comes to the stability of fuzzy PID controlled systems, bounded input–bounded output (BIBO) stability is mainly assessed using the well-known small gain theorem [17,18,20].

3.1.1 Fuzzy Emulation of a PID Controller

A PID controller has the following form in continuous time domain:

$$u(t) = K_{\mathrm{P}}\left[e(t) + \frac{1}{T_{\mathrm{I}}}\int_0^t e(t)\mathrm{d}t + T_{\mathrm{D}}\frac{\mathrm{d}e(t)}{\mathrm{d}t}\right] = K_{\mathrm{P}}e(t) + K_{\mathrm{I}}\int_0^t e(t)\mathrm{d}t + K_{\mathrm{D}}\frac{\mathrm{d}e(t)}{\mathrm{d}t} \tag{3.1}$$

where K_{P}, K_{I}, and K_{D} are constant proportional, integral, and derivative controller gains, respectively.

Discretization of Equation (3.1) by substituting the integral with the sum of rectangles of the width T_{d} and height $e(iT_{\mathrm{d}})$, $i = 0, 1, 2, \ldots$, where T_{d} is a sampling interval, yields a recursive equation of a discrete linear PID controller:

$$\begin{aligned} u(k) &= K_{\mathrm{P}}e(k) + K_{\mathrm{P}}\frac{T_{\mathrm{d}}}{T_{\mathrm{I}}}\sum_{i=0}^{k} e(i) + K_{\mathrm{P}}\frac{T_{\mathrm{D}}}{T_{\mathrm{d}}}\left[e(k) - e(k-1)\right] \\ &= K_{\mathrm{Pd}}e(k) + K_{\mathrm{Id}}\sum_{i=0}^{k} e(i) + K_{\mathrm{Dd}}\Delta e(k) \end{aligned} \tag{3.2}$$

where $K_{\mathrm{Pd}} = K_{\mathrm{P}}$, $K_{\mathrm{Id}} = K_{\mathrm{P}}T_{\mathrm{d}}/T_{\mathrm{I}}$, and $K_{\mathrm{Dd}} = K_{\mathrm{P}}T_{\mathrm{D}}/T_{\mathrm{d}}$ are corresponding constant proportional, integral, and derivative gains.

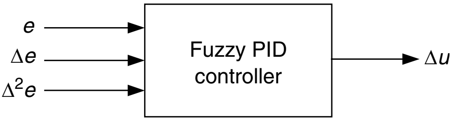


FIGURE 3.1 A fuzzy PID controller — variant A.

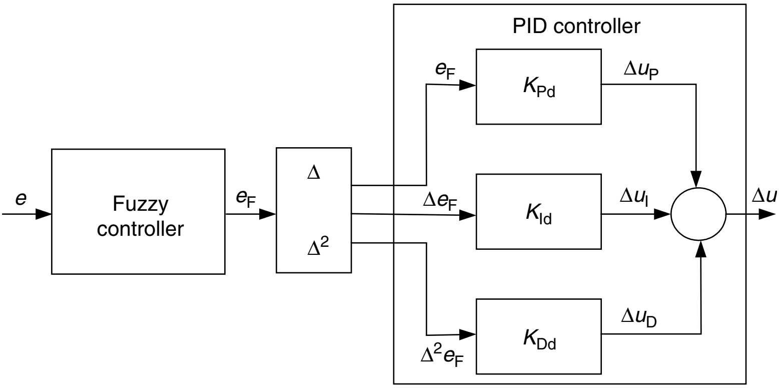


FIGURE 3.2 A fuzzy PID controller — variant B.

In Chapter 2, we have shown that fuzzy controllers are intrinsically nonlinear, so some steps must be taken to make their structure as close to a linear one as possible. A discrete form of a PID controller (3.2) is not convenient for implementation because it contains the sum of all previous control error values. The better solution is to define the difference between two consecutive values of controller output:

$$\begin{aligned}\Delta u(k) &= u(k) - u(k-1) = K_{\mathrm{Id}} e(k) + K_{\mathrm{Pd}} \Delta e(k) + K_{\mathrm{Dd}}[\Delta e(k) - \Delta e(k-1)] \\ &= K_{\mathrm{Id}} e(k) + K_{\mathrm{Pd}} \Delta e(k) + K_{\mathrm{Dd}} \Delta^2 e(k) = \Delta u_{\mathrm{I}}(k) + \Delta u_{\mathrm{P}}(k) + \Delta u_{\mathrm{D}}(k)\end{aligned} \tag{3.3}$$

where $\Delta^2 e(k) = \Delta e(k) - \Delta e(k-1)$.

The form of (3.3) suggests that three possible variants of a fuzzy PID controller could be implemented:

- *Variant A* — a fuzzy PID controller having three inputs e, Δe, and $\Delta^2 e$ and one output Δu, as shown in Figure 3.1.
- *Variant B* — a fuzzy PID controller composed of a linear PID controller and a SISO fuzzy controller with $e(k)$ and $e_{\mathrm{F}}(k)$ as its input and output, as shown in Figure 3.2.
- *Variant C* — a fuzzy PID controller composed of fuzzy P + fuzzy I + fuzzy D controllers having e and Δu_{P}, Δe and Δu_{I}, $\Delta^2 e$ and Δu_{D}, as respective inputs and outputs (Figure 3.3).

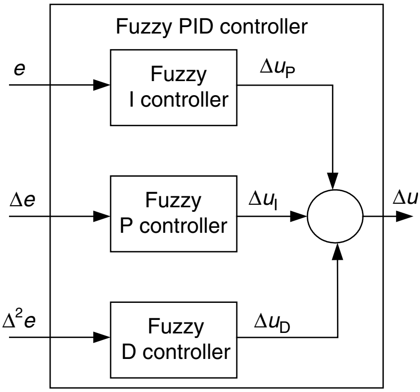


FIGURE 3.3 A fuzzy PID controller — variant C.

For all three variants, the total output of the fuzzy PID controller is

$$u(k) = u(k-1) + \Delta u(k) \tag{3.4}$$

Suppose that we have five fuzzy sets defined for each fuzzy PID controller input. Then variant A will have $5 \times 5 \times 5 = 125$ fuzzy rules, variant B will have only 5, and variant C will have $3 \times 5 = 15$ fuzzy rules. Let us now examine in greater detail what we get by choosing each of the variants as the platform for fuzzy PID controller design.

3.1.1.1 Fuzzy Emulation of a PID Controller — Variant A

In order to emulate a discrete linear PID algorithm described by Equations (3.3) and (3.4) according to the concept shown in Figure 3.1, a fuzzy controller will have $e(k)$, $\Delta e(k)$, and $\Delta^2 e(k)$ as inputs and $\Delta u(k)$ as an output. By knowing minimal and maximal values of these variables, we can determine their universes of discourse and define shapes and distributions of related fuzzy sets. Now, we need to decide on the form and the distribution of fuzzy sets. Let us choose triangular linearly distributed input membership functions, where only two membership functions are overlapping at the intersection point $\mu = 0.5$. Then, after the application of the *product* implication, defuzzification according to the center of gravity method converts into a very simple form (2.24). By using product T-norm for assessing the antecedent parts of fuzzy control rules, and by having singletons in the consequent parts of fuzzy control rules, both Zadeh AND operator and product operator will provide the same value of the rule contribution (see Section 2.3.1).

In Reference 15 it is theoretically proven that the control function of a SISO fuzzy controller with linearly distributed fuzzy partition $E = \{\mathrm{TE}_i\}, i = 1, 2, \ldots, l$, will be smooth. Also, the control function of a DISO fuzzy controller with linearly distributed fuzzy partitions $E = \{\mathrm{TE}_i\}$, $\mathrm{DE} = \{\mathrm{TDE}_j\}$, $i, j = 1, 2, \ldots, l$, will be smooth on every peak point $\Delta u_{ij} = \Delta u(c_i^e, c_j^{\Delta e})$ of a control surface, where c_i^e and

$c_j^{\Delta e}$ are the centers of respective input fuzzy sets TE_i and TDE_j. The connections between the nearest peak points (there are up to eight such points) will also be smooth. By simple extension of the above theoretical results to a three input-single output fuzzy controller with linearly distributed fuzzy partitions $E = \{\mathrm{TE}_i\}$, $\mathrm{DE} = \{\mathrm{TDE}_j\}$, $\mathrm{DDE} = \{\mathrm{TDDE}_k\}$, $i, j, k = 1, 2, \ldots, l$, the corresponding control function will be smooth on every peak point $\Delta u_{ijk} = \Delta u(c_i^e, c_j^{\Delta e}, c_k^{\Delta\Delta e})$ of the control space, where c_i^e, $c_j^{\Delta e}$, and $c_k^{\Delta\Delta e}$ are the centers of respective input fuzzy sets TE_i, TDE_j, and TDDE_k. Every peak point will be surrounded by maximally 26 other peak points (eight in the same layer plus nine in the layers above and below), which are smoothly connected to each other.

The numbers of input variables and their fuzzy sets define the number of rules. For three input variables with l fuzzy sets, the number of rules is l^3. We shall use singleton sets A_q, $1 \le q \le l^3$, whose values correspond with the above-mentioned peak points Δu_{ijk}, $i, j, k = 1, 2, \ldots, l$, instead of symmetrical triangular output fuzzy sets to make the implementation of a fuzzy PID controller simple.

Let TE_i, TDE_j, and TDDE_k be the ith, the jth, and the kth fuzzy set of $e(k)$, $\Delta e(k)$, and $\Delta^2 e(k)$, respectively, and let A_q be the qth singleton of fuzzy PID controller output $\Delta u(k)$. Then the i, j, kth fuzzy rule has the form

$$\begin{aligned}\mathrm{FR}^{ijk}\colon \text{IF } e(k) \text{ is } \mathrm{TE}_i \text{ AND } \Delta e(k) \text{ is } \mathrm{TDE}_j \text{ AND}\\ \Delta^2 e(k) \text{ is } \mathrm{TDDE}_k \text{ THEN } \Delta u(k) \text{ is } \mathrm{A}_q\end{aligned} \tag{3.5}$$

Defuzzification will be carried out according to the center of gravity method described in Equation (2.22). Since only two nearest input membership functions overlap, maximally one, two, four, or eight fuzzy rules can contribute to crisp controller output value. If controller input values $e_i(k)$, $\Delta e_j(k)$, and $\Delta^2 e_k(k)$ are such that they satisfy $\mu_i^e(e_i) = 1$, $\mu_j^{\Delta e}(\Delta e_j) = 1$, and $\mu_j^{\Delta\Delta e}(\Delta^2 e_k) = 1$ (which means that $e_i(k)$, $\Delta e_j(k)$, and $\Delta^2 e_k(k)$ correspond with centers c_i^e, $c_j^{\Delta e}$, and $c_k^{\Delta\Delta e}$ of the respective ith, the jth, and the kth input fuzzy sets), then regardless of whether T-norm, min, or product is used, fuzzy PID controller output is determined by only one rule and its value is equal to

$$\begin{aligned}\Delta u &= \frac{\min\left[\mu_i^e(c_i^e), \mu_j^{\Delta e}(c_j^{\Delta e}), \mu_k^{\Delta\Delta e}\left(c_k^{\Delta\Delta e}\right)\right]\mathrm{A}_q}{\min\left[\mu_i^e(c_i^e), \mu_j^{\Delta e}(c_j^{\Delta e}), \mu_k^{\Delta\Delta e}\left(c_k^{\Delta\Delta e}\right)\right]} = \frac{\min[1,1,1]\,\mathrm{A}_q}{\min[1,1,1]} = \mathrm{A}_q\\ \Delta u &= \frac{\mu_i^e\left(c_i^e\right)\cdot\mu_j^{\Delta e}\left(c_j^{\Delta e}\right)\cdot\mu_k^{\Delta\Delta e}\left(c_k^{\Delta\Delta e}\right)\cdot\mathrm{A}_q}{\mu_i^e\left(c_i^e\right)\cdot\mu_j^{\Delta e}\left(c_j^{\Delta e}\right)\cdot\mu_k^{\Delta\Delta e}\left(c_k^{\Delta\Delta e}\right)} = \frac{1\cdot1\cdot1\cdot\mathrm{A}_q}{1\cdot1\cdot1} = \mathrm{A}_q\end{aligned} \tag{3.6}$$

By equating (3.3) with (3.6) we obtain

$$\Delta u = \mathrm{A}_q = K_{\mathrm{Id}} c_i^e + K_{\mathrm{Pd}} c_j^{\Delta e} + K_{\mathrm{Dd}} c_k^{\Delta\Delta e} \tag{3.7}$$

Equation (3.7) defines values of all output singletons A_q, $1 \leq q \leq l^3$ by inserting earlier determined values of all input fuzzy set centers c_i^e, $c_j^{\Delta e}$, and $c_k^{\Delta\Delta e}$ for $i, j, k = 1, 2, \ldots, l$.

The result is a simple algebraic equation for calculating singleton values that can be easily implemented in control software. By having calculated these values, which correspond to peak points $\Delta u_{ijk} = \Delta u(c_i^e, c_j^{\Delta e}, c_k^{\Delta\Delta e})$ of the control space, we can expect a smooth fuzzy PID control function.

Having the initial setting algorithm (3.7) for a P-I-D-like fuzzy controller, it is easy to derive simpler forms for P-I-, P-D-, or P-like fuzzy controllers. For example, the algorithm for calculating singleton values for a fuzzy P-I controller is obtained directly from (3.7) for $K_{\mathrm{Dd}} = 0$:

$$\Delta u = u_{\mathrm{FC}} = \mathrm{A}_q = K_{\mathrm{Id}} c_i^e + K_{\mathrm{Pd}} c_j^{\Delta e} \tag{3.8}$$

where $1 \leq q \leq l^2$ and $i, j = 1, 2, \ldots, l$.

Now let us see how linear PID and fuzzy PID controllers of variant A are related. Since graphical representations and explanations in three-dimensional control space are not so practical, we shall explain their basic relations through the example of two-dimensional linear PI and fuzzy PI controllers. We shall then, by deduction, draw conclusions for the three-dimensional case.

Since we are using symmetrical triangular input fuzzy sets (Figure 3.4), we can describe the triangular membership function $\mu_i(x) = \mu_i^x$ with two membership functions: one for the left-hand-side domain $[L_i^x, c_i^x]$ and the other for the right-hand-side domain $[c_i^x, R_i^x]$:

$$\begin{aligned}
\mu_{L_i}^x &= \mu_{L_i}(x) = \frac{x - L_i^x}{c_i^x - L_i^x} = \frac{x - L_i^x}{w_i^x}, \quad x \in \left[L_i^x, c_i^x\right] \\
\mu_{R_i}^x &= \mu_{R_i}(x) = \frac{R_i^x - x}{R_i^x - c_i^x} = \frac{R_i^x - x}{w_i^x}, \quad x \in \left[c_i^x, R_i^x\right]
\end{aligned} \tag{3.9}$$

where the widths of the left-hand-side and the right-hand-side domains are denoted as w_i^x.

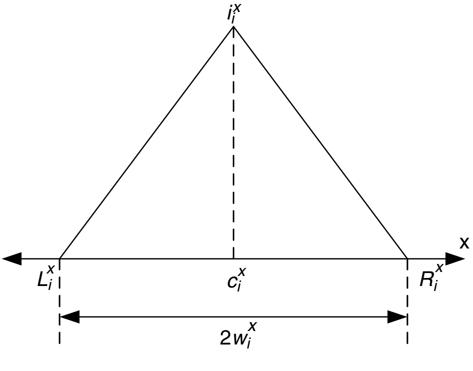


FIGURE 3.4 The parameters of a triangular fuzzy set.

Relations (3.9) indicate that the slopes of the triangular input membership functions are determined by the width of the fuzzy sets.

Equations (3.9) can be rewritten to reflect the degree of membership with regards to relative input value $\bar{x}_i(k) = x(k) - c_i^x$:

$$\mu_{L_i}^x = \mu_{L_i}(x) = \frac{x - c_i^x + c_i^x - L_i^x}{c_i^x - L_i^x} = \frac{\bar{x}_i + w_i^x}{w_i^x} = 1 + \frac{\bar{x}_i}{w_i^x}, \quad x \in [L_i^x, c_i^x]$$

$$\mu_{R_i}^x = \mu_{R_i}(x) = \frac{R_i^x - c_i^x + c_i^x - x}{R_i^x - c_i^x} = \frac{w_i^x - \bar{x}_i}{w_i^x} = 1 - \frac{\bar{x}_i}{w_i^x}, \quad x \in [c_i^x, R_i^x] \tag{3.10}$$

Similarly, the halves of symmetrical triangular input fuzzy sets TE$_i$ and TDE$_j$ are denoted $w_i^e = (R_i^e - L_i^e)/2 = R_i^e - c_i^e = c_i^e - L_i^e$ and $w_j^{\Delta e} = (R_j^{\Delta e} - L_j^{\Delta e})/2 = R_j^{\Delta e} - c_j^{\Delta e} = c_j^{\Delta e} - L_j^{\Delta e}$, respectively. In case of linear distribution of uniform fuzzy sets $\{\text{TE}_i\}$ and $\{\text{TDE}_j\}$, w_i^e and $w_j^{\Delta e}$ become constant parameters w_e and $w_{\Delta e}$.

Let the universe of discourse of $e(k)$ be $E = [e_{\min}, e_{\max}]$, and of $\Delta e(k)$ $\text{DE} = [\Delta e_{\min}, \Delta e_{\max}]$. For the l input fuzzy sets, $w_e = (e_{\max} - e_{\min})/(l-1)$, $w_{\Delta e} = (\Delta e_{\max} - \Delta e_{\min})/(l-1)$. Let every input of the fuzzy controller have seven fuzzy sets, $l = 7$. Then we get a graphical presentation of the phase plane as shown in Figure 3.5. One may see that all points on the control curve $\Delta u(k) = \psi[e(k), \Delta e(k)]$, lying on the control surface above the phase plane, were created by contributions of maximally four output singletons (peak points). When studying the control space of a fuzzy PID controller, we should anticipate that each controller output value $\Delta u(k) = \psi\ [e(k), \Delta e(k), \Delta^2 e(k)]$ will be surrounded by maximally eight such peak points (see Figure 3.6).

Figure 3.5 shows four singletons $A_{i,j} - A_{i+1,j+1}$ that surround the designated controller output value $\Delta u(k)$. Singleton $A_{i,j}$ contributes to $\Delta u(k)$ through fuzzy rule FR^{ij}, $A_{i,j+1}$ through fuzzy rule $\text{FR}^{i(j+1)}$, $A_{i+1,j+1}$ through fuzzy rule $\text{FR}^{(i+1)(j+1)}$, and $A_{i+1,j}$ through fuzzy rule $\text{FR}^{(i+1)j}$. In this segment of the phase plane, contributions of singletons to $\Delta u(k)$ are actually determined by the right-hand sides of μ_i^e and $\mu_j^{\Delta e}$, and the left-hand sides of μ_{i+1}^e and $\mu_{j+1}^{\Delta e}$. In other words, the domain of the phase plane segment is $e(k) \in [L_{i+1}^e, R_i^e]$, $\Delta e(k) \in [L_{j+1}^{\Delta e}, R_j^{\Delta e}]$.

Working with relative fuzzy controller input values $\bar{e}_i(k) = e(k) - c_i^e$, $\Delta\bar{e}_j(k) = \Delta e(k) - c_j^{\Delta e}$ and referring to (3.10) we obtain

$$\mu_{R_i}^e = 1 - \frac{\bar{e}_i}{w_e}, \quad \mu_{R_j}^{\Delta e} = 1 - \frac{\Delta\bar{e}_j}{w_{\Delta e}}, \quad \mu_{L(i+1)}^e = 1 + \frac{\bar{e}_{i+1}}{w_e}, \quad \mu_{L(j+1)}^{\Delta e} = 1 + \frac{\Delta\bar{e}_j}{w_{\Delta e}} \tag{3.11}$$

Since all adjacent triangular fuzzy sets overlap at crossover value $\mu = 0.5$, the following holds:

$$\begin{aligned} \mu_{L(i+1)}^e &= 1 - \mu_{R_i}^e = \frac{\bar{e}_i}{w_e} \\ \mu_{L(j+1)}^{\Delta e} &= 1 - \mu_{R_j}^{\Delta e} = \frac{\Delta\bar{e}_j}{w_{\Delta e}} \end{aligned} \tag{3.12}$$

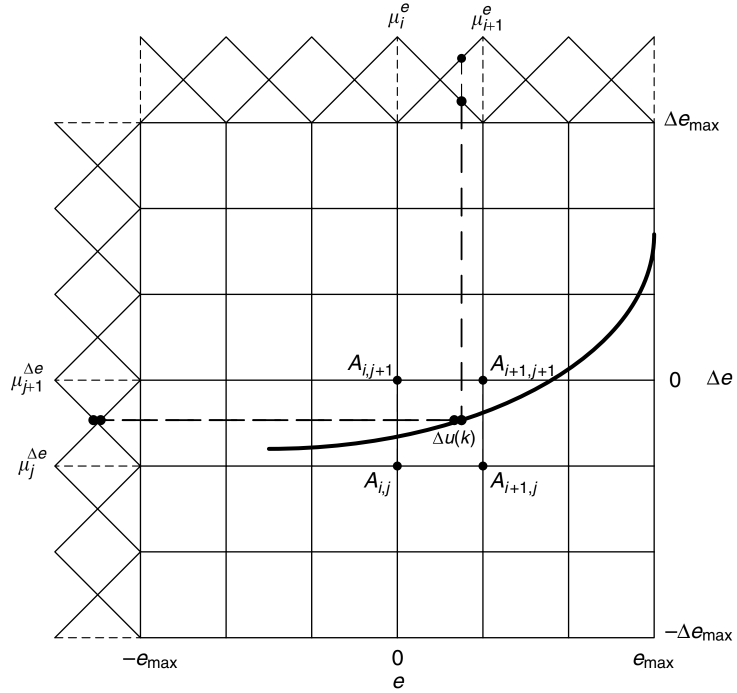


FIGURE 3.5 The phase plane of a fuzzy PI controller.

By using product T-norm for assessing the antecedent parts of the fuzzy control rules, and recalling from (2.24) that $\sum_{j=1}^{r} \mu_j = 1$, $\varphi_j = \mu_j$, and $\Delta u = \sum_{j=1}^{r} A_j \mu_j$, then the crisp output of fuzzy controller $\Delta u(k)$ depends on singletons $A_{i,j} - A_{i+1,j+1}$ in the following way:

$$\begin{aligned}
\Delta u(k) &= \mu_{R_i}^{e} \mu_{R_j}^{\Delta e} A_{i,j} + \mu_{R_i}^{e} \mu_{L_{(j+1)}}^{\Delta e} A_{i,j+1} + \mu_{L_{(i+1)}}^{e} \mu_{L_{(j+1)}}^{\Delta e} A_{i+1,j+1} \\
&\quad + \mu_{L_{(i+1)}}^{e} \mu_{R_j}^{\Delta e} A_{i+1,j} \\
&= \mu_{R_i}^{e} \mu_{R_j}^{\Delta e} A_{i,j} + \mu_{R_i}^{e} (1 - \mu_{R_j}^{\Delta e}) A_{i,j+1} + (1 - \mu_{R_i}^{e})(1 - \mu_{R_j}^{\Delta e}) A_{i+1,j+1} \\
&\quad + (1 - \mu_{R_i}^{e}) \mu_{R_j}^{\Delta e} A_{i+1,j} \\
&= \frac{1}{w_e} \left[w_e - \bar{e}_i(k) \right] \frac{1}{w_{\Delta e}} \left[w_{\Delta e} - \Delta \bar{e}_j(k) \right] A_{i,j} \\
&\quad + \frac{1}{w_e} \left[w_e - \bar{e}_i(k) \right] \frac{1}{w_{\Delta e}} \Delta \bar{e}_j(k) A_{i,j+1} + \frac{1}{w_e} \bar{e}_i(k) \frac{1}{w_{\Delta e}} \Delta \bar{e}_j(k) A_{i+1,j+1} \\
&\quad + \frac{1}{w_e} \bar{e}_i(k) \frac{1}{w_{\Delta e}} \left[w_{\Delta e} - \Delta \bar{e}_j(k) \right] A_{i+1,j} \qquad (3.13)
\end{aligned}$$

We can rewrite (3.13) in terms of $\bar{e}_i(k)$ and $\Delta\bar{e}_j(k)$ in the following way:

$$\Delta u(k) = \frac{1}{w_e w_{\Delta e}} \left\{ \begin{array}{l} \Delta\bar{e}_j(k)[\bar{e}_i(k)A_{i+1,j+1} - \bar{e}_i(k)A_{i+1,j} - \bar{e}_i(k)A_{i,j+1} \\ +w_e A_{i,j+1} + \bar{e}_i(k)A_{i,j} - w_e A_{i,j}] \\ +\bar{e}_i(k)(-w_{\Delta e}A_{i,j} + w_{\Delta e}A_{i+1,j}) + w_e w_{\Delta e} A_{i,j} \end{array} \right\} \tag{3.14}$$

When designing a fuzzy PI controller, each singleton is calculated according to expression (3.8). Thus,

$$\begin{aligned} A_{i,j} &= K_{\text{Pd}} c_j^{\Delta e} + K_{\text{Id}} c_i^e \\ A_{i+1,j} &= K_{\text{Pd}} c_j^{\Delta e} + K_{\text{Id}} c_{i+1}^e = K_{\text{Pd}} c_j^{\Delta e} + K_{\text{Id}}(c_i^e + w_e) = A_{i,j} + K_{\text{Id}} w_e \\ A_{i,j+1} &= K_{\text{Pd}} c_{j+1}^{\Delta e} + K_{\text{Id}} c_i^e = K_{\text{Pd}}(c_j^{\Delta e} + w_{\Delta e}) + K_{\text{Id}} c_i^e = A_{i,j} + K_{\text{Pd}} w_{\Delta e} \\ A_{i+1,j+1} &= K_{\text{Pd}} c_{j+1}^{\Delta e} + K_{\text{Id}} c_{i+1}^e = K_{\text{Pd}}(c_j^{\Delta e} + w_{\Delta e}) + K_{\text{Id}}(c_i^e + w_e) \\ &= A_{i,j} + K_{\text{Pd}} w_{\Delta e} + K_{\text{Id}} w_e \end{aligned} \tag{3.15}$$

Upon insertion of (3.15) into (3.14) we obtain:

$$\Delta u(k) = K_{\text{Pd}} \Delta e(k) + K_{\text{Id}} e(k) \tag{3.16}$$

which is the same as the control law of a linear PI controller. In this way we have provided the equality of the two controllers. Since we are dealing with fuzzy emulation of linear control laws, all conclusions valid for a two-dimensional PI controller are also applicable to a three-dimensional PID controller. The only difference in the proof of equality is that in three dimensions we deal with eight singletons $A_{i,j,k} - A_{i+1,j+1,k+1}$ (vertices of the prismatic subspace of the control space, see Figure 3.6).

Another way to design a fuzzy PID controller using variant A is to treat the PID controller Equation (3.3) in its condensed form

$$\Delta u(k) = \Delta u_{\text{P}}(k) + \Delta u_{\text{I}}(k) + \Delta u_{\text{D}}(k) \tag{3.17}$$

where

$$\begin{aligned} \Delta u_{\text{P}}(k) &= K_{\text{Pd}}\,[e(k) - e(k-1)] = K_{\text{Pd}} \Delta e(k) \\ \Delta u_{\text{I}}(k) &= K_{\text{Id}} e(k) \\ \Delta u_{\text{D}}(k) &= K_{\text{Dd}}\,[\Delta e(k) - \Delta e(k-1)] = K_{\text{Dd}} \Delta^2 e(k) \end{aligned} \tag{3.18}$$

represent controller output increment contributions related to the system error, the change in error and the change in error rate, respectively.

Since $\Delta u_{\text{P}}(k)$, $\Delta u_{\text{I}}(k)$, and $\Delta u_{\text{D}}(k)$ are proportional to standard fuzzy control inputs $e(k)$, $\Delta e(k)$, and $\Delta^2 e(k)$ in (3.18), they can be treated as modified fuzzy

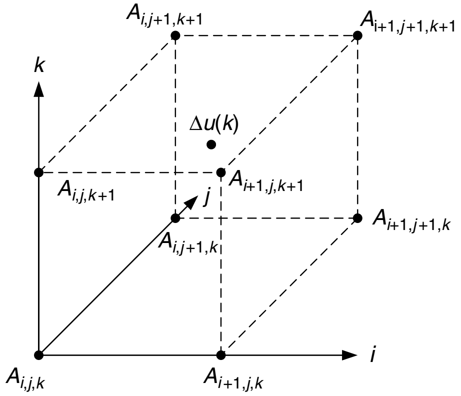


FIGURE 3.6 The segment of the fuzzy PID control space.

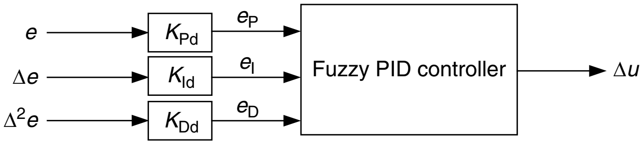


FIGURE 3.7 A variation of a fuzzy PID controller — variant A.

PID controller inputs $e_P(k) = \Delta u_P(k)$, $e_I(k) = \Delta u_I(k)$, and $e_D(k) = \Delta u_D(k)$, as shown in Figure 3.7. The difference with respect to the previous fuzzy PID controller form is in the input universe of discourse. Here, the domains of all inputs correspond to the domain of fuzzy controller output.

In order to make the fuzzification process linear, we must use evenly distributed fuzzy sets with uniform triangular membership functions, providing that only two adjacent membership functions are overlapping with crossover membership degree $\mu = 0.5$. The simplest possible case shown in Figure 3.8 has only two fuzzy sets, fuzzy set N with so-called Z-shape, fuzzy set P with so-called S-shape, and a predetermined threshold parameter equal to the expected maximum of controller output increment, $u_M = \max[\Delta u(k)]$. Normally, we may also have three, five, seven, or more fuzzy sets for each input, as shown in Figure 3.8 below (please notice that NL has a Z-shape and PL has an S-shape).

In the same fashion, we should arrange the even distribution of singletons or some other symmetrical membership functions along the controller output universe of discourse defined at interval $[-u_M, u_M]$. To create a rational fuzzy rule table, we would have to define at least three (N, Z, and P) or more controller output fuzzy sets.

Given the membership functions, a linear PID control law can be transformed into a set of fuzzy control rules. For input fuzzy sets having a form such as for the simplest case shown in Figure 3.8, we may create a fuzzy rule base with the total

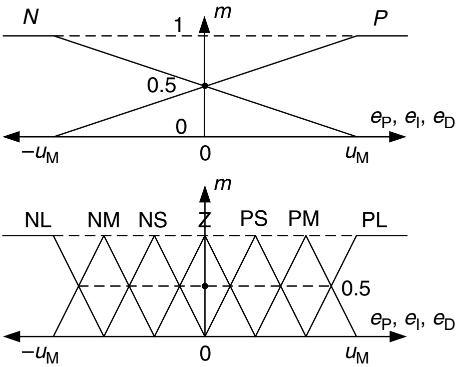


FIGURE 3.8 Input membership functions: the simplest case (above), a standard case (below).

of eight fuzzy control rules:

FR^1: IF e_P is N AND e_I is N AND e_D is N THEN Δu is A_1 (e.g., NL)

FR^2: IF e_P is N AND e_I is N AND e_D is P THEN Δu is A_2 (e.g., NS or NM)

⋮

FR^8: IF e_P is P AND e_I is P AND e_D is P THEN Δu is A_8 (e.g., PL)

Regardless of the number of fuzzy sets defined for each fuzzy PID controller input, if only two adjacent membership functions are overlapping, maximally eight fuzzy rules may contribute to fuzzy controller output.

Fuzzy PID controller output increment $\Delta u(k)$ is calculated for the discrete universe of discourse according to the center of gravity (COG) method in the following way:

$$\Delta u(e_\mathrm{P}, e_\mathrm{I}, e_\mathrm{D}, k) = \frac{\sum_i \Delta u_i \sum_{j=1}^{r} \mu_{\mathrm{FR}^j}(e_\mathrm{P}, e_\mathrm{I}, e_\mathrm{D}, \Delta u_i)}{\sum_i \sum_{j=1}^{r} \mu_{\mathrm{FR}^j}(e_\mathrm{P}, e_\mathrm{I}, e_\mathrm{D}, \Delta u_i)} \tag{3.19}$$

where $r \leq 8$ is the number of fuzzy rules activated by crisp inputs $e_\mathrm{P}(k)$, $e_\mathrm{I}(k)$, and $e_\mathrm{D}(k)$.

In DISO fuzzy controllers, the fuzzy rule base can be represented by a fuzzy rule table, while in three-input fuzzy PID controllers (3.17), the input space is a cube. The cube's dimension is defined by the inputs constraint $u_\mathrm{M} = \max[\Delta u(k)]$.

If controller input values $e_\mathrm{P}(k)$, $e_\mathrm{I}(k)$, and $e_\mathrm{D}(k)$ are such that they satisfy $\mu_i^{e_\mathrm{P}}(e_\mathrm{P}) = 1$, $\mu_j^{e_\mathrm{I}}(e_\mathrm{I}) = 1$, and $\mu_k^{e_\mathrm{D}}(e_\mathrm{D}) = 1$ (which means that $e_\mathrm{P}(k)$, $e_\mathrm{I}(k)$, and $e_\mathrm{D}(k)$ correspond to centers $c_i^{e_\mathrm{P}}$, $c_j^{e_\mathrm{I}}$, and $c_k^{e_\mathrm{D}}$ of the ith, jth, and the kth input membership functions, respectively), then just as in the previous case (see Equation [3.6]), controller output is determined by only one rule regardless of the used min or product T-norm. Its value is equal to

$$
\begin{aligned}
u_{\mathrm{FC}} &= \frac{\min[\mu_i^{e\mathrm{P}}(c_i^{e\mathrm{P}}), \mu_j^{e\mathrm{I}}(c_j^{e\mathrm{I}}), \mu_j^{e\mathrm{D}}(c_k^{e\mathrm{D}})]\mathrm{A}_q}{\min[\mu_i^{e\mathrm{P}}(c_i^{e\mathrm{P}}), \mu_j^{e\mathrm{I}}(c_j^{e\mathrm{I}}), \mu_j^{e\mathrm{D}}(c_k^{e\mathrm{D}})]} = \frac{\min[1,1,1]\mathrm{A}_q}{\min[1,1,1]} = \mathrm{A}_q \\
u_{\mathrm{FC}} &= \frac{\mu_i^{e\mathrm{P}}(c_i^{e\mathrm{P}}) \cdot \mu_j^{e\mathrm{I}}(c_j^{e\mathrm{I}}) \cdot \mu_j^{e\mathrm{D}}(c_k^{e\mathrm{D}}) \cdot \mathrm{A}_q}{\mu_i^{e\mathrm{P}}(c_i^{e\mathrm{P}}) \cdot \mu_j^{e\mathrm{I}}(c_j^{e\mathrm{I}}) \cdot \mu_j^{e\mathrm{D}}(c_k^{e\mathrm{D}})} = \frac{1 \cdot 1 \cdot 1 \cdot \mathrm{A}_q}{1 \cdot 1 \cdot 1} = \mathrm{A}_q
\end{aligned}
\tag{3.20}
$$

By equating (3.17) with (3.20) (having in mind that $e_{\mathrm{P}}(k) = \Delta u_{\mathrm{P}}(k)$, $e_{\mathrm{I}}(k) = \Delta u_{\mathrm{I}}(k)$, and $e_{\mathrm{D}}(k) = \Delta u_{\mathrm{D}}(k)$), we obtain

$$\Delta u = u_{\mathrm{FC}} = \mathrm{A}_q = c_i^{e\mathrm{P}} + c_j^{e\mathrm{I}} + c_k^{e\mathrm{D}} \tag{3.21}$$

Equation (3.21) directly defines values of all output singletons A_q, $1 \leq q \leq l^3$ by inserting values of all input membership function centers $c_i^{e\mathrm{P}}$, $c_j^{e\mathrm{I}}$, and $c_k^{e\mathrm{D}}$ for $i, j, k = 1, 2, \ldots, l$.

The result is, as in the first approach, a simple algebraic equation for calculating singleton values that can be easily implemented into any control software.

The aim of the two described approaches is to get a fuzzy PID controller which can be further modified by means of various adaptive and self-organizing algorithms. The only problem is in the large number of fuzzy rules, for $i, j, k = 1, 2, \ldots, l$, $l = 5$, it reaches 125.

3.1.1.2 Fuzzy Emulation of a PID Controller — Variant B

If our goal is to minimize the number of rules, then we may use a very simple configuration of a fuzzy PID controller, shown in Figure 3.2. This is a structure which contains a SISO fuzzy controller and a standard linear PID controller [13]. The fuzzy controller has $e(k)$ as its input and $e_{\mathrm{F}}(k)$ as its output. The number of input fuzzy sets l defines the total number of fuzzy rules. Providing that we are using triangular fuzzy sets, where only two adjacent sets are overlapping at $\mu = 0.5$ (see Figure 3.9), then maximally two fuzzy rules can contribute to crisp output $e_{\mathrm{F}}(k)$. The output $e_{\mathrm{F}}(k)$ can be generated according to the COG principle. Then the output of the SISO fuzzy controller gets the form:

$$e_{\mathrm{F}}(k) = \mu_{R_i}^{e} c_i^{e\mathrm{F}} + \mu_{L(i+1)}^{e} c_{(i+1)}^{e\mathrm{F}} \tag{3.22}$$

where $\mu_{R_i}^{e}$ denotes the right-hand side of the fuzzy set TE_i, while $\mu_{L(i+1)}^{e}$ denotes the left-hand side of the fuzzy set TE_{i+1}. Notations $c_i^{e\mathrm{F}}$ and $c_{(i+1)}^{e\mathrm{F}}$ stand for the centers of output fuzzy sets TEF_i and TEF_{i+1}, respectively.

The corresponding widths of the halves of fuzzy sets are $w_e = (e_{\max} - e_{\min})/(l-1)$ and $w_{e_{\mathrm{F}}} = (e_{\mathrm{F}_{\max}} - e_{\mathrm{F}_{\min}})/(l_{\mathrm{F}} - 1)$ for linearly distributed input and output

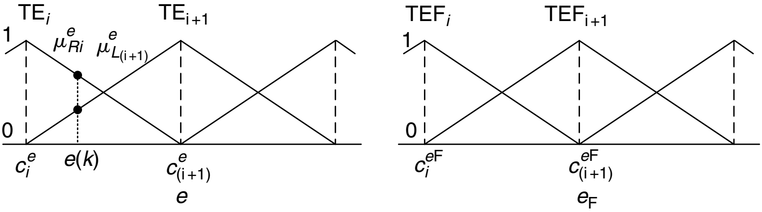


FIGURE 3.9 Membership functions of a SISO fuzzy controller.

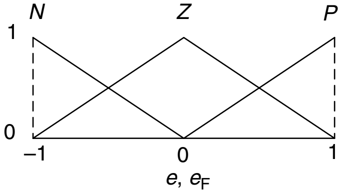


FIGURE 3.10 Membership functions of a three-rule fuzzy PID controller.

fuzzy sets. Since $\mu^e_{L_{(i+1)}} = 1 - \mu^e_{R_i}$, and $c^{e_F}_{(i+1)} = c^{e_F}_i + w_{e_F}$, by recalling (3.12) we get:

$$e_F(k) = \left(1 - \frac{\bar{e}_i(k)}{w_e}\right) c^{e_F}_i + \frac{\bar{e}_i(k)}{w_e}(c^{e_F}_i + w_{e_F}) = c^{e_F}_i + \frac{w_{e_F}}{w_e}\bar{e}_i(k)$$

$$\rightarrow e_F(k) - c^{e_F}_i = \frac{w_{e_F}}{w_e}[e(k) - c^e_i] = k_{lF}[e(k) - c^e_i] \qquad (3.23)$$

Example 3.1 A three-rule SISO fuzzy PID controller.

Let us consider a SISO fuzzy PID controller whose input and output have only three linearly distributed triangular fuzzy sets, $l = l_F = 3$, where only two adjacent sets are overlapping at $\mu = 0.5$. Also, let the input and output universes of discourse be normalized, as shown in Figure 3.10.

Then the fuzzy rule table has only these three rules:

FR^1: IF e is N THEN e_F is N

FR^2: IF e is Z THEN e_F is Z

FR^3: IF e is P THEN e_F is P

Dealing with normalized input and output universes of discourse and having $l = l_F$, according to (3.23) we get $k_{lF} = 1$, $c^{e_F}_i = c^e_i$, which eventually yields $e_F(k) = e(k)$.

Then, the output of the SISO fuzzy PID controller gets the form:

$$\begin{aligned}\Delta u(k) &= K_{\text{Id}} e_{\text{F}}(k) + K_{\text{Pd}} \Delta e_{\text{F}}(k) + K_{\text{Dd}} \Delta^2 e_{\text{F}}(k) = |e_{\text{F}}(k) = k_{\text{lF}} e(k)|_{k_{\text{lF}}=1} \\ &= K_{\text{Id}} e(k) + K_{\text{Pd}} \Delta e(k) + K_{\text{Dd}} \Delta^2 e(k) \qquad (3.24)\end{aligned}$$

We have achieved full compatibility of a linear and a fuzzy PID controller, but it would be much more effective to use a nonlinear potential of the SISO fuzzy controller (i.e., to adjust the nonlinear input–output mapping gain k_{lF} on-line).

3.1.1.3 Fuzzy Emulation of a PID Controller — Variant C

Let us see what we get if we consider a fuzzy PID controller composed of three parallel SISO fuzzy controllers — fuzzy P, fuzzy I, and fuzzy D — having $e(k)$, $\Delta e(k)$, and $\Delta^2 e(k)$ as inputs, and $u_{\text{P}}(k)$, $u_{\text{I}}(k)$, and $u_{\text{D}}(k)$ as respective outputs (Figure 3.3). Following the thinking behind the SISO fuzzy PID controller structure described in the previous section, the number of fuzzy sets for each input, l_e, $l_{\Delta e}$, and $l_{\Delta\Delta e}$, will determine the total number of fuzzy rules, equal to $l_e + l_{\Delta e} + l_{\Delta\Delta e}$. In this way, for $l_e = l_{\Delta e} = l_{\Delta\Delta e} = l$, we can have a complete fuzzy rule base with only $3 \times l$ fuzzy rules. Compared to l^3 fuzzy rules of the fuzzy PID controller — variant A, the number of rules is significantly reduced.

Dealing with three parallel SISO fuzzy controllers, we may apply the same approach we applied to the analysis of the SISO fuzzy PID controller — variant B. Generally, we deal with different values of l_e, $l_{\Delta e}$, and $l_{\Delta\Delta e}$, and $l_{\Delta u_{\text{I}}}$, $l_{\Delta u_{\text{P}}}$, and $l_{\Delta u_{\text{D}}}$. The same holds for the corresponding widths of fuzzy sets. Referring to Figure 3.9 and relation (3.22), we obtain:

$$\begin{aligned}\Delta u_{\text{I}}(k) &= \mu^{e}_{R_i} c^{\Delta u_{\text{I}}}_{i} + \mu^{e}_{L_{(i+1)}} c^{\Delta u_{\text{I}}}_{i+1} \\ \Delta u_{\text{P}}(k) &= \mu^{\Delta e}_{R_j} c^{\Delta u_{\text{P}}}_{j} + \mu^{\Delta e}_{L_{(j+1)}} c^{\Delta u_{\text{P}}}_{j+1} \qquad (3.25) \\ \Delta u_{\text{D}}(k) &= \mu^{\Delta\Delta e}_{R_k} c^{\Delta u_{\text{D}}}_{k} + \mu^{\Delta\Delta e}_{L_{(k+1)}} c^{\Delta u_{\text{D}}}_{k+1}\end{aligned}$$

For the given widths and centers of all fuzzy sets, as in Equation (3.23), we get:

$$\begin{aligned}\Delta u_{\text{I}}(k) - c^{\Delta u_{\text{I}}}_{i} &= k_{\text{lI}}[e(k) - c^{e}_{i}] \\ \Delta u_{\text{P}}(k) - c^{\Delta u_{\text{P}}}_{j} &= k_{\text{lP}}[\Delta e(k) - c^{\Delta e}_{j}] \qquad (3.26) \\ \Delta u_{\text{D}}(k) - c^{\Delta u_{\text{D}}}_{k} &= k_{\text{lD}}[\Delta^2 e(k) - c^{\Delta\Delta e}_{k}]\end{aligned}$$

This configuration allows the designer to adjust three independent input–output mapping gains k_{lI}, k_{lP}, and k_{lD} to achieve the desired nonlinear effect.

3.1.1.4 Sugeno Type of Fuzzy PID Controller

One of the many possibilities in designing a fuzzy controller that behaves as a linear PID controller is to use the structure of a so-called Sugeno type of fuzzy PID controller. Namely, the Takagi–Sugeno fuzzy rule has an explicit function in the consequent (THEN) part of the rule:

$$\text{FR:} \quad \text{IF } R \text{ THEN } f(\cdot)$$

If that function happens to be a recursive Equation (3.3) of a PID controller:

$$\text{FR:} \quad \text{IF } R \text{ THEN } \Delta u(k) = K_{\text{Id}}e(k) + K_{\text{Pd}}\Delta e(k) + K_{\text{Dd}}\Delta^2 e(k) \tag{3.27}$$

and if there are as many different PID controller functions as there are rules, then we have the perfect opportunity to change (adapt) the parameters of the PID controller with respect to the values of fuzzy controller inputs. There are no constraints regarding the number of inputs: the Sugeno type of fuzzy PID controller may have a single input (e.g., $e(k)$), or several inputs (e.g., $e(k)$, $\Delta e(k)$, and $\Delta^2 e(k)$).

3.2 Model Reference-Based Initial Setting of Fuzzy Controllers

In this chapter, we describe the initial setting of a fuzzy rule table by using a second-order reference model for defining the desired closed-loop system dynamics

$$y_{\text{M}}(k) = a_{\text{M1}}y_{\text{M}}(k-1) + a_{\text{M2}}y_{\text{M}}(k-2) + b_{\text{M1}}u_{\text{r}}(k-1) \tag{3.28}$$

where y_{M} is the reference model output and u_{r} is the system reference input.

The method is based on the assumption that a controlled process with measurable input and output can be appropriately described in a selected operating point with linear second order approximation:

$$y_{\text{A}}(k) = a_{\text{A1}}y_{\text{A}}(k-1) + a_{\text{A2}}y_{\text{A}}(k-2) + b_{\text{A1}}u(k-1) \tag{3.29}$$

where y_{A} is process output and u is control input (e.g., fuzzy controller output). The former assumption is true for a very large class of linear and nonlinear systems. Process approximation parameters a_{A1}, a_{A2}, and b_{A1} can be calculated from the acquired input–output data by using some of the standard process identification methods (e.g., the least square method).

The goal of fuzzy controller design is to find a controller that can keep the difference (i.e., tracking error) between the reference model and the process as small as possible. Because controller design is based on process approximation

(3.29), the controller dealing with nonmodeled process dynamics will not be able to fully eliminate the tracking error. In general, the better the process approximation is, the lower the tracking error will be.

The problem of fuzzy controller parameter determination is equal to controller synthesis, which will set closed-loop poles and zeros in the place of reference model poles and zeros. In the ideal case, closed-loop system behavior will not vary from the reference model.

Transfer functions of process approximation and of the reference model can be obtained from (3.28) and (3.29)

$$\begin{aligned} G_{\mathrm{A}}(z) &= \frac{Y_{\mathrm{A}}(z)}{U(z)} = \frac{B_{\mathrm{A}}(z)}{A_{\mathrm{A}}(z)} = \frac{b_{\mathrm{A}1}z}{z^2 - a_{\mathrm{A}1}z - a_{\mathrm{A}2}} \\ G_{\mathrm{M}}(z) &= \frac{Y_{\mathrm{M}}(z)}{U_{\mathrm{r}}(z)} = \frac{B_{\mathrm{M}}(z)}{A_{\mathrm{M}}(z)} = \frac{b_{\mathrm{M}1}z}{z^2 - a_{\mathrm{M}1}z - a_{\mathrm{M}2}} \end{aligned} \tag{3.30}$$

The generic form of a controller with two inputs, reference input u_{r} and measurement signal y_{A}, and one output $u(k)$, is described with

$$U(z) = \frac{1}{R(z)}\left[T(z)U_{\mathrm{r}}(z) - S(z)Y_{\mathrm{A}}(z)\right] \tag{3.31}$$

Controller (3.31) is the most frequently used controller in conventional control systems. It is a two-parameter configuration with system output as its feedback signal [21]. By selecting polynomials $R(z)$, $S(z)$, and $T(z)$, different structures of the control algorithm can be obtained depending on the desired dynamic behavior and the criterion for control quality. Thus, for $S = 0$, we get feedforward control, while for $S = T$ the feedforward part is excluded and the control system uses a feedback signal only.

The degrees of polynomials $R(z)$, $S(z)$, and $T(z)$ (i.e., controller degree) are defined by the request for causality and stability of a closed-loop system and its controller (i.e., by the degrees of polynomials $A_{\mathrm{A}}(z)$, $B_{\mathrm{A}}(z)$, $A_{\mathrm{M}}(z)$, and $B_{\mathrm{M}}(z)$) satisfying the following criteria:

$$\begin{aligned} \text{deg of } R(z) &\geq \text{deg of } T(z) \\ \text{deg of } R(z) &\geq \text{deg of } S(z) \end{aligned} \tag{3.32}$$

By insertion of (3.31) into (3.30) we get the well-known equality from which controller polynomials can be determined:

$$\frac{B_{\mathrm{A}}(z)T(z)}{A_{\mathrm{A}}(z)R(z) + B_{\mathrm{A}}(z)S(z)} = \frac{B_{\mathrm{M}}(z)}{A_{\mathrm{M}}(z)} \tag{3.33}$$

In general, before determining controller polynomials, we need to factorize polynomials B, B_{M}, and R to solve the problem of process zeros, which are placed outside the unity circle. With the determination of initial values of the controller's output singletons, we make the assumption that the transfer function of the process approximation does not contain such zeroes: therefore, factorization is not needed.

From Equation (3.33) we see that the poles of a closed-loop characteristic equation are determined by polynomial $A_{\mathrm{M}}(z)$. The degree of the model is usually lower or equal to the degree of the controlled process, so it is convenient to introduce a new polynomial, $A_0(z)$, that multiplies $A_{\mathrm{M}}(z)$ and contains poles, which can be interpreted as poles of an observer. Namely, a controller with a two-parameter configuration and the system output as the feedback signal implicitly contains an observer. In order to keep equality (3.33) sustained, polynomial $B_{\mathrm{M}}(z)$ must also be multiplied with $A_0(z)$.

Polynomials $R(z)$ and $S(z)$, which determine poles of a closed-loop characteristic equation, are calculated by solving this polynomial equation

$$A_{\mathrm{A}}(z)R(z) + B_{\mathrm{A}}(z)S(z) = A_0(z)A_{\mathrm{M}}(z) \tag{3.34}$$

while the solution to the equation

$$B_{\mathrm{A}}(z)T(z) = A_0(z)B_{\mathrm{M}}(z) \tag{3.35}$$

determines polynomial $T(z)$.

From Equations (3.34) and (3.35) we may see that the degrees of polynomials $R(z)$, $S(z)$, and $T(z)$ are determined by the degrees of the respective polynomials of model and process approximation transfer functions, which must satisfy the criteria of controller causality and closed-loop system stability (3.32).

In general, solving the polynomial equation (3.34) can be a demanding job. Since transfer functions (3.30) are very simple, finding the solution can also become simple if polynomials $R(z)$, $S(z)$, and $T(z)$ are selected as follows:

$$\begin{aligned} R(z) &= r_1 z + r_0 \\ S(z) &= s_1 z + s_0 \\ T(z) &= t_1 z + t_0 \end{aligned} \tag{3.36}$$

Since $\deg\{R(z)\} = \deg\{S(z)\} = \deg\{B_{\mathrm{A}}(z)\} = 1$, and $\deg\{A_{\mathrm{A}}(z)\} = \deg\{A_{\mathrm{M}}(z)\} = 2$, $A_0(z)$ is chosen to be a first degree polynomial with a pole placed in the origin of the z-plane, $A_0(z) = z$. In this way, the influence of $A_0(z)$ on system dynamics is reduced to the minimum. By inserting (3.36) and polynomial $A_0(z)$ in Equations (3.34) and (3.35) we get

$$\begin{aligned} &(z^2 - a_{\mathrm{A1}}z - a_{\mathrm{A2}})(r_1 z + r_0) + b_{\mathrm{A1}}z(s_1 z + s_0) = z(z^2 - a_{\mathrm{M1}}z - a_{\mathrm{M2}}) \\ &b_{\mathrm{A1}}z(t_1 z + t_0) = b_{\mathrm{M1}}z^2 \end{aligned} \tag{3.37}$$

To find a controller by solving Equations (3.37), let $r_1 = 1$ and $r_0 = 0$. Then controller polynomials are

$$\begin{aligned} R(z) &= z \\ S(z) &= \frac{a_{\mathrm{A1}} - a_{\mathrm{M1}}}{b_{\mathrm{A1}}} z + \frac{a_{\mathrm{A2}} - a_{\mathrm{M2}}}{b_{\mathrm{A1}}} \\ T(z) &= \frac{b_{\mathrm{M1}}}{b_{\mathrm{A1}}} z \end{aligned} \tag{3.38}$$

Upon the insertion of polynomials of Equation (3.38) into controller equation (3.31), we obtain

$$U(z) = \frac{b_{\mathrm{M1}}}{b_{\mathrm{A1}}} U_{\mathrm{r}}(z) - \left[\frac{a_{\mathrm{A1}} - a_{\mathrm{M1}}}{b_{\mathrm{A1}}} + \frac{a_{\mathrm{A2}} - a_{\mathrm{M2}}}{b_{\mathrm{A1}}} z^{-1} \right] Y_{\mathrm{A}}(z) \tag{3.39}$$

The inverse Z-transformation of (3.39) gives a recursive controller equation

$$u(k) = \frac{b_{\mathrm{M1}}}{b_{\mathrm{A1}}} u_{\mathrm{r}}(k) - \frac{a_{\mathrm{A1}} - a_{\mathrm{M1}}}{b_{\mathrm{A1}}} y_{\mathrm{A}}(k) + \frac{a_{\mathrm{A2}} - a_{\mathrm{M2}}}{b_{\mathrm{A1}}} y_{\mathrm{A}}(k-1) \tag{3.40}$$

From controller equation (3.40) we may see that when the reference model and process approximation dynamics are equal, signals $u(k) = u_{\mathrm{r}}(k)$ are equal. The most frequent form of fuzzy controllers has control error $e(k)$ and change of control error $\Delta e(k)$ as its inputs. In order to get the form of controller (3.40) compatible with the form of the fuzzy controller, we must transform Equation (3.40) so that it includes fuzzy controller inputs $e(k)$ and $\Delta e(k)$. As $e(k) = u_{\mathrm{r}}(k) - y_{\mathrm{A}}(k)$, Equation (3.40) obtains the following form:

$$\begin{aligned} u(k) = {} & \frac{a_{\mathrm{A1}} + a_{\mathrm{A2}} - a_{\mathrm{M1}} - a_{\mathrm{M2}}}{b_{\mathrm{A1}}} e(k) + \frac{a_{\mathrm{M2}} - a_{\mathrm{A2}}}{b_{\mathrm{A1}}} \Delta e(k) \\ & + \frac{a_{\mathrm{M1}} - a_{\mathrm{A1}} + b_{\mathrm{M1}}}{b_{\mathrm{A1}}} u_{\mathrm{r}}(k) + \frac{a_{\mathrm{M2}} - a_{\mathrm{A2}}}{b_{\mathrm{A1}}} u_{\mathrm{r}}(k-1) \end{aligned} \tag{3.41}$$

Assuming that reference input signal $u_{\mathrm{r}}(k)$ has a constant value or that it is changing slowly (i.e., $u_{\mathrm{r}}(k) = u_{\mathrm{r}}(k-1)$), Equation (3.21) becomes

$$u(k) = k_1 e(k) + k_2 \Delta e(k) + k_3 u_{\mathrm{r}}(k) \tag{3.42}$$

where

$$\begin{aligned} k_1 &= \frac{a_{\mathrm{A1}} + a_{\mathrm{A2}} - a_{\mathrm{M1}} - a_{\mathrm{M2}}}{b_{\mathrm{A1}}} \\ k_2 &= \frac{a_{\mathrm{M2}} - a_{\mathrm{A2}}}{b_{\mathrm{A1}}} \\ k_3 &= \frac{a_{\mathrm{M1}} - a_{\mathrm{A1}} + b_{\mathrm{M1}} + a_{\mathrm{M2}} - a_{\mathrm{A2}}}{b_{\mathrm{A1}}} \end{aligned} \tag{3.43}$$

As in Equation (3.40), one may see from (3.42) and (3.43) that signals $u(k)$ and $u_\mathrm{r}(k)$ will be equal if process approximation and the reference model are equal.

Controller (3.42) can be split into two parts

$$u(k) = u_\mathrm{FC}(k) + u_\mathrm{FF}(k) \tag{3.44}$$

where

$$u_\mathrm{FC}(k) = k_1 e(k) + k_2 \Delta e(k) \tag{3.45}$$

$$u_\mathrm{FF}(k) = k_3 u_\mathrm{r}(k) \tag{3.46}$$

Coefficient k_3 represents feedforward gain coefficient and Equation (3.46) represents the feedforward part of a controller that works in parallel with the "fuzzy" part of controller (3.45). As assumed, the controlled process is stable (i.e., it does not contain integral behavior) and since we are dealing with a PD-type of fuzzy controller, it is necessary to include a feedforward path in the controller that will compensate for static error. As a consequence, it is essential to determine the correct value for k_3. We may see from Equation (3.43) that if we have a reference model with unity gain ($b_\mathrm{M1} = 1$), k_3 is equal to an inverse value of the process gain coefficient.

Equation (3.45) becomes the basis for model reference-based fuzzy controller design. We may notice a great similarity with Equation (3.8) obtained for the design of a fuzzy P-I controller. If only two nearest input fuzzy sets overlap, maximally one, two, or four fuzzy rules can contribute to crisp controller output value. Following the same idea which has been used for the determination of the fuzzy rule table that emulates P-I-D and P-I algorithms (see Equations [3.7] and [3.8] in Section 3.1.2), we may choose the values of controller inputs $e_i(k)$ and $\Delta e_j(k)$ such that $\mu_i^e(e_i) = 1$ and $\mu_j^{\Delta e}(\Delta e_j) = 1$ (which means that $e_i(k)$ and $\Delta e_j(k)$ correspond with the centers c_i^e, $c_j^{\Delta e}$of the ith and the jth input fuzzy sets, respectively). In that case crisp fuzzy controller output is determined by only one fuzzy rule, that is,

$$u_\mathrm{FC} = k_1 c_i^e + k_2 c_j^{\Delta e} = \mathrm{A}_q \tag{3.47}$$

Equation (3.47) directly defines values of all output singletons A_q, $1 \leq q \leq l^2$ by inserting values of all input fuzzy set centers c_i^e and $c_j^{\Delta e}$ for $i, j = 1, 2, \ldots, l$.

By solving polynomial Equation (3.44) for second-order process (3.40), we have derived a controller whose aim is to enforce a closed-loop system to follow reference model dynamics. The result is the sum of the reference model-dependent feedforward term and the algebraic equation (3.47) similar in form to Equation (3.8). This algebraic expression, which includes parameters of the reference model, is used for the calculation of controller output singleton values. Due to its simplicity, it can be implemented into control software easily.

A large number of industrial processes that feature dead time T_m can be described with the recursive equation (3.29). However, it is necessary to modify the delay of signal $u(k)$ for the number of control intervals equal to $d = T_{\mathrm{m}}/T_{\mathrm{d}}$:

$$y_{\mathrm{A}}(k) = a_{\mathrm{A}1}y_{\mathrm{A}}(k-1) + a_{\mathrm{A}2}y_{\mathrm{A}}(k-2) + b_{\mathrm{A}1\mathrm{d}}u(k-1-d) \tag{3.48}$$

We can also use the model reference-based method for the initial setting of the fuzzy rule table for this class of systems. Besides the feedforward term, new elements related to the previous states of controller output $u(k)$ will appear:

$$u(k) = u_{\mathrm{FC}}(k) + u_{\mathrm{FF}}(k) + k_4u(k-1) + k_5u(k-2) + \cdots + k_{d+2}u(k-d+1) \tag{3.49}$$

These changes occur because the fuzzy controller part (3.45) does not contain memory elements, that is, its output does not depend on its previous states, due to the purely static character of the fuzzy controller input–output mapping function. When controlling systems with dead time, memory elements must be added to ensure causality (feasibility) of the controller.

3.3 Phase Plane-Based Initial Setting of Fuzzy Controllers

Fuzzy rule tables obtained by emulating linear controllers and by using reference models can be characterized as "linear" due to the linear character of their initial setting algorithms by Equations (3.7), (3.8), and (3.47). This is a severe constraint if we wish to mimic human operator decisions or existing nonlinear controller actions while they control a process. Phase plane-based initial setting of the fuzzy controller is proposed as a solution to this problem.

As discussed in Section 2.4.1, the fuzzy rule table can also be viewed as the phase plane, while singleton values in the fuzzy rule table, depending on a type of defuzzification, form the control surface ψ above the phase plane. Provided that the controller inputs and output are measurable, we can extract (record) triples of the form $[u(k), e(k), \Delta e(k)]$ or $[\Delta u(k), e(k), \Delta e(k)]$ in every control interval, depending on whether the controlled process is astatic or static. Triples often have the form $[u(k), e(k), \Delta y_{\mathrm{f}}(k)]$ or $[\Delta u(k), e(k), \Delta y_{\mathrm{f}}(k)]$ where a change of control error $\Delta e(k)$ has been replaced with the change of measured system output $\Delta y_{\mathrm{f}}(k)$. By using triples acquired under different operating conditions of a mimicked controller, we may form a set of phase plane trajectories and accompanying controller output responses (series of connected discrete points), which not only belong to but also constitute fuzzy control surface ψ. The main advantage of phase plane-based synthesis of a fuzzy controller is that it does not depend on the type of a mimicked controller. The mimicked controller can be any type of linear or nonlinear controller that includes a system operator, which means that the controller is treated as a "black box."

We shall describe phase plane-based synthesis of the fuzzy rule table by using triples $[u(k), e(k), \Delta e(k)]$. The procedure is, moreover, applicable to all the other

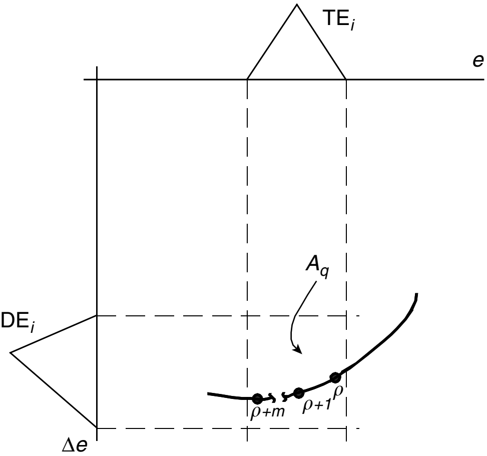


FIGURE 3.11 The determination of output singletons by using phase plane trajectory. (From Bogdan, S. and Kovačić, Z., *IEEE Conf. Control Appl.*, 648–652, 1998. With permission.)

forms we have described. Accordingly,

$$u_{\mathrm{FC}} = \psi[e, \Delta e] \tag{3.50}$$

where $\psi[e, \Delta e]$ is calculated according to the center of gravity method described in Equation (2.23).

To simplify fuzzy controller design we can start by defining the number of membership functions and their shapes for inputs $e(k)$ and $\Delta e(k)$, as well as their distribution on the universes of discourse. In that case, the tuning of the controller can be reduced to the tuning of output singletons, parameters that will model the control surface ψ.

Let the jth trajectory (partly shown in Figure 3.11) contain a series of n points expressed with triples $[u^j(1), e(1), \Delta e(1)]$, $[u^j(2), e(2), \Delta e(2)]$, ... , $[u^j(\rho), e(\rho), \Delta e(\rho)]$, $[u^j(\rho+1), e(\rho+1), \Delta e(\rho+1)], \ldots, [u^j(\rho+m), e(\rho+m), \Delta e(\rho+m)], \ldots$, $[u^j(n), e(n), \Delta e(n)]$. By writing Equation (3.50) for each trajectory point and assuming that both input variables have the same number of l fuzzy sets, we get a set of n equations with l^2 unknowns.

Because of overlapping fuzzy controller membership functions, several fuzzy rules contribute to crisp output value. In our case, this means that several output singletons contribute to it. Providing that only two neighboring input membership functions overlap, which is often the case, we can write Equation (2.23) for the ρth trajectory point as

$$\begin{aligned} &\mathrm{A}_q^i \varphi_q(\rho+i) + A_{q+1}^i \varphi_{q+1}(\rho+i) + A_{q+2}^i \varphi_{q+2}(\rho+i) + A_{q+3}^i \varphi_{q+3}(\rho+i) \\ &\quad = \psi[e(\rho+i), \Delta e(\rho+i)] = u_{\mathrm{FC}}(\rho+i) \quad \text{for } i = 0, 1, 2, \ldots, m \end{aligned} \tag{3.51}$$

where m is the number of trajectory points that activate the same fuzzy rule FR^q (see Figure 3.11).

Finding the solution for the set of m equations of form (3.51) is rather complex. The condition for the solution's existence is that every rule involved must be activated at least three times for every trajectory. That is not often the case, especially if the controller has many rules. This is why we need to apply a procedure for the approximate determination of output singletons in order to get an approximate model of the control surface ψ.

Among n points of the jth trajectory there exist m of them that activate the qth fuzzy rule FR^q. This fuzzy rule has singleton A$_q$ in its consequent part. Among those m points let us find the point on which singleton A$_q$ has the largest influence (i.e., the largest fuzzy basis function)

$$\varphi_q^j(\rho + \vartheta) = \sup_{\rho \le i \le \rho + m} \{\varphi_q^j(i)\} \tag{3.52}$$

and let us assume that the contributions of other rules may be neglected. Then, we get an approximate relation:

$$\mathrm{A}_q^j \varphi_q^j(\rho + \vartheta) = u^j(\rho + \vartheta) \tag{3.53}$$

Accordingly, the value of output singleton A$_q$ determined by the jth trajectory is approximately determined as

$$\mathrm{A}_q^j = \frac{u^j(\rho + \vartheta)}{\varphi_q^j(\rho + \vartheta)} \tag{3.54}$$

Essentially, this described method represents the search for a trajectory point that has the greatest influence on a particular fuzzy rule. When interpreting the method graphically, we may say that we are looking for the trajectory point, which lies nearest to singleton A$_q$. The more accurate the result of this simple algorithm, the larger the fuzzy basis function $\varphi_q^j(\rho + \vartheta)$ will be with respect to its counterparts from other contributing rules. To do this, we can embed a mechanism in the initial setting algorithm. This mechanism would take into consideration only those trajectory points which contribute to singleton A$_q$ significantly more than other points.

In case that fuzzy rule FR^q has been activated by more than one trajectory, the final value of singleton A$_q$ is equal to the mean value of singletons obtained from (3.54):

$$\mathrm{A}_q = \frac{\sum_{j=1}^{\xi} \mathrm{A}_q^j}{\xi} \tag{3.55}$$

where ξ is the number of trajectories that activated fuzzy rule FR^q.

The method requires several different trajectories because the points of one trajectory activate only a subset of fuzzy rules, not all of them. If some rules remain idle after the determination of output singletons, the corresponding empty places in the fuzzy rule table can be filled by other initial setting procedures described in the previous chapters, or they can be left empty.

Phase plane-based initial setting method requires real-time operation of the control system. This can become advantageous in those systems which are already in routine exploitation and which are controlled, for example, by conventional linear controllers. The method of phase plane-based initial setting of the fuzzy rule table by using relations (3.52) to (3.55) can be easily implemented. As such, it is intended to mimic and replace various existing controllers in industry (in Section 7.2.3 we shall show how this method of initial setting has been effectively used in a fuzzy controller function block for industrial programmable logic controllers [PLCs]).

3.4 Practical Examples: Initial Setting of a Fuzzy Controller

The aim of the three methods described in Sections 3.1 to 3.3 is to allow for automated initial setting of the fuzzy controller rule base assuming that the designer has already defined the number of membership functions and their shapes for inputs $e(k)$ and $\Delta e(k)$, as well as their distribution on the universes of discourse. This means that a selected initial setting method is supposed to be implemented as a part of the fuzzy controller algorithm, that is, as its additional feature, which would partly or completely substitute the heuristic way of fuzzy controller commissioning [22].

Let us show step-by-step how each of the three methods were worked out and tested experimentally on selected laboratory equipment shown in Figure 3.12. Process simulator Feedback PCS 327 enabled the physical simulation of a linear high-order controlled process with the following transfer function,

$$G_{\mathrm{p}}(s) = \frac{0.94}{(1+0.5s)(1+s)(1+3s)} \tag{3.56}$$

Controller output voltage was fed to the process through a 12-bit digital-to-analog (D/A) output channel, while the voltage value of the process output was taken from the simulator panel and fed to a 12-bit analog-to-digital (A/D) input channel of the PC I/O board. The voltage range of A/D and D/A converters was ±9 V, which determined input and output universes of discourse on the interval $[-2048, 2047]$. Control algorithms were implemented into a personal computer and executed in real time with control interval $T_{\mathrm{d}} = 200$ msec. A noise generator was used to add white noise to the system output, simulating noise picked up by a sensor and wiring. Noise amplitude was set to 5% of reference input magnitude.

We usually design controllers around a selected operating point. In the control of nonlinear systems, the higher the system nonlinearity is, the narrower operating range of a linear controller will be. The same holds for determining the universe

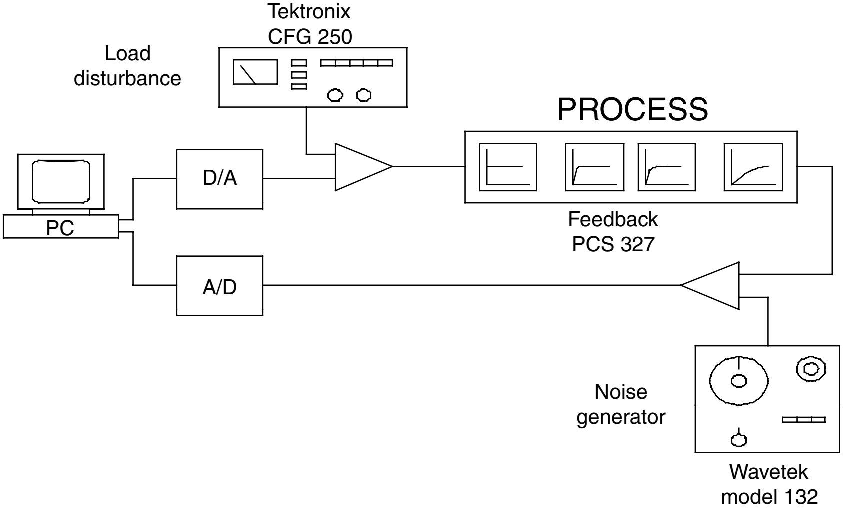


FIGURE 3.12 A laboratory setup for the experimental validation of methods for the initial setting of a fuzzy controller's fuzzy rule table.

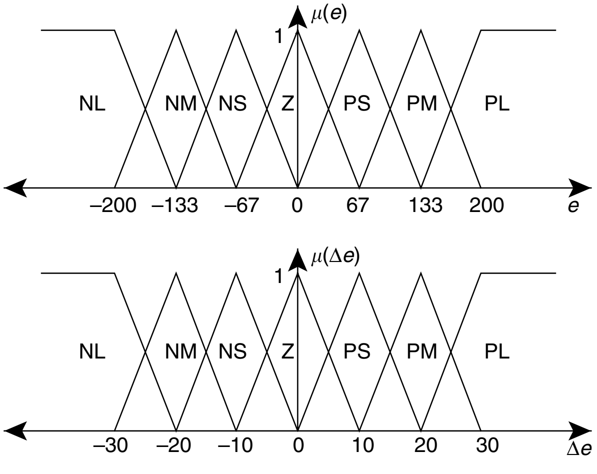


FIGURE 3.13 The distribution of input membership functions.

of discourse of compatible fuzzy controllers. In our example, we expect inputs to take values from the constrained voltage range ±0.88 V that defines input domains $e \in [-200, 200]$ and $\Delta e \in [-30, 30]$ from the characteristics of the A/D converter. Over these domains, a fuzzy controller has seven linearly distributed triangular fuzzy sets for both inputs, as shown in Figure 3.13. The input values that exceed the limits of the specified input domains belong only to boundary fuzzy sets with the maximum degree of membership. COG defuzzification method was used in all experiments.

3.4.1 Emulation of a PI Controller

In the first experiment we shall test the performance of a control system that contains a simulated process (3.56) and a fuzzy controller whose fuzzy rule table emulates a linear PI controller according to the relation (3.8). Among many possible ways, the synthesis of PI controller parameters may be carried out according to the so-called technical optimum criterion. The idea of this criterion is to compensate the dominant time constant in a control loop with an integral time constant T_{I}, which in our case implies $T_{\mathrm{I}} = 3$ sec. Gain coefficient K_{P} can be determined from Bode plots of open loop frequency characteristics by applying the following useful approximate relation for high-order control systems

$$\gamma[^\circ] + \sigma_{\mathrm{m}}[\%] \approx 63 \tag{3.57}$$

which connects values of phase margin γ expressed in degrees and a percentage of an overshoot in the system response σ_{m}.

In our case, overshoot was set to be $\sigma_{\mathrm{m}} = 5\%$, which yielded phase margin $\gamma = 58^\circ$, and eventually, gain coefficient value $K_{\mathrm{P}} = 2.2$. By insertion of T_{I} and K_{P} in Equation (3.8) for the given control interval $T_{\mathrm{d}} = 200$ msec, we obtained

$$\Delta u = u_{\mathrm{FC}} = \mathrm{A}_q = K_{\mathrm{Pd}} c_j^{\Delta e} + K_{\mathrm{Id}} c_i^{e} = 2.2 c_j^{\Delta e} + 0.1467 c_i^{e} \tag{3.58}$$

The insertion of the centers of input fuzzy sets e_{ci}=$\{-200, –133, -67, 0, 67, 133, 200\}$ and $\Delta e_{cj} = \{–30, –20, –10, 0, 10, 20, 30\}$ into the initial setting algorithm (3.58) yields the values of output singletons appearing in the fuzzy rule table shown in Table 3.1. The values are expressed as the number of least significant bits (LSB) with respect to the full range of a 12-bit D/A converter [−2048, 2047]. These singletons form an initial fuzzy control surface tailored to emulate PI controller function on the defined constrained universe of discourse associated with a selected operating point.

The controller was analyzed in operating point $u_{\mathrm{r}} = 0.6$ V (digital value 2200), with imposed change of reference input $\Delta u_{\mathrm{r}} = 0.88$ V (200 LSB). For the sake of comparison, Figure 3.14 shows fuzzy PI-controlled and standard PI-controlled system responses together with an open-loop system response. We may see that the difference between the responses (mainly in the overshoot of the response) is tolerable.

We may ask what kind of a performance can be expected from a fuzzy-emulated PI controller designed for constrained input domains when input values notably exceed the limits of these domains. This may occur, among other reasons, due to a greater change of reference input (in our case for $u_{\mathrm{r}} > 0.88$ V) or to a greater influence of external disturbance. In that case, input values will belong to boundary fuzzy sets with the maximum degree of membership. From the control point of view, this can be interpreted as the effect of saturation. This will cause slower build-up of the fuzzy-emulated PI controller output in comparison to standard PI controller output, eventually resulting in slower closed-loop system responses.

Table 3.1
A Fuzzy Rule Table Obtained by the Emulation of a PI Controller

	e						
d*e*	**NLE**	**NME**	**NSE**	**ZE**	**PSE**	**PME**	**PLE**
NLDE	−95	−85	−76	−66	−56	−46	−37
NMDE	−73	−64	−54	−44	−34	−24	−15
NSDE	−51	−42	−32	−22	−12	−2	7
ZDE	−29	−20	−10	0	10	20	29
PSDE	−7	2	12	22	32	42	51
PMDE	15	24	34	44	54	64	73
PLDE	37	46	56	66	76	85	95

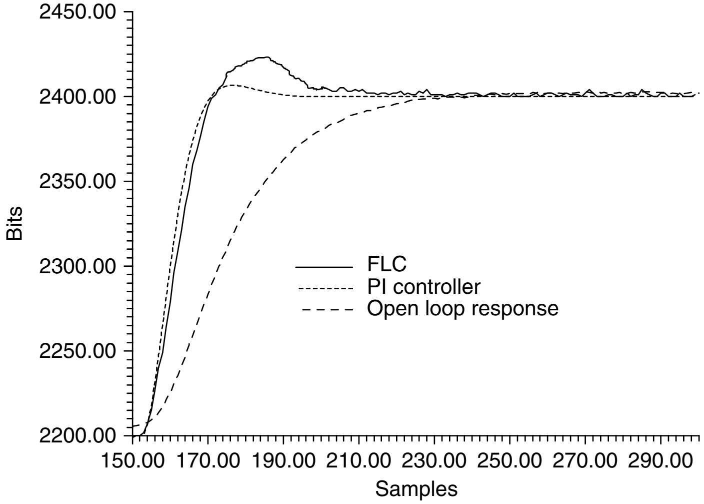


Figure 3.14 Comparison of transient responses in fuzzy emulation of a PI controller. (From Bogdan, S. and Kovačić, Z., *IEEE Conf. Control Appl.*, 648–652, 1998. With permission.)

As far as linear systems are concerned, the solution for the saturation problem can be in the definition of larger input domains. The largest one will be equal to the entire input range of the A/D converter, which in our example is ±9 V or digitally, [−2048, 2047]. Accounting for the linear law of initial setting (3.8), input values (e.g., for $\Delta u_\mathrm{r} = 8.8$ V) that are ten times greater should result in controller output values that are also approximately ten times greater.

In terms of nonlinear systems, a fuzzy-emulated PI controller designed for constrained input domains can satisfy the control quality criterion only if it is in the vicinity of a selected (nominal) operating point. If we wish to use the same fuzzy emulated PI controller on the entire input range of the A/D converter, we must be aware that the performance of such a controller worsens as the system departs from the nominal operating point. The solution may be in splitting the nonlinear system characteristic into a number of linear segments, each of them controlled by a corresponding gain-varying fuzzy-emulated PI controller (designed on the constrained portion of the input domain). This technique is known in control as gain-scheduling.

If we wish to control a nonlinear system on the entire input domain using a single fuzzy controller which should provide uniform control quality, using a fuzzy-emulated PI controller is not the best solution. However, we may use this type of a controller as the first step of heuristic or self-organizing fuzzy controller design, which will be the subject of the chapters that follow.

3.4.2 Model Reference-Based Initial Setting

The initial setting of a fuzzy rule table by using a second-order reference model for the defining a desired closed-loop system dynamics is based on the assumption that a high-order controlled process with measurable input and output can be described in a selected operating point with linear second-order approximation

$$G_{\mathrm{P}}(s) = \frac{Y_{\mathrm{f}}(s)}{U(s)} \approx \frac{K}{T_{\mathrm{A}}^2 \cdot s^2 + 2 \cdot \xi \cdot T_{\mathrm{A}} \cdot s + 1} = G_{\mathrm{A}}(s) \tag{3.59}$$

For the processes with the aperiodic type of response, $G_{\mathrm{A}}(s)$ may assume the form

$$G_{\mathrm{A}}(s) = \frac{K}{(1 + T_1 s)(1 + Ts)} \tag{3.60}$$

In that case, we can use graph-analytical methods, which determine the approximate process transfer function from an open-loop process response like the one shown in Figure 3.15. In doing that, we should make certain that a high-order process with dead time is replaced by a second-order process with dead time.

Suppose that we have a graph of the process transient response in a selected operating point, as the one shown in Figure 3.15. If we draw a tangent in the inflection point, we will be able to determine parameters of the transient response (T_{p}, τ, K, y_{in}, t_{in}) whose values after using diagrams shown in Figure 3.16 yield the parameters of an approximate second-order process transfer function [23].

By applying this method to our third-order process (3.56) we get:

$$G_{\mathrm{A}}(s) = \frac{0.94}{(1 + 3.66s)(1 + 1.46s)}$$

After applying an Euler discretization which substitutes $\mathrm{d}y/\mathrm{d}t$ with $(y(k) - y(k-1))/T_{\mathrm{d}}$ and $\mathrm{d}^2y/\mathrm{d}t^2$ with $(y(k) - 2y(k-1) + y(k-2))/T_{\mathrm{d}}^2$, and

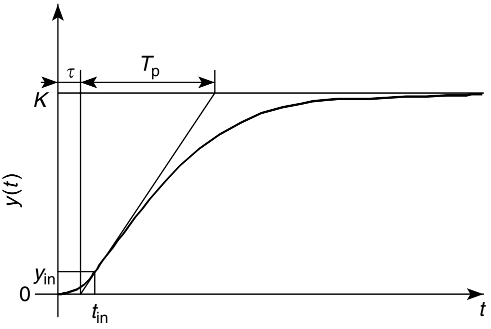


FIGURE 3.15 The determination of parameters of a linear second-order process approximation.

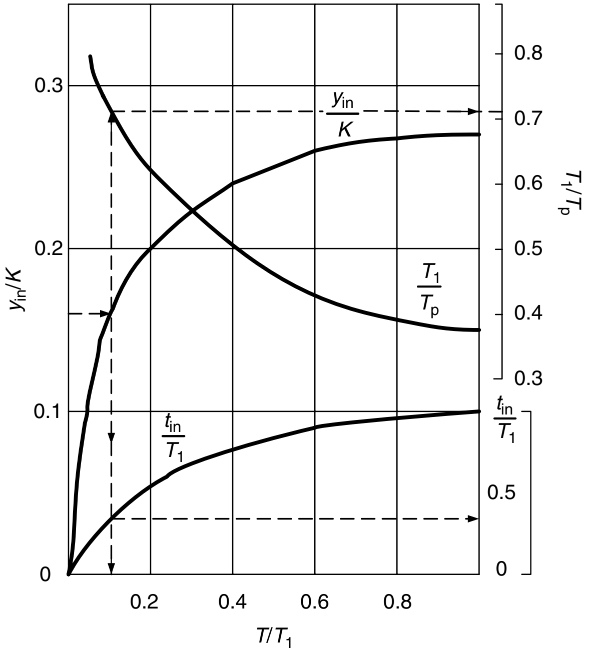


FIGURE 3.16 Diagrams for the determination of parameters of a linear second-order process approximation.

by accounting for the given control interval value $T_d = 200$ msec, we get the discrete form of $G_A(s)$:

$$G_A(z) = \frac{0.00587z}{z^2 - 1.8277z + 0.834} \tag{3.61}$$

The desired closed-loop dynamics is defined with a second-order reference model (3.28). But we must first determine the reference model parameters such as overshoot σ_{m} and peak time t_{m}. In our example, let $\sigma_{\mathrm{m}} = 5\%$, $t_{\mathrm{m}} = 5$ sec. Then from equations

$$\xi = \sqrt{\frac{\ln^2(\sigma_{\mathrm{m}}[\%]/100)}{\pi^2 + \ln^2(\sigma_{\mathrm{m}}[\%]/100)}}, \quad \omega_{\mathrm{n}} = \frac{\pi}{t_{\mathrm{m}}\sqrt{1-\xi^2}} \tag{3.62}$$

we can determine damping coefficient ξ and natural frequency ω_{n} which figure as parameters in the reference model transfer function

$$G_{\mathrm{M}}(s) = \frac{Y_{\mathrm{M}}(s)}{U_{\mathrm{r}}(s)} = \frac{\omega_{\mathrm{n}}^2}{s^2 + 2\xi\omega_{\mathrm{n}}s + \omega_{\mathrm{n}}^2} \tag{3.63}$$

After applying an Euler discretization for $T_{\mathrm{d}} = 200$ msec, we get the discrete form of $G_{\mathrm{M}}(s)$:

$$G_{\mathrm{M}}(z) = \frac{0.0237z}{z^2 - 1.7638z + 0.7875} \tag{3.64}$$

From Equations (3.61) and (3.64) we can read the values of transfer function parameters $G_{\mathrm{A}}(z)$ and $G_{\mathrm{M}}(z)$ denoted in Equation (3.30): $a_{\mathrm{A1}} = 1.8277$, $a_{\mathrm{A2}} = -0.834$, $b_{\mathrm{A1}} = 0.00587$, $a_{\mathrm{M1}} = 1.7638$, $a_{\mathrm{M2}} = -0.7875$, and $b_{\mathrm{M1}} = 0.0237$. These parameters define the coefficients of controller (3.39) as well as the coefficients of controller (3.42): $k_1 = 2.9642$, $k_2 = 7.9216$, and $k_3 = 1.0733$. Since the reference model has a unity gain coefficient, the value of coefficient k_3 is reciprocal to the value of the process gain coefficient. By inserting k_1 and k_2 in Equation (3.47), we obtain

$$u_{\mathrm{FC}} = 2.9642c_i^e + 7.9216c_j^{\Delta e} = \mathrm{A}_q \tag{3.65}$$

The insertion of centers of input fuzzy sets $c_i^e = \{-200, -133, -67, 0, 67, 133, 200\}$ and $c_j^{\Delta e} = \{-30, -20, -10, 0, 10, 20, 30\}$ into the initial setting algorithm (3.65) yields the values of output singletons, which appear in the fuzzy rule table shown in Table 3.2. The values are expressed as the number of LSB with respect to the full range of a 12-bit D/A converter [−2048, 2047]. These singletons form an initially set fuzzy control surface tailored to enforce the desired system dynamics on the defined constrained universe of discourse associated with a selected operating point.

The model reference-based initially set fuzzy controller was analyzed in operating point $u_{\mathrm{r}} = 0.6$ V (digital value 2200), with the imposed change of reference input $\Delta u_{\mathrm{r}} = 0.88$ V (200 LSB). For the sake of comparison, Figure 3.17 shows the reference model and system responses together with an open-loop system response. We may see that the difference between peak values of model and process responses is especially noticeable, which is the result of reducing high-order

TABLE 3.2
The Fuzzy Rule Table Obtained by Model Reference-Based Initial Setting of a Fuzzy Controller

	e						
de	**NLE**	**NME**	**NSE**	**ZE**	**PSE**	**PME**	**PLE**
NLDE	−830	−631	−436	−237	−39	156	355
NMDE	−751	−552	−357	−158	40	235	434
NSDE	−672	−473	−277	−79	119	315	513
ZDE	−592	−394	−198	0	198	394	592
PSDE	−513	−315	−119	79	277	473	672
PMDE	−434	−235	−40	158	357	552	751
PLDE	−355	−156	39	237	436	631	830

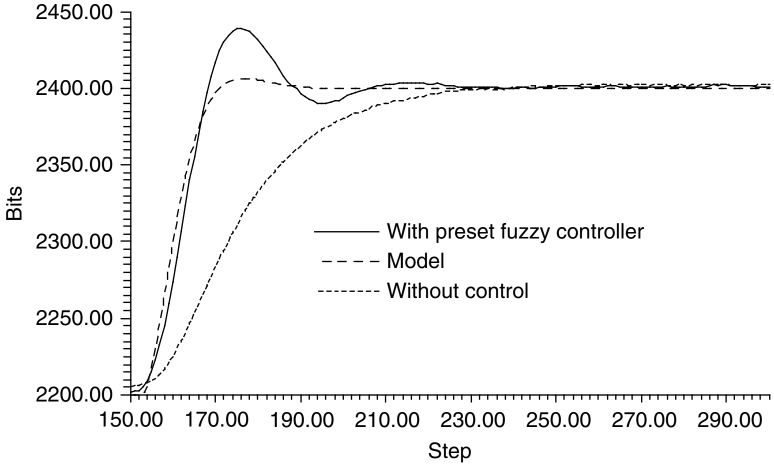


FIGURE 3.17 The comparison of transient responses in model reference-based initial setting of a fuzzy controller. (From Bogdan, S. and Kovačić, Z., *IEEE Conf. Control Appl.*, 648–652, 1998. With permission.)

system dynamics to second-order. The imprecision of the graph-analytical method of process identification may also contribute to the difference.

Despite its imperfection, the method is simple enough to be implemented into almost any hardware. The differences in dynamics can be easily corrected, for example, by automatically changing the output scaling factor k_u or by further heuristic or self-organization-based intervention in the controller.

A fuzzy controller obtained by model reference-based synthesis can be seen as a "linear" controller due to the linear character of the initial setting algorithm

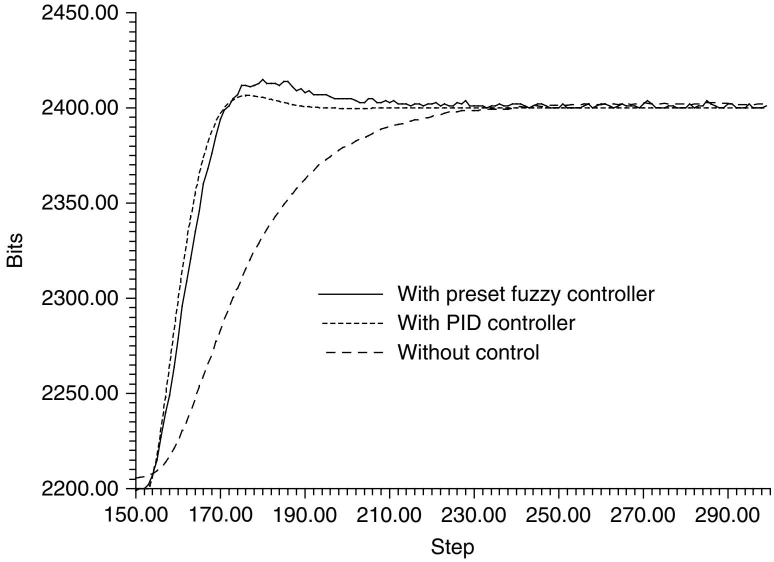


FIGURE 3.18 The comparison of transient responses in phase plane-based initial setting of a fuzzy controller. (From Bogdan, S. and Kovačić, Z., *IEEE Conf. Control Appl.*, 648–652, 1998. With permission.)

(3.47). Therefore, this type of controller is just as appropriate for the control of nonlinear processes as the fuzzy controller obtained by emulating a PI controller.

3.4.3 Phase Plane-Based Initial Setting

Let a third-order process (3.56) be controlled by a PID controller having the following parameter values: $K_R = 3$, $T_I = 3$ sec, and $T_D = 0.2$ sec. From the closed-loop system response on a stepwise change of the reference input shown in Figure 3.18 and from the PID controller output response, we can generate a series of triples [$\Delta u(k)$, $e(k)$, and $de(k)$] representing the discrete record of a phase trajectory and the corresponding control curve lying above it. This phase trajectory will serve as a basis for the determination of output singletons in the fuzzy rule table by using relations (3.52) to (3.55). Output singleton values shown in Table 3.3 are expressed as the number of LSB with respect to the full range of a 12-bit D/A converter [−2048, 2047]. By comparing the PID controlled system response and the response of a phase plane-based initially set fuzzy controller, we may see that there is apparent resemblance between them (only a small difference may be noticed in their peak values).

As expected, the singleton values in Tables 3.1 and 3.3 that represent the increments of controller output $\Delta u_{FC}(k)$ indicate that the two fuzzy rule tables are different. The results obtained clearly show the practical value of all these methods.

TABLE 3.3
The Fuzzy Rule Table Obtained by Phase Plane-Based Setting of a Fuzzy Controller

	e						
d*e*	**NLE**	**NME**	**NSE**	**ZE**	**PSE**	**PME**	**PLE**
NLDE	−330	−300	−230	90	110	150	200
NMDE	−360	−260	−200	70	130	184	300
NSDE	−355	−230	−170	20	140	200	340
ZDE	−350	−210	−150	0	150	210	350
PSDE	−340	−200	−140	−20	170	230	355
PMDE	−300	−184	−130	−70	200	260	360
PLDE	−200	−150	−110	−90	230	300	330

For a given process, reference model and a PI(D) controller, phase plane-based initial setting seemed to give the best results. In general, it is difficult to say which method is better. This may depend on the type of the process, the operating point, the accuracy of process approximation, the number of input fuzzy sets, etc. The choice of the initial setting method is completely up to the designer, but any of the methods presented here will lead to an operative fuzzy controller in less time and design effort than when using heuristic trial-and-error procedure.

REFERENCES

1. Guely, F. and Siarry, P., "A centered formulation of Takagi–Sugeno rules for improved learning efficiency," *Fuzzy Sets and Systems*, 62, 277–285, 1994.
2. Ishibuchi, H., Noyaki, K., Tanaka, H., Hosaka, Y., and Matsuda, M., "Empirical study on learning in fuzzy systems by rice taste analysis," *Fuzzy Sets and Systems*, 64, 129–144, 1994.
3. Gürocak, H.B. and de San Lazaro, A., "A fine tuning method for fuzzy rule bases," *Fuzzy Sets and Systems*, 67, 147–161, 1994.
4. Vuorimaa, P., "Fuzzy self-organizing map," *Fuzzy Sets and Systems*, 66, 223–231, 1994.
5. Bandyopadhyay, R. and Patranabis, D., "A new autotuning algorithm for PID controllers using dead-beat format," *ISA Transactions*, 40, 255–266, 2001.
6. Lu, J., Chen, G., and Ying, H., "Predictive fuzzy PID control: theory, design and simulation," *Information Sciences*, 137, 157–187, 2001.
7. Precup, R.-E., Preitl, S., and Faur, G., "PI predictive fuzzy controllers for electrical drive speed control: methods and software for stable development," *Computers in Industry*, 52, 253–270, 2003.
8. Ying, H., Siler, W., and Buckley, J., "Fuzzy control theory: a nonlinear case," *Automatica*, 26, 513–520, 1990.
9. Malki, H., Li, H., and Chen, G., "New design and stability analysis of a fuzzy proportional-derivative control system," *IEEE Transactions on Fuzzy Systems*, 2, 245–254, 1995.

10. Misir, D., Malki, H., and Chen, G., "Design and analysis of a fuzzy proportional–integral–derivative controller," *International Journal of Fuzzy Sets and Systems*, 79, 297–314, 1996.
11. Malki, H., Feigenspan, D., Misir, D., and Chen, G., "Fuzzy PID control of a flexible-joint robot arm with uncertainties from time-varying loads," *IEEE Transactions on Control Systems Technology*, 5, 371–378, 1997.
12. Sooraksa, P. and Chen, G., "Mathematical modeling and fuzzy control for flexible link robots," *Mathematical and Computer Modeling*, 27, 73–93, 1998.
13. Hu, B., Mann, G.K.I., and Gosine, R.G., "New methodology for analytical and optimal design of fuzzy PID controllers," *IEEE Transactions on Fuzzy Systems*, 7, 521–539 1999.
14. Carvajal, J., Chen, G., and Ogmen, H., "Fuzzy PID controller: design, performance evaluation, and stability analysis," *Information Sciences*, 123, 249–270, 2000.
15. Li, H.X. and Gatland, H.B., "A new methodology for designing a fuzzy logic controller," *IEEE Transactions on Systems, Man and Cybernetics*, 25, 505–512, 1996.
16. Ying, H., "The simplest fuzzy controllers using different inference methods are different nonlinear proportional–integral controllers with variables gains," *Automatica*, 29, 1579–1589, 1993.
17. Ding, Y., Ying, H., and Shao, S., "Typical Takagi–Sugeno PI and PD fuzzy controllers: analytical structures and stability analysis," *Information Sciences*, 151, 245–262, 2003.
18. Patel, A.V. and Mohan, B.M., "Analytical structures and analysis of the simplest fuzzy PI controllers," *Automatica*, 38, 981–993, 2002.
19. Kukolj, D.D., Kuzmanović, S.B., and Levi Emil, "Design of a PID-like compound fuzzy logic controller," *Engineering Applications of Artificial Intelligence*, 14, 785–803, 2001.
20. Desoer, C.A. and Vidyasagar, M., *Feedback Systems: Input–Output Properties*, Academic Press, New York, 1975.
21. Astrom, K.J. and Wittenmark, B., *Computer Controlled Systems*, Prentice Hall, Englewood Cliffs, NJ, 1984.
22. Bogdan, S. and Kovačić, Z., "Methods for automated design of a singleton fuzzy logic controller," *Proceedings of the 1998 IEEE Conference on Control Applications*, Trieste, pp. 648–652, 1998.
23. Netushil, A., *Theory of Automatic Control*, Mir Publishers, Moscow, 1978.

4 Complex Fuzzy Controller Structures

It happens very often in practice that a designed controller works satisfactorily in one operating regime, but not in the other. For another type of controller it could be quite contrary. For example, the PD-type controllers usually cannot maintain the steady-state accuracy if they control a static control process, while the PI-type controllers can do that very well. Having that in mind, new, hybrid types of controllers can be designed. By combining different types of controllers into more complex structures, a design objective is to join good control characteristics of each controller into overall characteristics of a hybrid controller.

In the previous chapters, we have shown that the design of fuzzy controllers can end up with a versatility of control functions (i.e., control surfaces in the case of double input–single output [DISO] fuzzy controllers). It must be noted that such controllers do not have some *a priori* recognized inherent features (like robustness, for example), as these features primarily depend on a design procedure carried out.

Fuzzy controllers can be easily combined with traditional controllers thus making various combinations of complex or so-called hybrid fuzzy control schemes [1–4]. One possible way to go is to combine fuzzy and linear (P, PI, PID) controllers to work in the complementary or parallel regime (Figure 4.1). The usage of a fuzzy controller in parallel with a PI controller has been adopted as a standard industry solution (e.g., in Reference 5). By adding a nonlinear component to the existing PI controller, a new controller can cope much better with process nonlinearities in a certain range around the operating point. In this chapter, we go a little bit further and describe a design of a multimode hybrid controller where parallel work of PI and fuzzy controllers is just one of operating modes. We show that in a selected servo control application, design of such a controller can ensure high robustness to moderate parameter variations and fair robustness to large parameter variations.

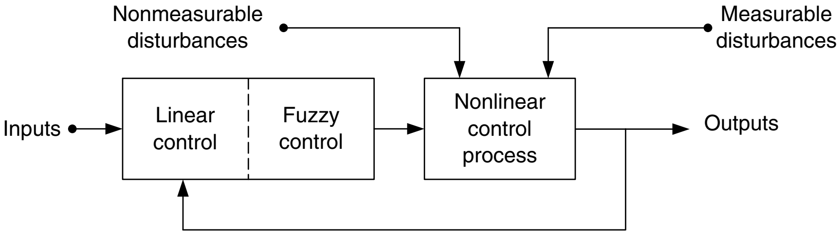


FIGURE 4.1 Combined linear + fuzzy control of nonlinear control processes.

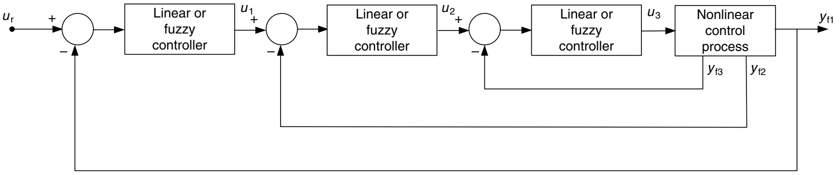

u_r
+
−
Linear or fuzzy controller
u_1
+
−
Linear or fuzzy controller
u_2
+
−
Linear or fuzzy controller
u_3
Nonlinear control process
y_{f1}
y_{f3}
y_{f2}

FIGURE 4.2 Combined linear + fuzzy control of cascade control systems.

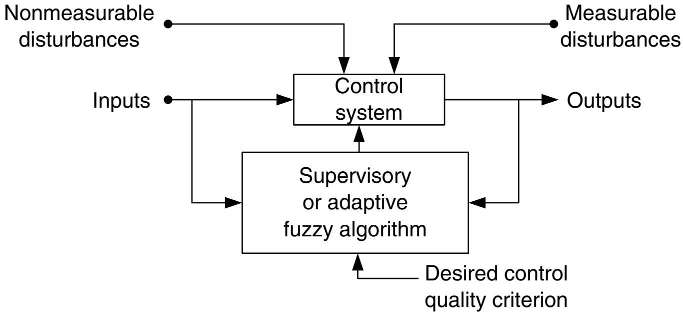

Nonmeasurable disturbances
Measurable disturbances
Inputs
Control system
Outputs
Supervisory or adaptive fuzzy algorithm
Desired control quality criterion

FIGURE 4.3 Supervisory or adaptive complex fuzzy control systems.

Thanks to the variety of control problems, fuzzy controllers can form more complex control schemes, such as modal control systems (control of process state variables) or cascade control systems (control of process variables) shown in Figure 4.2. The basic structure of the fuzzy controller can also be used for implementation of a supervisory algorithm in supervisory control schemes or an adaptation algorithm in adaptive control schemes (Figure 4.3). The most frequently used adaptation technique is gain-scheduling, where, depending on the operating point, gain coefficients of conventional controllers are changed according to the designed nonlinear fuzzy mapping function(s) [6–8]. In that case, the fuzzy algorithm acts on the control loop as an external control element.

In the complex fuzzy control systems, we can count all forms of fuzzy controllers combined with conventional nonlinear controllers, as well as all combinations of fuzzy control algorithms and other intelligent control techniques. Neural Networks (NN) and Genetic Algorithms (GA) are most often used to enhance the control characteristics of the fuzzy controller. NN-fuzzy and GA-fuzzy control algorithms have proved to be effective in many practical applications [9–15]. In this chapter, we shall focus on some hybrid and adaptive fuzzy control structures, which are, due to their simplicity and effectiveness, attractive for implementation in industrial control systems.

4.1 Hybrid Fuzzy Control

This chapter covers the design of fuzzy control schemes that contain, besides a fuzzy controller, other control elements known from the classical control practice.

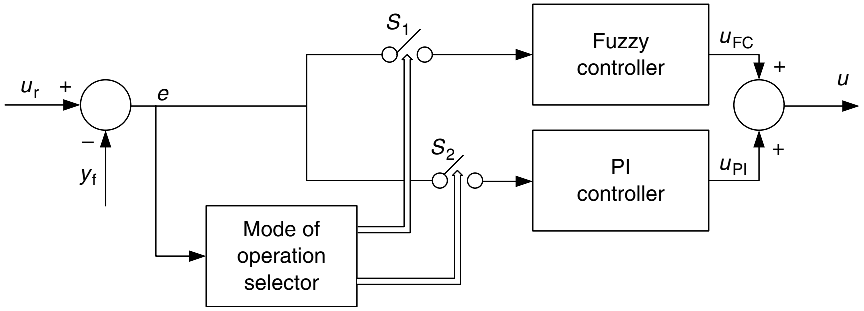


FIGURE 4.4 Structure of the hybrid fuzzy controller. (From Kovačić, Z. and Bogdan, S., *Eng. Appl. Artif. Intelligence*, 7(5), 501–511, 1994. With permission from Elsevier.)

We discuss typical problems that may occur in cases of a parallel and multimode operation, such as the chattering problem and the problem of providing bumpless transitions among controller operating modes.

In order to get control schemes that would be less sensitive to parameter variations than traditional linear PID controllers, let us analyze the hybrid controller structure shown in Figure 4.4. As can be seen, it is a controller that contains a PD-type fuzzy and a linear PI control algorithm. It has a single input, error signal $e(k)$, which internally yields another fuzzy controller input, change in error signal $\Delta e(k)$.

This controller is meant as a multimode controller, which has three modes of operation dictated by the mode of operation selector (Figure 4.4). The change of modes depends on the magnitudes of fuzzy controller inputs according to the following set of relations:

$$\begin{array}{llllll} 1. & e(k) \in \mathrm{ZE} & \text{and} & \Delta e(k) \in \mathrm{ZDE} \Rightarrow S_1 = \mathrm{OFF}, & S_2 = \mathrm{ON} & \\ 2. & e(k) \notin \mathrm{ZE} & \text{and} & \Delta e(k) \in \mathrm{ZDE} \Rightarrow S_1 = \mathrm{ON}, & S_2 = \mathrm{ON} & (4.1) \\ 3. & & & \Delta e(k) \notin \mathrm{ZDE} \Rightarrow S_1 = \mathrm{ON}, & S_2 = \mathrm{OFF} & \end{array}$$

where ZE and ZDE are zero fuzzy subsets of the fuzzy controller inputs.

The fuzzy control algorithm, activated when switch $S_1 =$ "ON," acts in the case of sufficiently large reference input changes, while the PI control algorithm, activated by switch S_2, mainly supports steady-state accuracy and cancels disturbance effects. Both controllers operate together in the case of moderate control error values (usually due to the impact of disturbances). Therefore, PI controller parameters K_{PI} and T_{PI} may be specified, for example, according to the symmetrical optimum criterion to ensure optimal compensation of disturbance effects [16].

In a variant of the discussed hybrid fuzzy controller, a PI-type fuzzy controller can take the role of the PI controller. Then the PD-type fuzzy controller would act only in the case of sufficiently large reference input changes, while the PI-type fuzzy controller would overtake the control in other cases. In this way, there would be only two basic operating modes.

Regarding the software implementation, special attention must be paid to the switching of operating modes, as the hybrid fuzzy controller contains two control algorithms, which may work either separately or together. In order to avoid the chattering problem of two control algorithms, switching from the integral (PI or PI-fuzzy) to the nonintegral (PD-fuzzy) mode of operation should be made in such a way that the controller output value in the previous mode becomes the initial value for the controller output in the current mode.

Example 4.1 Design of a hybrid fuzzy controller.

Let us demonstrate the effectiveness of a hybrid fuzzy controller design in the case of controlling the angular speed of a permanent magnet synchronous motor (PMSM) drive. The performance of the controller is tested and validated by simulations in MATLAB® +Simulink. The implementation of MATLAB+Simulink simulation models of the hybrid fuzzy controller and the PMSM drive are described in more detail in Section 6.2.

The vector-controlled PMSM drive considered for hybrid fuzzy control is a nonstationary high-order control system. As it is known from the theory of electrical machines control, the goal of vector control is to keep the control characteristics of the PMSM as close as possible to the control characteristics of a DC motor. Instead in the three-phase (R, S, T) coordinate frame, vector control is performed in two-dimensional d–q coordinate frame, where d denotes the direct axis, and q the quadrature axis, respectively. A description of the PMSM in d–q coordinates is obtained by using R–S–T to d–q Park's transformation [18], which holds equally for the phase voltages, currents, and flux linkages. During constant flux operation, the air–gap flux linkage lies in d-axis, while the q-axis stator current (the torque producing current) is maintained at 90° to the air–gap flux.

In case of constant flux $(\mathrm{d}\Psi_\mathrm{m}/\mathrm{d}t = 0)$, the PMSM is fully described with the following state space equations [19]

$$\begin{aligned}
\frac{\mathrm{d}i_\mathrm{d}}{\mathrm{d}t} &= \frac{1}{L_\mathrm{d}}(u_\mathrm{d} - Ri_\mathrm{d} + \omega_\mathrm{r}L_\mathrm{q}i_\mathrm{q}) \\
\frac{\mathrm{d}i_\mathrm{q}}{\mathrm{d}t} &= \frac{1}{L_\mathrm{q}}(u_\mathrm{q} - Ri_\mathrm{q} - \omega_\mathrm{r}L_\mathrm{d}i_\mathrm{d} - \omega_\mathrm{r}\Psi_\mathrm{m}) \\
\frac{\mathrm{d}\omega_\mathrm{r}}{\mathrm{d}t} &= \frac{1}{J}(p_\mathrm{m}\tau_\mathrm{e} - p_\mathrm{m}\tau_\mathrm{l} - B\omega_\mathrm{r}) \\
\frac{\mathrm{d}\theta_\mathrm{r}}{\mathrm{d}t} &= \omega_\mathrm{r}
\end{aligned} \tag{4.2}$$

where $u_\mathrm{d}, u_\mathrm{q}$ are the d- and q-axis stator voltages [V]; $i_\mathrm{d}, i_\mathrm{q}$, the d- and q-axis stator currents [A]; R, the stator resistance [Ω]; $L_\mathrm{d}, L_\mathrm{q}$, the stator d and q inductances [H]; p_m, the number of pole pairs; J, the moment of inertia [kg m^2]; B, the coefficient of viscous friction [Nmsec]; τ_e, the electric torque [Nm]; and τ_l is the load torque [Nm].

The PMSM is producing a torque described as

$$\tau_{\mathrm{e}} = \tfrac{3}{2} p_{\mathrm{m}} [i_{\mathrm{q}} \Psi_{\mathrm{m}} + (L_{\mathrm{d}} - L_{\mathrm{q}}) i_{\mathrm{d}} i_{\mathrm{q}}]. \tag{4.3}$$

In case of constant flux ($i_{\mathrm{d}} = 0$), torque equation (4.3) attains the form

$$\tau_{\mathrm{e}} = K i_{\mathrm{q}} \tag{4.4}$$

where $K = 3 p_{\mathrm{m}} \Psi_{\mathrm{m}}/2$.

In case of constant flux, torque τ_{e} is proportional to the q-axis stator current, that is, Equation (4.4) attains a form similar to the torque equation of a DC motor. The PMSM drive considered for servo applications contains a PI controller in the outer angular speed control loop, and a ramp comparison controller (PWM) in the inner stator current control loop. Chopper switching frequencies have typical values of 5 to 20 kHz, thus providing almost instantaneous current control. Therefore, a closed-current control loop may be approximated by the following transfer function

$$G_{\mathrm{cc}}(s) = \frac{I_q(s)}{U(s)} = K_{\mathrm{cc}}\, \mathrm{e}^{-T_{\mathrm{cc}} s} \approx \frac{K_{\mathrm{cc}}}{1 + T_{\mathrm{cc}} s} \tag{4.5}$$

where $T_{\mathrm{cc}} = 1/f_{\mathrm{ch}}$ [sec], andf_{ch} is the switching frequency [Hz].

Let us now transform Equations (4.2) and (4.4) by using the Laplace transformation to get the following transfer functions

$$\Omega(s) = \frac{K_{\mathrm{M}}}{1 + T_{\mathrm{M}} s} [T_{\mathrm{e}}(s) - T_{\mathrm{l}}(s)] \tag{4.6}$$

$$T_{\mathrm{e}}(s) = K I_{\mathrm{q}}(s) \tag{4.7}$$

where $K_{\mathrm{M}} = 1/B$, $T_{\mathrm{M}} = J_{\mathrm{T}}/B$.

If combined, transfer functions (4.5–4.7) yield the plant transfer function

$$G_{\mathrm{p}}(s) = \frac{\Omega(s)}{U(s)} = \frac{K_{\mathrm{cc}}}{1 + T_{\mathrm{cc}} s} \frac{K_{\mathrm{M}}}{1 + T_{\mathrm{M}} s} \tag{4.8}$$

The angular speed is measured with a tachogenerator. In the case of filter time constant $T_\omega \approx 0$, the angular speed feedback transfer function may be described approximately by

$$G_\omega(s) = \frac{U_\omega(s)}{\Omega(s)} = \frac{K_\omega}{1 + T_\omega s} \tag{4.9}$$

The plant transfer function including a feedback path assumes the following form

$$G_{\mathrm{s}}(s) = \frac{U_{\omega}(s)}{U(s)} = \frac{K_{\mathrm{cc}}KK_{\mathrm{M}}K_{\omega}}{(1+T_{\mathrm{cc}}s)(1+T_{\mathrm{M}}s)(1+T_{\omega}s)} \tag{4.10}$$

Since the switching frequency f_{ch} is rather high, the time constant T_{cc} is regularly much smaller than other time constants of the controlled system and therefore it can be neglected in analysis. In this case, the transfer function (4.10) assumes the following form

$$G_{\mathrm{s}}(s) = \frac{U_{\omega}(s)}{U(s)} = \frac{K_{\mathrm{cc}}KK_{\mathrm{M}}K_{\omega}}{(1+T_{\mathrm{M}}s)(1+T_{\omega}s)} \tag{4.11}$$

The transfer function of a PI controller has the form

$$G_{\mathrm{PI}}(s) = \frac{U(s)}{U_{\mathrm{rA}}(s)} = K_{\mathrm{PI}}\frac{1+T_{\mathrm{PI}}s}{T_{\mathrm{PI}}s} \tag{4.12}$$

The PI controller parameters K_{PI} and T_{PI} may be specified according to the technical or symmetrical optimum criteria [16], that is, the controller may be adjusted to react optimally to the changes of the reference input u_{r} or the disturbance (load torque τ_{ℓ}), respectively. If the PI controller parameters were defined according to the symmetrical optimum criterion, then the transfer function of the closed-loop angular speed control system would assume the following form

$$G_{\mathrm{cs}}(s) = \frac{U_{\omega}(s)}{U_{\mathrm{r}}(s)} = \frac{K_{\mathrm{o}}(1+T_{\mathrm{PI}}s)}{s(1+T_{\mathrm{M}}s)(1+T_{\omega}s)+K_{\mathrm{o}}(1+T_{\mathrm{PI}}s)} \tag{4.13}$$

In this case, the studied system would have the transfer function of a third-order system. A linearized model of the angular speed control system is shown as a part of the whole fuzzy hybrid control system in Figure 4.5. The parameters that vary most in the system are the torque coefficient, K and the moment of inertia, J_{T}. Sometimes it may also be difficult to measure the exact value of the viscous friction coefficient B. The torque coefficient K changes due to the weakening of magnetic

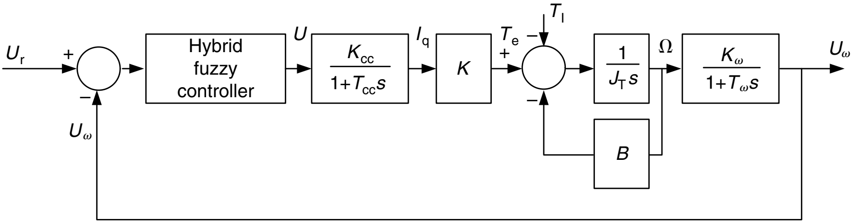


FIGURE 4.5 A block scheme of the studied hybrid fuzzy control system.

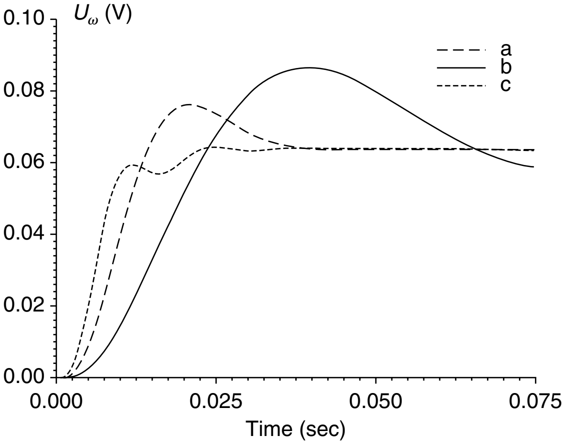


FIGURE 4.6 The measured angular speed responses of the PMSM drive controlled with a PI controller in the case of: J_T (a), $3J_T$ (b), $J_T/3$ (c). (From Kovačić, Z. and Bogdan, S., *KoREMA, Automatika*, 34(3–4), 99–102, 1993. With permission.)

flux in the constant power mode of operation used from the rated to the maximum angular speed [20]. PMSM servo drives are widely used in robots, where the moment of inertia is expected to change in value in the range of 1:10 [21].

The rated values of linearized model parameters (Figure 4.5) were as follows: $K_{cc} = 1$ A/V, $T_{cc} = 50\ \mu$sec, $K = 0.9837$ Vsec, $J_T = 0.00176$ kg m^2, $B = 0.000388$ Nmsec, $T_M = J_T/B = 4.536$ sec, $K_\omega = 0.063$ Vsec, and $T_\omega = 2.5$ msec.

Parameters of the PI controller are synthesized for a selected operating point and their values, $K_{PI} = 6$ V/V, $T_{PI} = 0.013$ sec, are associated with the rated parameter values of the system. The PI controller parameters were specified according to the symmetrical optimum criterion, which gives 40% of overshoot in the output response, all in order to obtain quick compensation of load torque variations. A lower overshoot (in our case 20%) in response to the reference input changes was achieved by adding an appropriate low-pass filter into the reference input signal path.

The robustness of PI controllers to parameter variations is rather weak, especially in cases of large parameter variations. Figure 4.6 shows the measured angular speed responses obtained for large moment of inertia variations, $J_T/3$ and $3J_T$. The change of the dynamic behavior is more than obvious, indicating that PI control should be replaced with a more effective control.

The fuzzy control algorithm belongs to the group of nonlinear PD-type control algorithms. Seven linguistic subsets have been defined for both inputs: NL, NM, NS, Z, PS, PM, and PL. Based on knowledge about the characteristics of the angular speed control loop, the maximum values for both inputs and the output of the fuzzy angular speed controller can be estimated. It is assumed that the maximum change of reference input for the system given in Figure 4.5 is $\Delta u_{rm} = K_\omega \Delta\omega$, where $\Delta\omega$ is taken to be 1 rad/sec, then $-0.063 \leq e(k) \leq 0.063$. The maximum change of

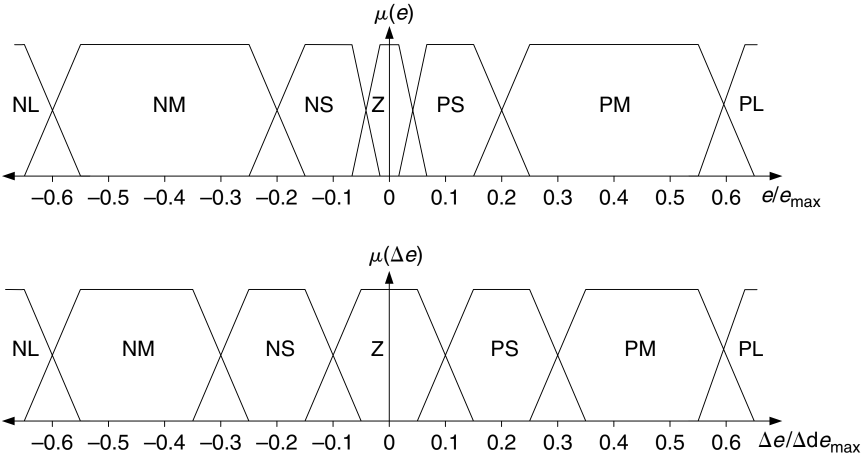


FIGURE 4.7 Input membership functions of a hybrid fuzzy controller: (up) $\mu[e(k)]$, (down) $\mu[\Delta e(k)]$.

error signal during the sampling interval $T_d = 0.5$ msec is estimated to be 0.015, that is, $-0.015 \leq \Delta e(k) \leq 0.015$. The distribution of membership functions related to normalized e and Δe subsets is shown in Figure 4.7. Different forms of membership functions can be used, but experiments have proved that trapezoidal forms contribute the most to achieving lower sensitivity to parameter variations in the designed fuzzy controller.

The universe of discourse of the fuzzy controller output is discrete and contains 15 uniform fuzzy subsets. The distribution of accompanying membership functions is symmetrical and slightly nonlinear because of one extra subset added next to the zero subset to ensure a smooth change of operating modes defined by relations (4.1). The corresponding centroids have the following values: −1, −0.667, −0.5, −0.333, −0.167, −0.0083, −0.0042, 0, 0.0042, 0.0083, 0.167, 0.333, 0.5, 0.667, and 1. It must be noted that these values were normalized with respect to the maximum controller output value $\max(u_{FC}) = 0.96$ V. Because of simplicity and good interpolative features, fuzzy controller output $u_{FC}(k)$ is computed according to the center of gravity principle (2.22).

Experience with the target system helps in the creation of fuzzy control rules, which are organized in the fuzzy rule table. The goal of such design was to provide a 5% overshoot in the system response and to get the peak time close to 15 msec. In the case of defuzzification according to the center of gravity principle, the controller output fuzzy subsets, which are normally found in the fuzzy rule table, can be substituted by their centroids (singletons). The fuzzy rule table of the hybrid fuzzy controller is shown in Table 4.1.

Figure 4.8 and Figure 4.9 show the measured angular speed and hybrid fuzzy controller output responses in the case of stepwise change of the reference input. Figure 4.10 shows the time flow of controller output signals following the sequence

TABLE 4.1
The Fuzzy Rule Table of a Hybrid Fuzzy Controller

	NLE	NME	NSE	ZE	PSE	PME	PLE
NLDE	1	1	0.667	0.5	0.333	0.0083	0
NMDE	1	1	0.667	0.167	0.0083	−0.0042	−0.0083
NSDE	1	0.667	0.167			−0.333	−0.5
ZDE	0.833	0.333	0.0083	0	0	−0.333	−0.833
PSDE	0.667	0.333	0	−0.0083	−0.167	−0.667	−1
PMDE	0.0083	−0.0083	−0.167	−0.333	−0.667	−1	−1
PLDE	−0.0083	−0.167	−0.333	−0.5	−0.667	−1	−1

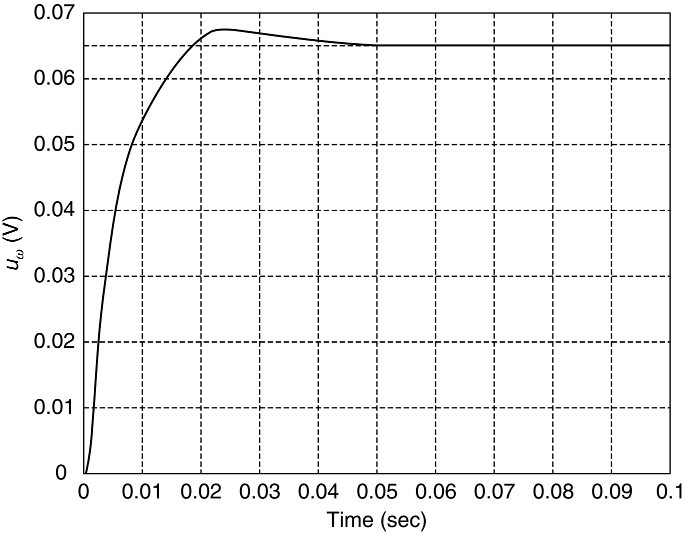


FIGURE 4.8 The measured angular speed response of a hybrid fuzzy control system.

of induced operating mode changes. According to expectations, at the beginning of transient response only the fuzzy controller is active, then fuzzy and PI controllers work together, and eventually, the PI controller takes over control in the steady state. The analysis of responses proves that transitions from one operating mode to another are smooth, without abrupt changes in the controller output signal.

The designed hybrid fuzzy controller was tested by simulation experiments in the cases of moderate and very large changes of the moment of inertia J_T. As shown in Figure 4.11, for moderate changes of moment of inertia equal to $\pm 50\%$ of the rated value (i.e., $J_\mathrm{T}/2$ and $3J_\mathrm{T}/2$), the measured angular speed responses are almost unaffected by the simulated parameter variations. In cases of very large changes of moment of inertia ($J_\mathrm{T}/3, 3J_\mathrm{T}$), as shown in Figure 4.12, the overshoot

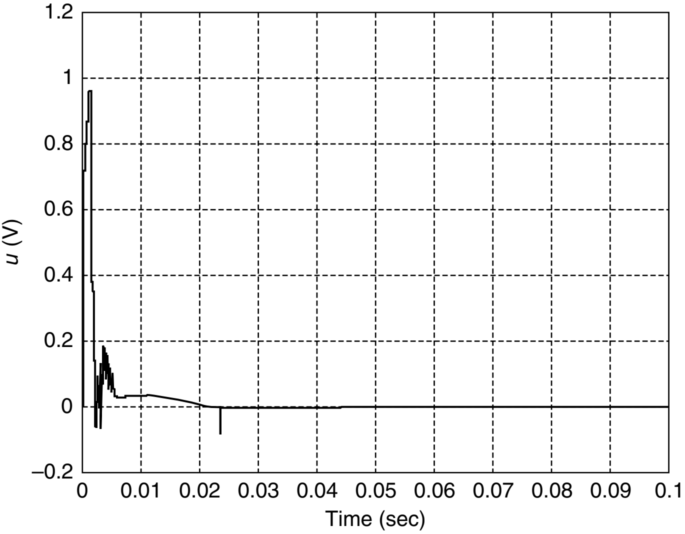


FIGURE 4.9 The hybrid fuzzy controller output response.

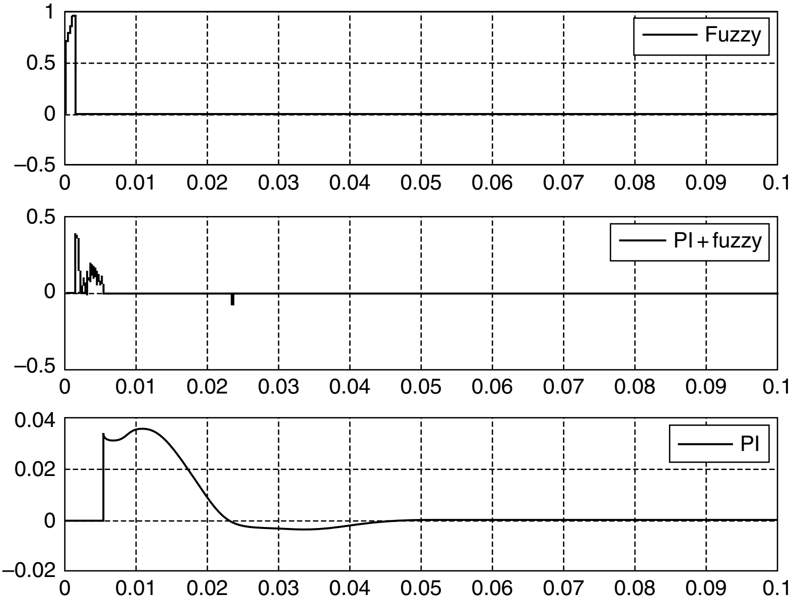


FIGURE 4.10 The hybrid fuzzy controller output response during the change of operating modes.

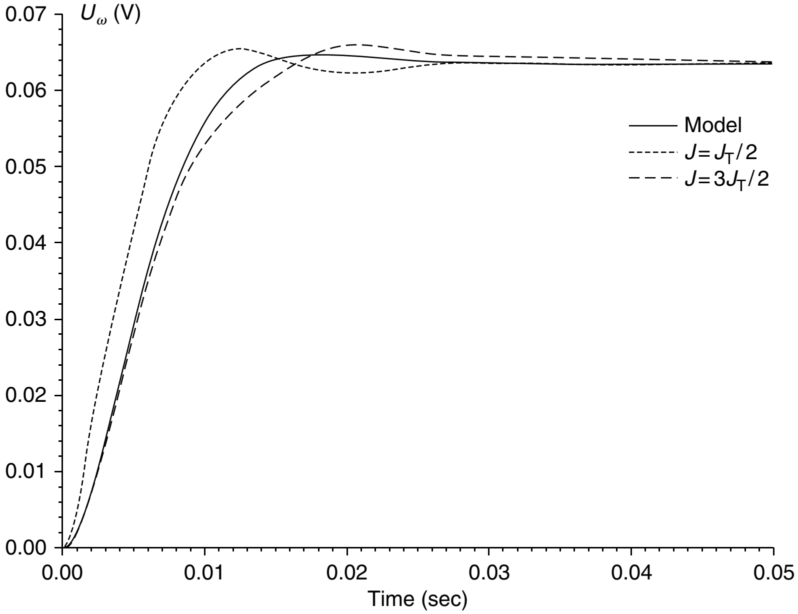


FIGURE 4.11 The measured angular speed responses of the fuzzy control system for moderate changes of J_T. (From Kovačić, Z. and Bogdan, S., *Eng. Appl. Artif. Intelligence*, 7(5), 501–511, 1994. With permission from Elsevier.)

has remained almost constant, while the rise times have changed noticeably. The comparison with the PI-controlled system responses (Figure 4.6) shows significant increase of robustness due to usage of the hybrid fuzzy controller.

If we look at amplitude and phase frequency characteristics of such a system, the ability to keep an almost constant overshoot can be interpreted as an achievement of a wide region of constant phase margin. This suggests that the change of controller gain (i.e., output scaling factor K_u) could be sufficient to compensate for noticed changes of system dynamics.

Figure 4.13 and Figure 4.14 show the responses of the hybrid fuzzy controller output. As can be seen, they are smoothly generated by the sequence of controller's operating modes.

4.2 Adaptive Fuzzy Control

Adaptive control has an important role in modern control systems. During operation, many controlled processes experience abrupt or continuous parameter variations, varying external conditions and, in some occasions, alternations of operating modes. For example, continuous changes of inertial moments and gravity-dependent loads are affecting robot joint servo control loops. Control of mass flow in plastic extruders must deal with a gradual change of material density, temperature, and viscosity as well as varying homogeneity of the material

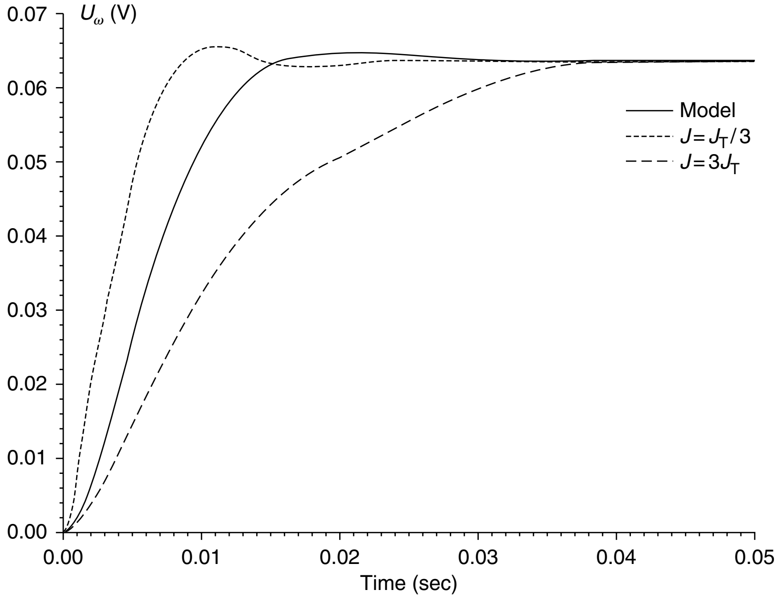


FIGURE 4.12 The measured angular speed responses of the fuzzy control system for large changes of J_T. (From Kovačić, Z. and Bogdan, S., *Eng. Appl. Artif. Intelligence*, 7(5), 501–511, 1994. With permission from Elsevier.)

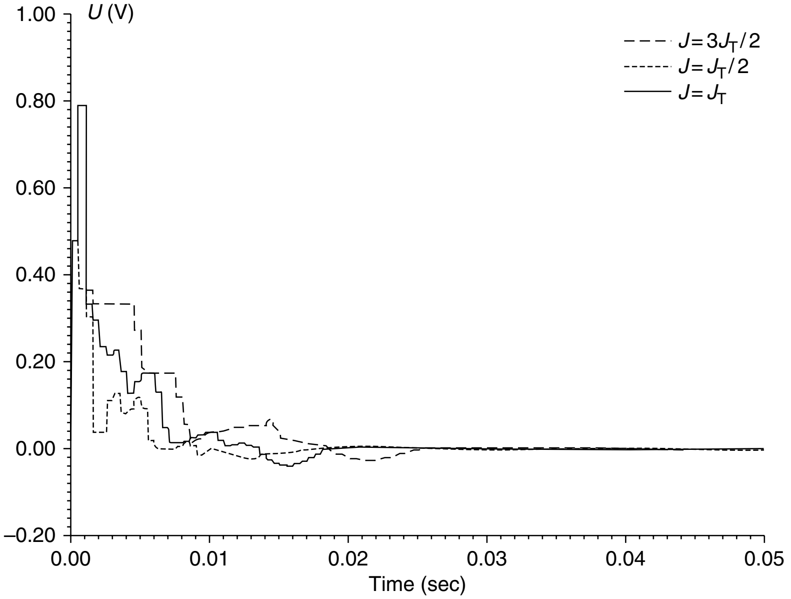


FIGURE 4.13 The hybrid fuzzy controller responses for moderate changes of J_T. (From Kovačić, Z. and Bogdan, S., *Eng. Appl. Artif. Intelligence*, 7(5), 501–511, 1994. With permission from Elsevier.)

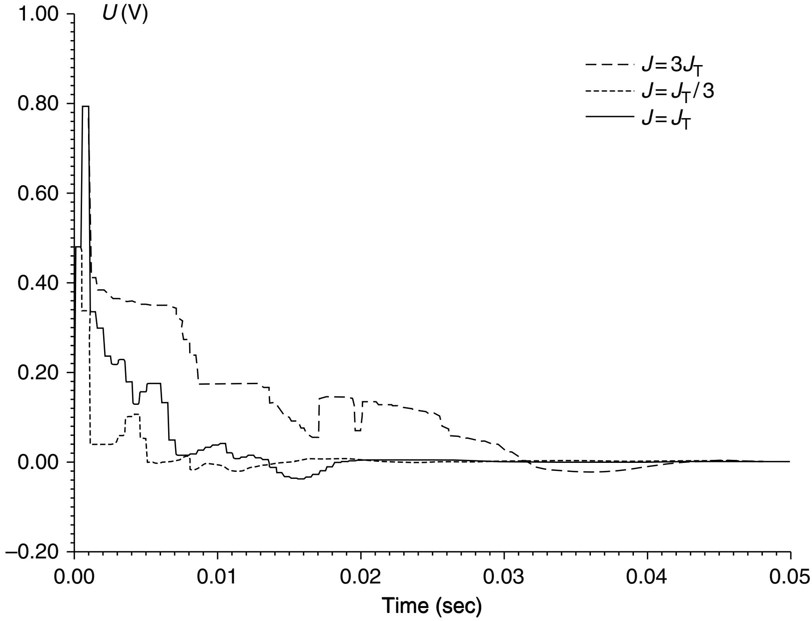


FIGURE 4.14 The hybrid fuzzy controller responses for large changes of J_T. (From Kovačić, Z. and Bogdan, S., *Eng. Appl. Artif. Intelligence*, 7(5), 501–511, 1994. With permission from Elsevier.)

at machine cross-sections starting from the input to the output. We should not forget to mention the first big success of adaptive control, which was achieved in control of guided missiles, rockets, and space ships with mass-varying caused by fuel consumption and rejection of body parts.

When all aforementioned causes of nonuniform system behavior are not excessive, then they are usually well handled with standard feedback controllers. But, when that is not the case, then standard feedback controllers cannot maintain the desired control quality and some sort of adaptation to the new situation in the process is needed. Adaptation may be used for the purpose of improving system dynamics or to reduce system sensitivity to parameter variations.

Since the year 1951 when Draper and Li [22] introduced an adaptive control system, which searched for the optimal operating point of an internal combustion engine, different adaptive control methods have been developed [23–26], some of them focused on adapting parameters (so-called parameter adaptation) and some of them on adapting signals (so-called signal adaptation), and most of them are represented with a general structure shown in Figure 4.15. In order to maintain a uniform dynamic performance of the control system, we need to set a control goal in the form of a desired control quality criterion. Then we need to gather information from the control system, for assessment of the achieved control quality. Based on a difference between them, a decision about necessary changes, either in feedback controller parameters or in adaptation signal is made. An adaptation law contained

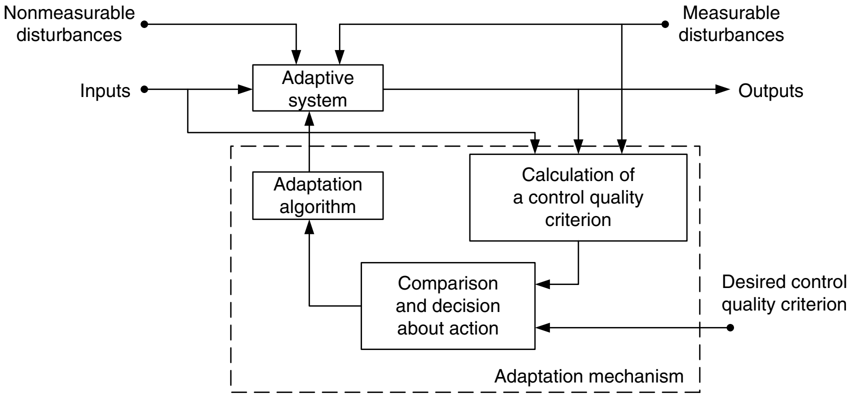

Nonmeasurable disturbances
Measurable disturbances
Inputs
Adaptive system
Outputs
Adaptation algorithm
Calculation of a control quality criterion
Comparison and decision about action
Desired control quality criterion
Adaptation mechanism

FIGURE 4.15 A general structure of an adaptive control system.

in the adaptation algorithm defines how parameters or signal will be changed. All these functions together form an adaptation mechanism.

While planning to solve a control problem by using an adaptation mechanism, we should keep in mind real system characteristics, especially the imposed limits on key system variables, type of implementation, and the width of operating range. A digital implementation usually requires a sufficiently fast adaptation algorithm so that it can be executed within a desired control interval. This in turn imposes a requirement for the simplest possible design of the adaptation algorithm structure.

4.2.1 Direct and Indirect Adaptive Control

Adaptive systems can be of a direct or an indirect type. A basic structure of a direct adaptive control system does neither involve identification nor estimation of process parameters, but may involve estimation of process variables. An adaptation action is formed directly from a specified control quality criterion. A typical example of direct adaptive control is model reference-based adaptive control (MRAC), which was first introduced by Whitaker et al. for control of an aircraft [27]. An aircraft represented a system with large parameter variations, so a reference model was used for setting desired dynamics and for adaptation of controller parameters in order to compensate for process parameter variations. In such control schemes adaptation is based on the current value of the tracking error e_M, which denotes the difference between outputs of the reference model, y_M, and the actual system, y_f. MRAC structure with a parallel reference model and parameter adaptation is shown in Figure 4.16. A reference model can also be used as a basis for generation of an adaptation signal u_A, which may be added either to the input or to the output of the feedback controller (Figure 4.17).

AMRAC approach has been widely adopted in fuzzy adaptive control schemes [28–35]. In fact, the resemblance of classical and fuzzy MRAC control schemes is absolute; only the way in which the design is carried out is different. In the

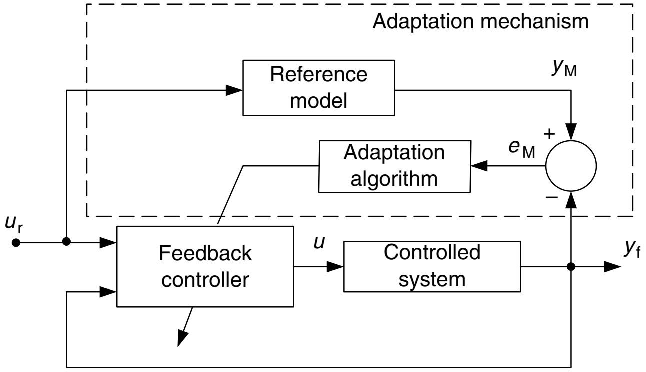


FIGURE 4.16 The structure of a model reference adaptive control system with parameter adaptation.

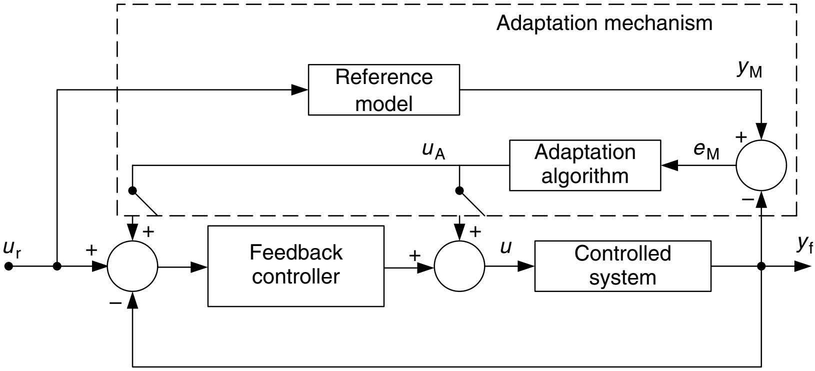


FIGURE 4.17 The structure of a model reference adaptive control system with signal adaptation.

following sections we explain the differences and describe how particular fuzzy MRAC schemes can be designed for selected control problems.

The concept of Variable Structure Systems (VSS) is another popular approach to a direct adaptive control design, which was first introduced by Emelyanov [36]. Very often, this type of control is called Sliding Mode Control (SMC). The goal of the VSS adaptive system is to make the system insensitive to parameter variations and external disturbances by changing the control law, depending on the current state of system state space vector. Parameter variations and disturbances cause deviations from a desired quality of control and the idea of the VSS approach is to push the disturbed control system first toward the point of equilibrium defined by a desired control quality criterion (this action is called the reaching mode of operation), and then keep the system there by using a switching type of an adaptation law function. This eventually leads to the so-called sliding mode of

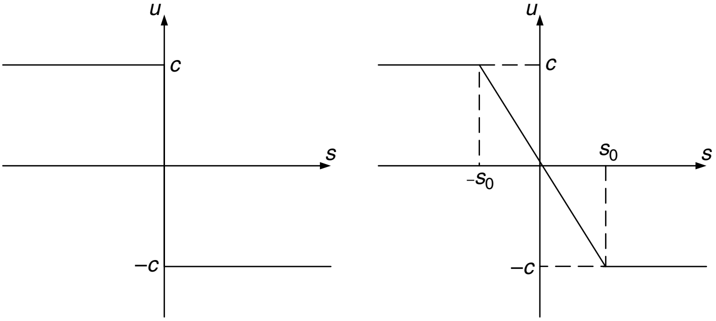


FIGURE 4.18 Switching laws in variable structure system adaptive control systems.

operation. Let n denote the number of inputs of the sliding mode controller. When SMC with only one input is considered (e.g., system error e), then a sliding mode is associated with a switching line

$$S|_{n=1} = s = e \tag{4.14}$$

If two inputs are considered (e.g., e and $\mathrm{d}e/\mathrm{d}t$), then a sliding mode is associated with a switching plane

$$S|_{n=2} = e + \lambda \frac{\mathrm{d}e}{\mathrm{d}t} \tag{4.15}$$

For a multi-input sliding mode controller, under the assumption that parameter variations and external disturbances are bounded (which is true for real systems), a sliding mode operation is associated with a predefined bounded hyperplane $S|_n$ of the state space. Once the sliding mode is reached, staying in it implies that we have ensured stability of an SMC system in the equilibrium point, which may be achieved by providing that the product of switching hyperplane $S|_n$ and its derivative $\mathrm{d}(S|_n)/\mathrm{d}t$ is always negative

$$S|_n \cdot \frac{\mathrm{d}(S|_n)}{\mathrm{d}t} < 0 \tag{4.16}$$

The switching carried out on hyperplane $S|_n$ should enforce the point on the hyperplane defined by current system dynamics to slide down the hyperplane to the equilibrium point defined by the origin of all coordinates. The problem with SMC systems lies in switching control laws having a form of a discontinuous signum function (see Figure 4.18)

$$u = -c\,\mathrm{sign}(s) \tag{4.17}$$

what may induce a serious chattering problem and may excite unmodelled system dynamics. Therefore, in alternative SMC solutions, the signum function is usually replaced with a continuous saturation function. Besides the form with a constant gain

$$u = -\frac{c}{s_0}s \tag{4.18}$$

shown in Figure 4.18 (recall that $s = e$), other saturation forms with different gains for various s values can also be used.

A VSS adaptive control approach, due to its operation on a predefined bounded hyperplane of the state space, is very convenient for introduction of fuzzy logic. For example, in single input control systems we may reduce the SMC problem to control of a system error phase trajectory. Then, instead of defining exact gain coefficients for various s values in the switching algorithm, nonlinear interpolating characteristics of a fuzzy control algorithm can contribute to a smooth online tuning of the saturation gain coefficient on the basis of heuristic statements such as “If s is high then increase the gain,” or “If s is low then decrease the gain.” Such reasoning has been adopted and successfully applied in different applications [37–42].

Indirect adaptive control involves identification and estimation of process parameters, which are thereafter used for the synthesis of controller parameters or for generation of an adaptation signal. Indirect adaptive control can be recognized in the concept of a dynamic programming of Bellman and Feldbaum, who introduced the so-called dual control that included two parts, an identification (parameter estimation) part and a control part [43,44]. On the same track, Astrom and Wittenmark introduced self-tuning regulators [45,46], which first performed identification of process parameters by using, for example, a recursive least squares (RLS) method of parameter estimation and then, based on these parameter values, did the synthesis of controller parameters using methods such as Ziegler–Nichols, Takahashi, and Dahlin procedures [47–49].

Indirect adaptive control schemes are usually more complex than direct control schemes and demand more computational effort. The structure of a self-tuning controller is shown in Figure 4.19.

In practice, adaptation is used for improving control of nonlinear systems, and therefore, adaptation algorithms are usually nonlinear. The nonlinear control theory, although well developed, still does not have answers to all questions regarding a practical adaptive controller design. Mainly, the theory is concerned with a problem of system stability, and the synthesis of adaptation algorithms is guided by the idea of satisfying a certain stability criterion. Such a design results in the required form of an adaptation mechanism, but it does not give answers about exact values of adaptation mechanism parameters. The designer becomes responsible for their determination, which presents, although theoretically guided, actually a heuristic intervention (see discussion about stability issues in Section 2.5).

A problem arises when the controlled process contains more dominant nonlinearities, making things even worse if they were periodic or discontinuous. Good

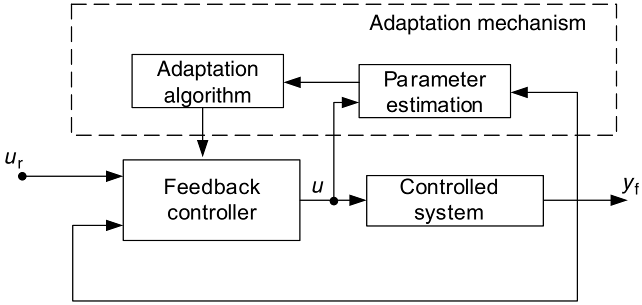


FIGURE 4.19 The structure of a self-tuning controller.

examples are robot joint positioning servo systems affected by periodic gravity-dependent loads and discontinuous static friction forces (not to neglect possible shaft elasticity and torsion effects). The design of adaptive controllers is based on a mathematical model of the system, and for such nonlinear systems the synthesis can be very demanding and sometimes very difficult from a practical point of view. Very often, in order to be able to find out a suitable adaptation law, observers need to be used to generate those signals from the process, which are not directly measurable (e.g., acceleration in a robot joint).

Because the fuzzy control design is primarily a model-free design, indirect adaptive fuzzy control schemes with implemented process identification and parameters estimation parts are rarely or never used in practice. Indeed, the mathematical model of a process can be replaced with a general (e.g., operator's) knowledge about the process reaction to certain control actions, and this knowledge may be used, for example, for generation of a fuzzy process model, but it cannot be used for determination of feedback controller parameters without large involvement of heuristics.

In the following sections, we describe a design of several direct adaptive fuzzy control schemes, which utilize a reference model in different ways for enforcement of the closed-loop control system to reach a desired level of performance. Elaborated design examples should help in clarifying some design issues, showing advantages and some drawbacks, as well as a practical value of each described method.

4.2.2 Model Reference Fuzzy Adaptive Control Systems

In Section 3.2, we have described the initial setting of a fuzzy rule table by using a second-order reference model. We have also mentioned that a model reference-based approach has been adopted in fuzzy adaptive control schemes. The goal of a model reference adaptive fuzzy controller design is to find a controller capable to keep a difference (i.e., a tracking error) between the reference model and the process as small as possible. This goal can be accomplished in several possible ways, as shown in Figure 4.20. One possible way is an adaptation of a fuzzy

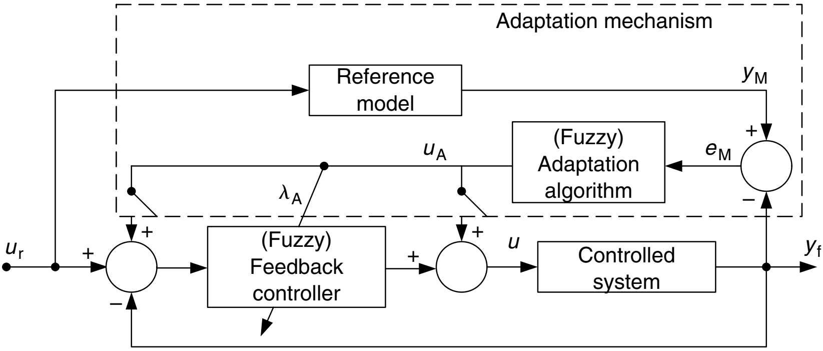


FIGURE 4.20 The structure of a fuzzy MRAC system.

feedback controller by using some of, say, classical adaptation methods. The other is the implementation of a fuzzy adaptation mechanism by using a form of the standard fuzzy control algorithm for this purpose. There is also a possibility to combine both ways and adapt a fuzzy feedback controller with a fuzzy adaptation mechanism.

Likewise in classical model reference adaptive control structures, a reference model determines a desired dynamic behavior of the closed-loop control system. As we know, at least from the experience of how a controlled process responds to its control input, when determining the reference model we must value that experience and try to pick the right dimension (order) of the reference model and the right values of its parameters.

Many practical systems allow insertion of a test signal in a selected operating point, so that we can draw conclusions about open-loop process dynamics. There are no constraints on the selection of the reference model type, so it may be linear, nonlinear, with a dead time, etc. However, in principle, we define the simplest possible form, which properly describes what we want from the closed-loop system. Under assumption that the controlled process is nonlinear, but linearizable in a selected operating point, then the reference model can also be a linear one.

From a theoretical point of view, determination of any linear reference model is not a problem. From a practical point of view, conditions in the field may not always be in favor of easy assessment of process dynamics, so determination of the recursive equation coefficients of a higher-order reference model could be a problem. Therefore, the definition of reduced-order reference models is advised [50]. For the plants with a relative degree not greater than 2, a second-order reference model becomes a preferable choice because of its simplicity and easy way of definition.

A recursive equation (3.28) describes a second-order reference model ready for implementation in a microcomputer. In order to get values of coefficients a_{M1}, a_{M2}, and b_{M1}, we must first determine the value of control interval T_d and values of reference model parameters. The easiest way, even for a nonexpert, is to determine "obvious" model response parameters such as overshoot σ_m and peak time t_m, which uniquely define a second-order model. Then from Equations (3.62), which we repeat here for convenience

$$\xi = \sqrt{\frac{\ln^2(\sigma_\mathrm{m}[\%]/100)}{\pi^2 + \ln^2(\sigma_\mathrm{m}[\%]/100)}}, \quad \omega_n = \frac{\pi}{t_\mathrm{m}\sqrt{1-\xi^2}} \tag{4.19}$$

we can determine damping coefficient ξ and natural frequency ω_n. They figure as parameters in the reference model transfer function (3.63), which can be rewritten as

$$\begin{aligned} G_\mathrm{M}(s, \boldsymbol{\lambda}_\mathrm{M}) = \frac{Y_\mathrm{M}(s, \boldsymbol{\lambda}_\mathrm{M})}{U_\mathrm{R}(s)} &= \frac{1}{(s^2/\omega_\mathrm{n}^2) + (2\xi/\omega_\mathrm{n})s + 1} \\ &= \frac{1}{(T_\mathrm{M}/K_\mathrm{M}) \cdot s^2 + (1/K_\mathrm{M}) \cdot s + 1} \end{aligned} \tag{4.20}$$

where $\boldsymbol{\lambda}_\mathrm{M} = [K_\mathrm{M}\ T_\mathrm{M}]^\mathrm{T}$, $K_\mathrm{M} = \omega_\mathrm{n}/2\xi$, $T_\mathrm{M} = (1/2\xi\omega_\mathrm{n})$ are the model gain and the time constant, respectively.

After applying an Euler discretization for the given control interval T_d, we first get a discrete form $G_\mathrm{M}(z)$ and then a recursive equation (3.28).

Care must be taken to avoid unrealistic settings of model dynamics that would be too far from the achievable process dynamics (e.g., with inappropriate model settings we might induce saturation of the control input that would prevent the system from following the model). Also, at first we must observe whether the process possesses a significant delay, and if that is the case, then we must account for it either by including the delay in the reference model or by changing the feedback controller.

If a model reference fuzzy adaptive controller is thought of as a ready-to-use function block for industry applications, setting the overshoot σ_m and peak time t_m must cover both aperiodic and oscillatory reference model responses. The way in which this can be done is described in the case study in Section 7.2.3.

As depicted in Figure 4.6, besides a reference model, an adaptation mechanism contains also an adaptation algorithm, whose input signal is model tracking error e_M. In general, the positive sign of e_M means that in this particular moment the system is lagging behind the model, for the negative sign it is just the opposite. For a given model and a process, e_M can change its sign an arbitrary number of times during a transient response. How can the model tracking error e_M then be used for adaptation? If e_M is constantly positive, this normally means that we need to increase the control input to speed up the system response and cancel the steady-state error. If e_M is constantly negative, we need to slow down, that is, we must decrease the control input.

If a reduced-order (e.g., second-order) reference model is used to dictate dynamics of a high-order controlled process, due to the influence of unmodeled dynamics, we can never fulfill the requirement $e_\mathrm{M} = 0$, as there will always be

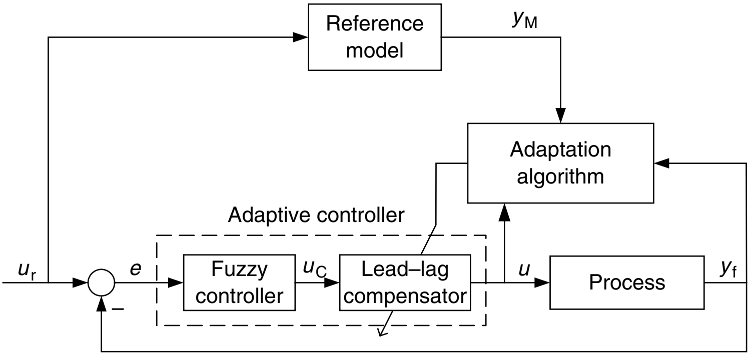


FIGURE 4.21 The structure of the adaptive control system.

some error. Therefore, we set a design goal

$$\{|e_M(t)|, |e_M(k)|\} < \varepsilon_M \tag{4.21}$$

that should keep the model tracking error within the specified boundary ε_M.

4.2.2.1 Sensitivity Model-Based Adaptation

A decision on which algorithm would be appropriate for adaptation of the target control system depends on many factors; desired system precision and dynamics, online computational power, noise level, nonlinearities encountered in the system, accuracy of the linearized model, etc. We may use a whole palette of design possibilities already discussed in Section 4.2.1, but if we think of fuzzy control as a model-free control, then we do not want to use adaptation methods based on the system's mathematical model.

Let us consider an adaptation of fuzzy control systems with unknown, time-varying, probably nonlinear, and stable processes, whose dynamics in a selected operating point can be approximated with a second-order transfer function (3.59).

For the purpose of adaptation, a second-order reference model (4.20) will be used to define the desired (nominal) behavior of the adaptive fuzzy control system.

By changing the operating point, nonlinear system parameters will also change. This will impose the need for adjustment of fuzzy controller parameters in various operating points. Instead of adapting a fuzzy controller, let us add a lead–lag compensator in series to the fuzzy controller and adapt its parameters, as shown in Figure 4.21.

The lead–lag compensator is described with a transfer function

$$G_f(s) = \frac{U(s)}{U_c(s)} = K_f \cdot \frac{T_f \cdot s + 1}{T_M \cdot s + 1} \tag{4.22}$$

where K_f is a gain coefficient and T_f is a lead-time constant, while T_M refers to the reference model time constant.

Lead–lag compensators are quite extensively used in control. A lead compensator can increase the stability or speed of the system's response; a lag compensator can reduce (but not eliminate) the steady-state error. A lead–lag compensator combines the effects of the lead compensator with those of the lag compensator, allowing the user to adjust frequency characteristics of the open-loop control system in order to obtain required closed-loop performance. The result is a system with improved transient response, stability, and reduced steady-state error.

By observing a difference between reference model and system outputs, our intention is to adapt K_f and T_f so that the model tracking error satisfies condition (4.21). In order to do that, one possible way is to build a sensitivity model that would describe the dependence of y_f on K_f and T_f. Then we can use the sensitivity functions obtained in that way [51,52] as a basis for adaptation. Since the sensitivity functions are the measure of how y_f changes when K_f and T_f change, this imposes differentiability of functions describing the relations of y_f on K_f and T_f.

When only influence on y_f is considered, like in our case, then the Kokotović method of sensitivity points is preferable to the computationally more demanding canonical system sensitivity model, due to its simplicity and parallel computation of all sensitivity functions [52,53]. The idea of using a sensitivity model as a basis for adaptation has its roots in classical control, let us just mention that the Kokotović sensitivity model with output sensitivity functions has been effectively used for the synthesis and adjustment of standard linear and dead-beat control algorithm parameters [54–56].

Denoting the controller parameter vector as $\boldsymbol{\lambda}_\mathrm{c}$, the lead–lag compensator parameter vector as $\boldsymbol{\lambda}_\mathrm{f}$, and the process parameter vector as $\boldsymbol{\lambda}_\mathrm{P}$, the system output depends on the controlled system parameter vector $\boldsymbol{\lambda} = [\boldsymbol{\lambda}_\mathrm{c} \quad \boldsymbol{\lambda}_\mathrm{f} \quad \boldsymbol{\lambda}_\mathrm{P}]^\mathrm{T}$ in the following way:

$$Y_\mathrm{f}(s,\boldsymbol{\lambda}) = G_\mathrm{cl}(s,\boldsymbol{\lambda})U_\mathrm{r}(s) = \frac{G_\mathrm{o}(s,\boldsymbol{\lambda})}{1+G_\mathrm{o}(s,\boldsymbol{\lambda})}U_\mathrm{r}(s) \tag{4.23}$$

where $G_\mathrm{cl}(s,\boldsymbol{\lambda})$ is the closed-loop transfer function, $G_\mathrm{o}(s,\boldsymbol{\lambda}) = G_\mathrm{c}(s,\boldsymbol{\lambda}_\mathrm{c})G_\mathrm{f}(s,\boldsymbol{\lambda}_\mathrm{f})$ $G_\mathrm{P}(s,\boldsymbol{\lambda}_\mathrm{P})$ is the open-loop transfer function, $G_\mathrm{c}(s,\boldsymbol{\lambda}_\mathrm{c})$ is the transfer function of the feedback (fuzzy) controller, $G_\mathrm{f}(s,\boldsymbol{\lambda}_\mathrm{f})$ is the transfer function of the lead–lag compensator, and $G_\mathrm{P}(s,\boldsymbol{\lambda}_\mathrm{P})$ is the transfer function of the process.

A semirelative sensitivity function describes the dependence of $Y_\mathrm{f}(s,\boldsymbol{\lambda})$ on the relative variation of particular system parameter λ_i:

$$\begin{aligned}\tilde{\eta}_\mathrm{i}(s,\boldsymbol{\lambda}) &= \left.\frac{\partial Y_\mathrm{f}(s,\boldsymbol{\lambda})}{\partial \ln \lambda_\mathrm{i}}\right|_{\boldsymbol{\lambda}} = \lambda_\mathrm{i}\left.\frac{\partial Y_\mathrm{f}(s,\boldsymbol{\lambda})}{\partial \lambda_\mathrm{i}}\right|_{\boldsymbol{\lambda}} = \lambda_\mathrm{i}\left.\frac{\partial G_\mathrm{cl}(s,\boldsymbol{\lambda})}{\partial \lambda_\mathrm{i}}\right|_{\boldsymbol{\lambda}} U_\mathrm{r}(s) \\ &= \frac{1}{1+G_\mathrm{o}(s,\boldsymbol{\lambda})}S_\mathrm{i}(s)Y_\mathrm{f}(s,\boldsymbol{\lambda})\end{aligned} \tag{4.24}$$

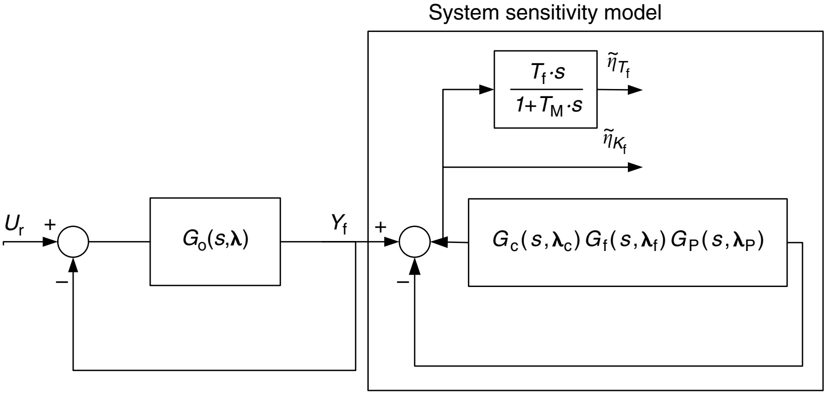


FIGURE 4.22 A sensitivity model of the controlled system.

where $S_i(s) = (\lambda_i/G_o(s,\boldsymbol{\lambda}))/(\delta G_o(s,\boldsymbol{\lambda})/\delta\lambda_i)$ is a so-called Bode sensitivity transfer function.

Since we want to compensate all changes in system parameters by changes of K_f and T_f, the first step in the adaptive control design is using the Kokotović method of sensitivity points in order to get Bode sensitivity transfer functions related to K_f and T_f:

$$S_{K_f}(s,\boldsymbol{\lambda}) = \frac{K_f}{G_o(s,\boldsymbol{\lambda})}\frac{\partial G_o(s,\boldsymbol{\lambda})}{\partial K_f} = \frac{K_f}{G_o(s,\boldsymbol{\lambda})}\frac{\partial G_o(s,\boldsymbol{\lambda})}{\partial G_f(s,\boldsymbol{\lambda}_f)}\frac{\partial G_f(s,\boldsymbol{\lambda}_f)}{\partial K_f} = 1$$

$$S_{T_f}(s,\boldsymbol{\lambda}) = \frac{T_f}{G_o(s,\boldsymbol{\lambda})}\frac{\partial G_o(s,\boldsymbol{\lambda})}{\partial T_f} = \frac{T_f}{G_o(s,\boldsymbol{\lambda})}\frac{\partial G_o(s,\boldsymbol{\lambda})}{\partial G_f(s,\boldsymbol{\lambda}_f)}\frac{\partial G_f(s,\boldsymbol{\lambda}_f)}{\partial T_f} = \frac{T_f s}{1+T_M s} \tag{4.25}$$

The block diagram of the corresponding sensitivity model is shown in Figure 4.22. Because the transfer function of the controlled process is unknown (it is known only in terms of desired system dynamics defined by reference model [4.20]), and because the feedback controller is treated as a black box, the system sensitivity model cannot be determined.

Instead of deriving an unknown sensitivity model of the controlled system, we can derive a sensitivity model of reference model (4.20) and use the so-obtained semirelative sensitivity functions

$$\tilde{\eta}_{K_M}(s,\boldsymbol{\lambda}_M) = \frac{\partial Y_M(s,\boldsymbol{\lambda}_M)}{\partial K_M/K_M} \quad \text{and} \quad \tilde{\eta}_{T_M}(s,\boldsymbol{\lambda}_M) = \frac{\partial Y_M(s,\boldsymbol{\lambda}_M)}{\partial T_M/T_M}$$

in place of $\tilde{\eta}_{K_\mathrm{f}}$ and $\tilde{\eta}_{T_\mathrm{f}}$ for adaptation of K_f and T_f. For the given model and its parameters, semirelative sensitivity functions have the following form:

$$\tilde{\eta}_{M_\mathrm{i}}(s,\boldsymbol{\lambda}_\mathrm{M}) = \lambda_{M_\mathrm{i}}\frac{\delta Y_\mathrm{M}(s,\boldsymbol{\lambda}_\mathrm{M})}{\delta\lambda_{M_\mathrm{i}}} = \frac{1}{1+G_\mathrm{oM}(s,\boldsymbol{\lambda}_\mathrm{M})}S_{M_\mathrm{i}}(s)Y_\mathrm{M}(s,\boldsymbol{\lambda}_\mathrm{M}) \tag{4.26}$$

where

$$\begin{aligned} S_{M_\mathrm{i}}(s,\boldsymbol{\lambda}_\mathrm{M}) &= \frac{\lambda_{M_\mathrm{i}}}{G_\mathrm{oM}(s,\boldsymbol{\lambda}_\mathrm{M})}\frac{\delta G_\mathrm{oM}(s,\boldsymbol{\lambda}_\mathrm{M})}{\delta\lambda_{M_\mathrm{i}}} \\ G_\mathrm{oM}(s,\boldsymbol{\lambda}_\mathrm{M}) &= \frac{K_\mathrm{M}}{s(1+T_\mathrm{M}s)} \\ \boldsymbol{\lambda}_\mathrm{M} &= [K_\mathrm{M} \quad T_\mathrm{M}]^\mathrm{T} \end{aligned} \tag{4.27}$$

Sensitivity functions are obtained in the sensitivity points of the sensitivity model by adding Bode transfer functions blocks $S_\mathrm{KM}(s)$ and $S_\mathrm{TM}(s)$. For reference model (4.20), we get

$$S_\mathrm{KM}(s) = 1, \quad S_\mathrm{TM}(s) = -\frac{T_\mathrm{M}s}{1+T_\mathrm{M}s} \tag{4.28}$$

Once we have reference model sensitivity model (4.28) derived, we can use it as an approximation of an unknown system sensitivity model. As shown in Figure 4.23, we can feed process output $Y_\mathrm{f}(s,\boldsymbol{\lambda})$ into the reference model sensitivity model, generating approximate semi-relative sensitivity functions $\tilde{\eta}_{K_\mathrm{fM}}$ and $\tilde{\eta}_{T_\mathrm{fM}}$, which may be used instead of the real ones, that is, $\tilde{\eta}_{K_\mathrm{f}} \approx \tilde{\eta}_{K_\mathrm{fM}}$ and $\tilde{\eta}_{T_\mathrm{f}} \approx \tilde{\eta}_{T_\mathrm{fM}}$.

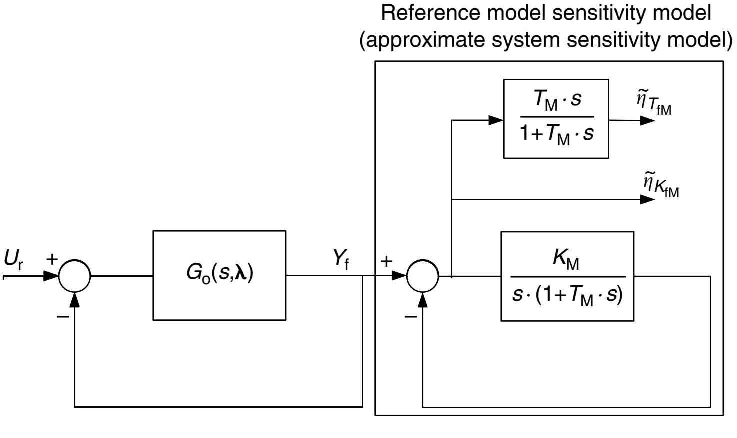

Reference model sensitivity model
(approximate system sensitivity model)
$\frac{T_\mathrm{M}\cdot s}{1+T_\mathrm{M}\cdot s}$
$\tilde{\eta}_{T_\mathrm{fM}}$
$\tilde{\eta}_{K_\mathrm{fM}}$
U_r
+
−
$G_\mathrm{o}(s,\boldsymbol{\lambda})$
Y_f
+
−
$\frac{K_\mathrm{M}}{s\cdot(1+T_\mathrm{M}\cdot s)}$

FIGURE 4.23 An approximate sensitivity model of the high-order control process.

By switching to the discrete time domain, changes of system output $y_\mathrm{f}(k)$ due to small variations of the lead–lag compensator parameters are given by the following:

$$\Delta y_\mathrm{f}^{\kappa+1}(k, \boldsymbol{\lambda}_\mathrm{f}^{\kappa}) = \sum_i \tilde{\eta}^{\kappa+1}_{\lambda_{\mathrm{fM}i}}(k)\frac{\Delta\lambda^{\kappa+1}_{f_\mathrm{i}}}{\lambda^{\kappa}_{f_\mathrm{i}}} = \tilde{\eta}^{\kappa+1}_{K_\mathrm{fM}}(k)\frac{\Delta K_\mathrm{f}^{\kappa+1}}{K_\mathrm{f}^{\kappa}} + \tilde{\eta}^{\kappa+1}_{T_\mathrm{fM}}(k)\frac{\Delta T_\mathrm{f}^{\kappa+1}}{T_\mathrm{f}^{\kappa}} \tag{4.29}$$

$$\begin{aligned} K_\mathrm{f}^{\kappa+1} &= K_\mathrm{f}^{\kappa} + \Delta K_\mathrm{f}^{\kappa+1} \\ T_\mathrm{f}^{\kappa+1} &= T_\mathrm{f}^{\kappa} + \Delta T_\mathrm{f}^{\kappa+1} \end{aligned} \tag{4.30}$$

where κ denotes the number of the tuning iteration, and $\Delta K_\mathrm{f}^{\kappa+1}$ and $\Delta T_\mathrm{f}^{\kappa+1}$ are respective changes of lead–lag compensator parameters.

Initially ($\kappa = 0$), $K_\mathrm{f}^0 = 1$, and $T_\mathrm{f}^0 = T_\mathrm{M}$, so that the lead–lag compensator has no influence on the control loop dynamics at the beginning of adaptation. As κ is increasing and K_f is changing, the open-loop gain of the control system is expected to converge to the value of reference model gain K_M. Similarly, lead-time constant T_f is expected to converge to the value of a dominant process time constant, so that lag time constant T_M determined by the reference model becomes a dominant one.

In order to get better control over the tuning algorithm dynamics (speed of convergence), let us make the following modification of tuning algorithm (4.30):

$$\begin{aligned} K_\mathrm{f}^{\kappa+1} &= K_\mathrm{f}^{\kappa} + \gamma_\mathrm{K} \cdot \Delta K_\mathrm{f}^{\kappa+1} \\ T_\mathrm{f}^{\kappa+1} &= T_\mathrm{f}^{\kappa} + \gamma_\mathrm{T} \cdot \Delta T_\mathrm{f}^{\kappa+1} \end{aligned} \tag{4.31}$$

where γ_K and γ_T are tuning coefficients.

There are two problems related to the viability of the tuning law (4.31): how $\Delta K_\mathrm{f}^{\kappa+1}$ and $\Delta T_\mathrm{f}^{\kappa+1}$ should vary with $\Delta y_\mathrm{f}^{\kappa+1}(k)$, and how to find tuning coefficients γ_K and γ_T, which guarantee convergence of tuning and thus provide overall stability of the closed-loop system.

Lead–lag compensator parameter variations $\Delta K_\mathrm{f}^{\kappa+1}$ and $\Delta T_\mathrm{f}^{\kappa+1}$, which provide desired changes of system response $\Delta y_\mathrm{f}^{\kappa+1}(k)$, may be computed directly from (4.30) if semirelative sensitivity functions $\tilde{\eta}^{\kappa+1}_{K_\mathrm{fM}}(k)$ and $\tilde{\eta}^{\kappa+1}_{T_\mathrm{fM}}(k)$ are known. The following strategy is adopted: $\Delta K_\mathrm{f}^{\kappa+1}$ is calculated in the moment $k = k_\mathrm{m}$ when $\tilde{\eta}^{\kappa+1}_{K_\mathrm{fM}}(k)$ reaches its maximum, while $\Delta T_\mathrm{f}^{\kappa+1}$ is calculated in the moment $kT_\mathrm{d} = t_\mathrm{m}$ when system output $y_\mathrm{f}^{\kappa+1}(k)$ reaches the peak value:

$$\begin{aligned} \Delta K_\mathrm{f}^{\kappa+1} &= \frac{\Delta y_\mathrm{f}^{\kappa+1}(k_\mathrm{m})}{\max\left[\eta^{\kappa+1}_{K_\mathrm{fM}}(k_\mathrm{m})\right]} K_\mathrm{f}^{\kappa} \\ \Delta T_\mathrm{f}^{\kappa+1} &= \frac{\Delta y_\mathrm{f}^{\kappa+1}(t_\mathrm{m})}{\eta^{\kappa+1}_{T_\mathrm{fM}}(t_\mathrm{m})} T_\mathrm{f}^{\kappa} \end{aligned} \tag{4.32}$$

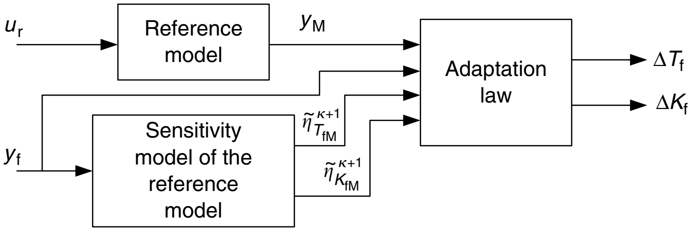

u_r
Reference model
y_M
Adaptation law
ΔT_f
ΔK_f
y_f
Sensitivity model of the reference model
$\tilde{\eta}^{\kappa+1}_{T_{fM}}$
$\tilde{\eta}^{\kappa+1}_{K_{fM}}$

FIGURE 4.24 The block diagram of an adaptation algorithm.

During execution of tuning algorithm (4.32) we must pay attention to avoid possible division by zero. This can be handled by setting minimal threshold values for the sensitivity functions $\tilde{\eta}^{\kappa+1}_{K_{fM}}(k)$ and $\tilde{\eta}^{\kappa+1}_{T_{fM}}(k)$. For the sensitivity functions' values, below these threshold values the tuning algorithm will not be executed and K_f and T_f will not change.

Given change of the system output $\Delta y_f^{\kappa+1}(k)$ coincides in the model reference control concept with model tracking error $e_M^{\kappa+1}(k) = y_M^{\kappa+1}(k) - y_f^{\kappa+1}(k)$. Thus $\Delta y_f^{\kappa+1}(k)$ in (4.32) must be replaced with $e_M^{\kappa+1}(k)$.

Now remains the definition of tuning coefficients γ_K and γ_T . In general, larger values of tuning coefficients cause larger changes of K_f and T_f. Care must be taken to choose a value of γ_T, which would not cause a negative value for T_f. This would change the structure of the lead–lag compensator and make the closed-loop system unstable.

The structure of the adaptation mechanism is shown in Figure 4.24.

Example 4.2 Sensitivity model-based adaptation of the fuzzy controller.

In order to examine the efficacy of the reference model and the sensitivity model-based adaptation algorithm in so-called "ideal conditions," the adaptive control loop contains only a lead–lag compensator and time-varying second-order process $G_p(s)$. The second-order reference model is defined with maximum overshoot $\sigma_m = 5.5\%$ and peak time $t_m = 0.025$ sec, which yields $K_M = 121$, $T_M = 4.3$ msec. The lead–lag compensator parameters were initially set to $K_f^0 = 1$ and $T_f^0 = T_M$. Tuning coefficients have been set to $\gamma_K = 0.4$ and $\gamma_T = 0.3$.

In the simulation experiment, both process parameters have been changed to a large extent, $K_P = K_M/3$, $T_P = 3 \cdot T_M$. The responses of the adaptive control system during the adaptation process are shown in Figure 4.25.

Although changes of both process parameters are assumed to be very large, convergence of adaptation in the case of controlling a second-order process is very fast, as indicated in Figure 4.26.

Now, let us test the adaptive fuzzy logic controller in a linearized model of a high-order process — a servo system with a permanent magnet synchronous motor. The structure of the controlled process is shown in Figure 4.27. Nominal values of process parameters are as follows: $K_{ch} = 0.9837$ V/V — chopper gain, $T_{ch} = 0.05$ msec — chopper time constant, $J = 0.00176$ kg m^2 — moment of inertia, $K = 0.000388$ Vsec — motor constant, $K_t = 0.063$ Vsec — tachometer gain, $T_t = 0.0025$ sec — tachometer time constant.

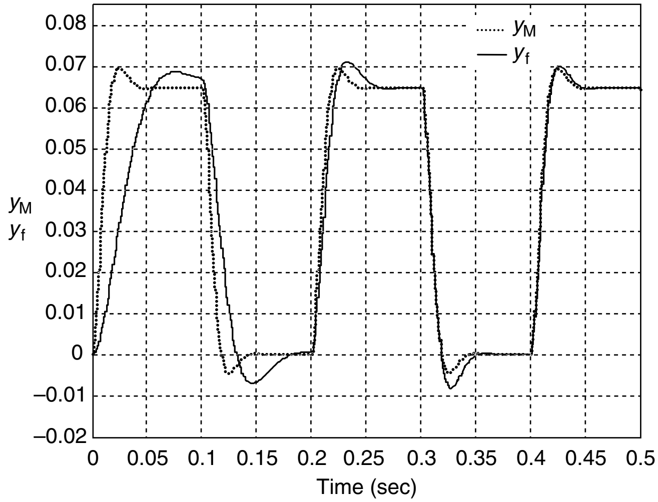


FIGURE 4.25 Process and reference model responses from the start of adaptation, $K_P = K_M/3$, $T_P = 3 \cdot T_M$. (From Kovačić, Z., Bogdan, S., and Punčec, M., *IEEE Intl. Conf. Industr. Technol. ICIT'03*, 321–326, 2003. With permission.)

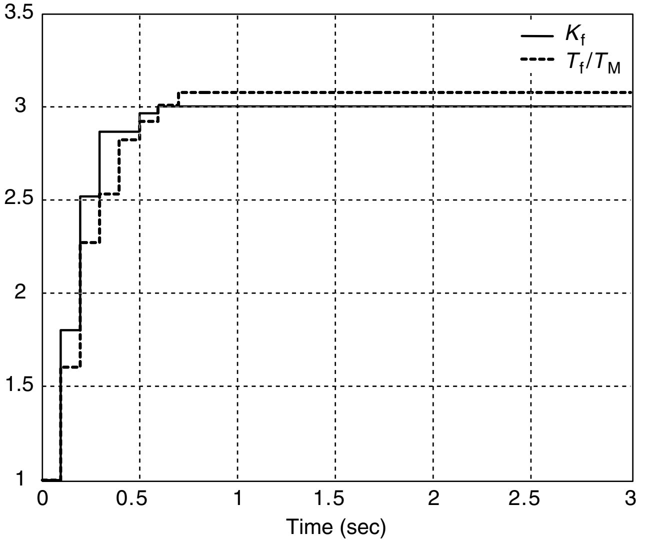


FIGURE 4.26 Convergence of lead–lag compensator gain K_f and lead-time constant T_f, $K_P = K_M/3$, $T_P = 3 \cdot T_M$. (From Kovačić, Z., Bogdan, S., and Punčec, M., *IEEE Intl. Conf. Industr. Technol. ICIT'03*, 321–326, 2003. With permission.)

The second-order reference model (4.20) defined with maximum overshoot $\sigma_M = 5.5\%$ and peak time $t_m = 0.025$ sec describes a desired closed-loop behavior. The control surface of the fuzzy controller has been set by emulation of the PI controller according to (3.8). PI controller parameters were determined for nominal process parameters. The lead–lag compensator parameters were initially set to

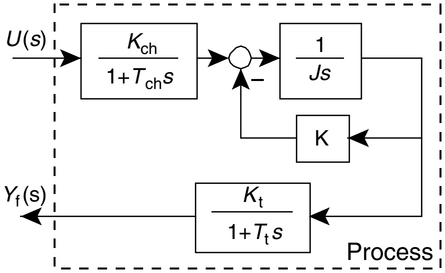


FIGURE 4.27 The structure of the controlled process. (From Kovačić, Z., Bogdan, S., and Punčec, M., *IEEE Intl. Conf. Industr. Technol. ICIT'03*, 321–326, 2003. With permission.)

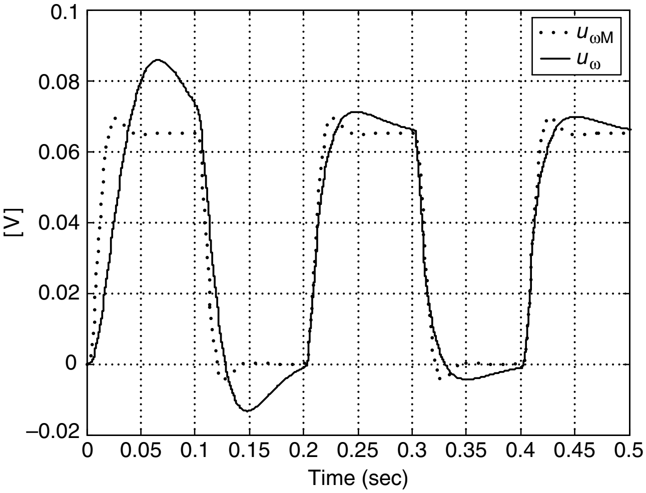


FIGURE 4.28 Process and reference model responses from the start of adaptation, $J = 5 \cdot J_n$. (From Kovačić, Z., Bogdan, S., and Punčec, M., *IEEE Intl. Conf. Industr. Technol. ICIT'03*, 321–326, 2003. With permission.)

$K_f^0 = 1$ and $T_f^0 = T_M$. Tuning coefficients have been set to $\gamma_K = 0.4$ and $\gamma_T = 0.3$.

Simulation experiments have a goal to test adaptation to large variations of the moment of inertia. In the first experiment, J has been set five times larger than the nominal value, $J = 5 \cdot J_n$. Figure 4.28 and Figure 4.29 show development of process and reference model responses from the start to the end of adaptation.

A very large initial tracking error $e_M = y_M - y_f$ is reduced several times after only two iterations. One can see in Figure 4.30 that values of lead–lag compensator parameters converge to their steady-state values in four iterations.

In the experiment that follows, the open-loop gain K_o has been increased five times. A rapid improvement of process responses is shown in Figure 4.31 resulting in the close follow-up of the reference model (Figure 4.32) providing rather swift

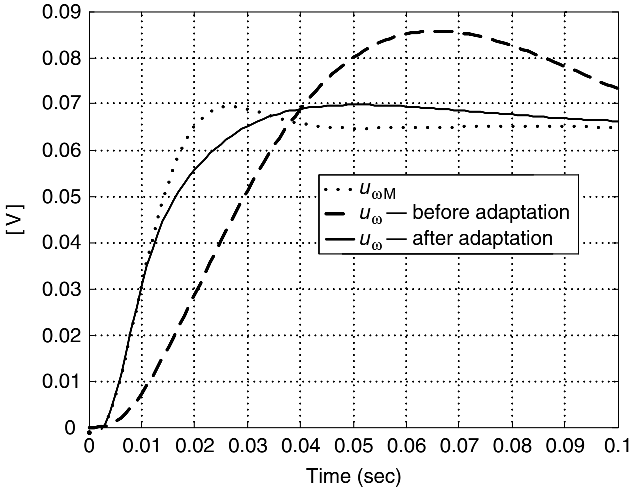


FIGURE 4.29 Process and reference model responses before and after adaptation, $J = 5 \cdot J_n$. (From Kovačić, Z., Bogdan, S., and Punčec, M., *IEEE Intl. Conf. Industr. Technol. ICIT'03*, 321–326, 2003. With permission.)

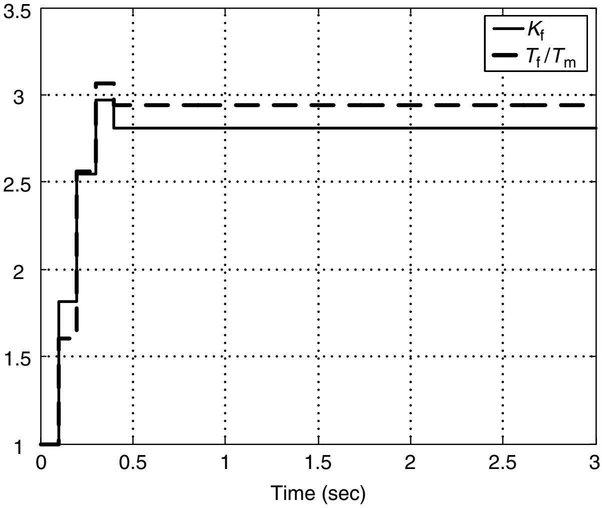


FIGURE 4.30 Convergence of the lead–lag compensator gain coefficient K_f and lead-time constant T_f, $J = 5 \cdot J_n$. (From Kovačić, Z., Bogdan, S., and Punčec, M., *IEEE Intl. Conf. Industr. Technol. ICIT'03*, 321–326, 2003. With permission.)

reduction of the tracking error. Also, initial oscillations in the system response have been completely eliminated.

Convergence of parameter values is rather fast and completed in approximately ten iterations (Figure 4.33). Adaptation is stopped when the average sum of last

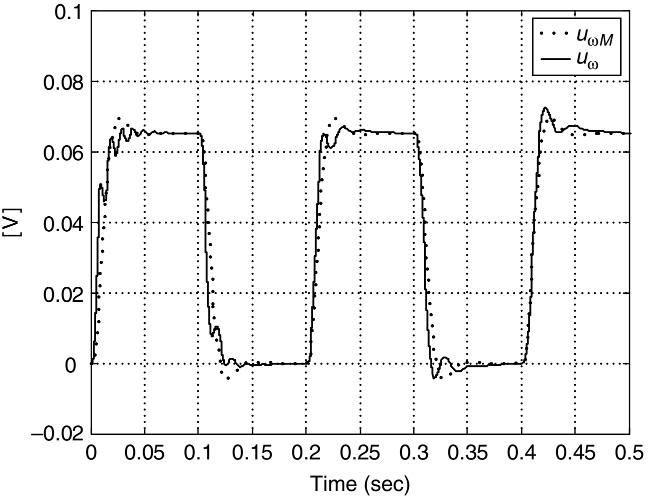


FIGURE 4.31 Process and reference model responses from the start of adaptation, $5 \cdot K_{\text{on}}$. (From Kovačić, Z., Bogdan, S., and Punčec, M., *IEEE Intl. Conf. Industr. Technol. ICIT'03*, 321–326, 2003. With permission.)

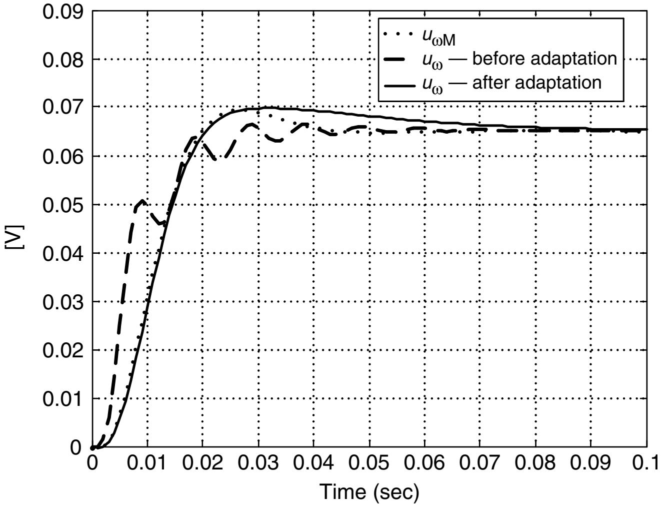


FIGURE 4.32 Process and reference model responses before and after adaptation, $5 \cdot K_{\text{on}}$. (From Kovačić, Z., Bogdan, S., and Punčec, M., *IEEE Intl. Conf. Industr. Technol. ICIT'03*, 321–326, 2003. With permission.)

n values of $\Delta K_{\text{f}}^{\kappa+1}$ and $\Delta T_{\text{f}}^{\kappa+1}$ drops below specified threshold values. It should be noted that faster convergence could be obtained with larger γ_{K} and γ_{T}, but then adaptation might not be so smooth and might become unstable in the final instance. Therefore, system stability conditions should be worked out in terms of finding the tuning coefficients, which would guarantee the overall stability of the adaptive control system.

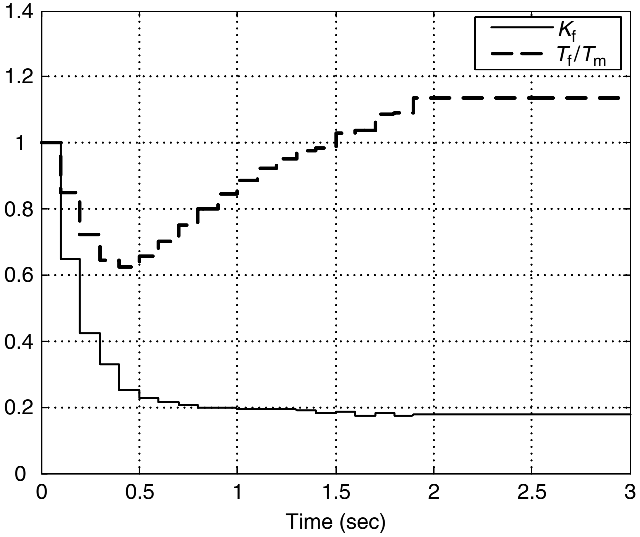


FIGURE 4.33 Convergence of the lead–lag compensator gain coefficient K_f and lead-time constant T_f, $5 \cdot K_\mathrm{on}$. (From Kovačić, Z., Bogdan, S., and Punčec, M., *IEEE Intl. Conf. Industr. Technol. ICIT'03*, 321–326, 2003. With permission.)

4.2.2.2 Integral Criterion-Based Adaptation

Besides the model tracking error e_M and the sensitivity model used in the adaptive control scheme shown in Figure 4.24, we can also use other reference model-based performance indices to find $\Delta K_\mathrm{f}^{\kappa+1}$ and $\Delta T_\mathrm{f}^{\kappa+1}$ in the tuning law (4.31). Very often integral criteria are used, such as the integral of squared error $\mathrm{ISE} = \int_0^T e_\mathrm{M}^2(t)\,\mathrm{d}t$, the integral of absolute error magnitude $\mathrm{IAE} = \int_0^T |e_\mathrm{M}(t)|\,\mathrm{d}t$, the integral of time-weighted absolute error $\mathrm{ITAE} = \int_0^T t|e_\mathrm{M}(t)|\,\mathrm{d}t$, and the integral of time-weighted squared error $\mathrm{ITSE} = \int_0^T te_\mathrm{M}^2(t)\,\mathrm{d}t$. The goal of adaptation is to adjust system parameters so that the selected integral criterion depending on the model tracking error e_M reaches a minimum value. In such case, an adaptive system is considered an optimal control system.

Instead of the above commonly used integral criteria, let us observe a performance criterion calculated as the ratio of integrals (areas) determined by the reference model output response y_M and the system output response y_f reaching a specified portion of a complete transient response (we assume that system noise has characteristics of white noise):

$$\rho_\mathrm{y} = \frac{\int_0^{t_{y_\mathrm{f}}} y_\mathrm{f}(t)\cdot\mathrm{d}t}{\int_0^{t_{y_\mathrm{M}}} y_\mathrm{M}(t)\cdot\mathrm{d}t} = \frac{I_{y_\mathrm{f}}(t_{y_\mathrm{f}})}{I_{y_\mathrm{M}}(t_{y_\mathrm{M}})} \tag{4.33}$$

where $y_\mathrm{f}(t_{y_\mathrm{f}}) = y_\mathrm{M}(t_{y_\mathrm{M}}) = a \cdot y_\mathrm{f}(t)|_{t\to\infty} = a \cdot y_\mathrm{M}(t)|_{t\to\infty}$, $a \in [0.5, 0.9]$.

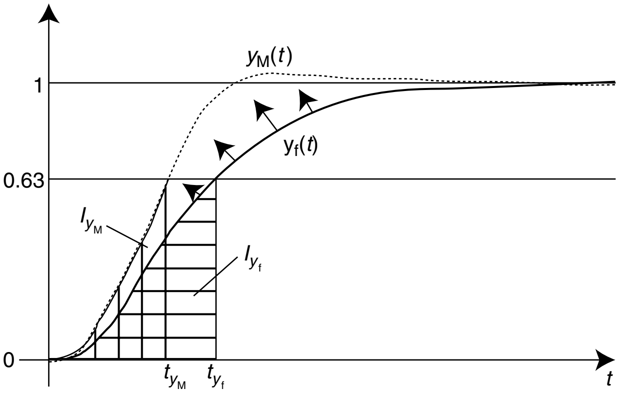


FIGURE 4.34 A reference model output integral (I_{y_M}) and a closed-loop system integral (I_{y_f}) for $a = 0.63$.

In order to get a broader range of values of ρ_y, a good choice is to calculate ρ_y during the rise time interval, when the picture of system dynamics is very clear (Figure 4.34).

For tuning law (4.31), we must find an adaptation algorithm, which will change K_f and T_f with respect to changes of ρ_y. For a given parameter a, $I_{y_M}(t_{y_M})$ is always constant, while $I_{y_f}(t_{y_f})$ and ρ_y vary with system dynamics. Treatment of the process as a black box, does not allow us to relate varying performance index ρ_y with the nominal process dynamics. However, we can search for an approximate solution in the reference model counterpart. Namely, variations of reference model parameters K_M and T_M will cause variations of $I_{y_M}(t_{y_M})$ similarly as process parameter variations will cause variations of $I_{y_f}(t_{y_f})$. So, if we apply an integral criterion (4.33) to the closed-loop system having a form of a second-order reference model (4.20), we obtain:

$$\rho_{y_M} = \frac{\int_0^{t_{y_M}+\Delta t_{y_M}} y_M(t, K_M + \Delta K_M, T_M + \Delta T_M) \cdot \mathrm{d}t}{\int_0^{t_{y_M}} y_M(t, K_M, T_M) \cdot \mathrm{d}t} = \frac{I_{\Delta y_M}(t_{y_M} + \Delta t_{y_M})}{I_{y_M}(t_{y_M})} \tag{4.34}$$

The question arises how to find functions that describe dependence of ρ_{y_M} on $K_M(\Delta K_M)$ and $T_M(\Delta T_M)$ for different values of K_M and T_M (i.e., different values of damping coefficient ξ and natural frequency ω_n), so that we can find inverse functions $\Delta K_M = f(\rho_{y_M})$ and $\Delta T_M = f(\rho_{y_M})$. While the way from σ_m and t_m to ξ and ω_n (K_M and T_M) and time t_{y_M} is straightforward, the way back is not that elegant. Namely, for detected t_{y_M} we do not know the exact values of ξ and ω_n. Therefore, we cannot know the exact values of K_M and T_M as they both depend on ξ and ω_n. Instead, we must use a computer and calculate functions describing dependence of ρ_{y_M} on $K_M(\Delta K_M)$ and $T_M(\Delta T_M)$ obtained by simulation of the reference model (4.20) and by calculation of ρ_{y_M} for different values of K_M

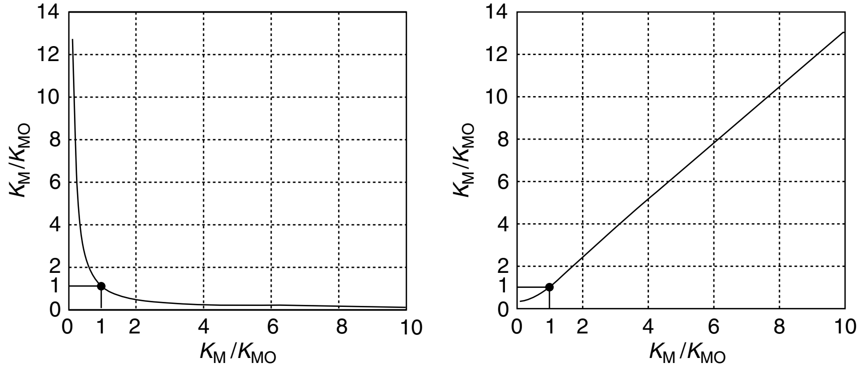


FIGURE 4.35 Relations between performance index ρ_{yM} and model gain coefficient K_M (left) and model time constant T_M (right). (From Kovačić, Z., Bogdan, S., and Punčec, M., *2000 IEEE Intl. Symp. Intell. Ctrl.*, 55–60, 2000. With permission.)

and T_M. These functions are graphically presented in Figure 4.35. One can see that, for example, for $\rho_{y_M} = 0.5$, K_M must be decreased two times in order to reestablish the nominal value $\rho_{y_M} = 1$, that is, the relation $\Delta K_M = f(\rho_{y_M})$ could be interpolated with a hyperbolic function. In the same time, the relation $\Delta T_M = f(\rho_{y_M})$ is almost linear. It must be pointed out that these two relations can be calculated either before (i.e., off-line) or during the startup of the controller (i.e., on-line).

Having ρ_{y_M} calculated, relations $\Delta K_M/K_M = f(\rho_{y_M})$ and $\Delta T_M/T_M = f(\rho_{y_M})$ exactly determine how much K_M and T_M must be changed to enforce a second-order system (4.20) to have $\rho_y = \rho_{y_M} = 1$. In order to be able to use relations derived for the second-order system for adaptation of a suitable high-order closed-loop system, we must replace ΔK_M and ΔT_M with ΔK_f and ΔT_f, and ρ_{y_M} with ρ_y. New forms of relations $\Delta K_f = f(\rho_y)$ and $\Delta T_f = f(\rho_y)$ represent an adaptation law, which determines the sign and magnitude of ΔK_f and ΔT_f, needed to push ρ_y toward the unity value. Because of using relations originally derived for a second-order system, even in the case of nominal system dynamics there will be some error, that is, $|\rho_y - 1| < \varepsilon_\rho$, where ε_ρ is a small positive parameter. The bigger the ε_ρ, the greater will be the influence of the adaptation algorithm in the case of nominal system dynamics.

Likewise in the sensitivity model-based adaptation algorithm, initial values of lead–lag compensator parameters are set to $K_f^0 = 1$ and $T_f^0 = T_M$, while tuning coefficients γ_K and γ_T should provide a smooth convergence of adaptation (recommended range $0 < \gamma_K, \gamma_T \leq 1$).

Example 4.3 Integral criterion-based adaptation of fuzzy controller.

The structure of the adaptive fuzzy control system is shown in Figure 4.36. It consists of a hybrid fuzzy logic controller described in Section 4.1, a lead–lag compensator, a second-order reference model, and an adaptation mechanism [57].

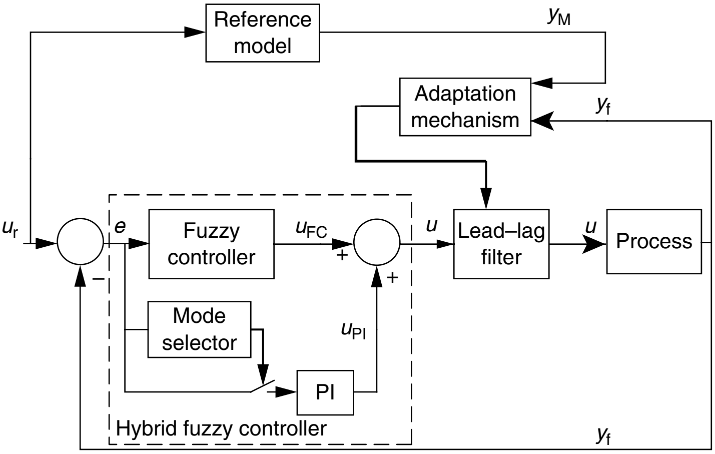


FIGURE 4.36 The structure of an adaptive fuzzy control system. (From Kovačić, Z., Bogdan, S., and Punčec, M., *2000 IEEE Intl. Symp. Intell. Ctrl.*, 55–60, 2000. With permission.)

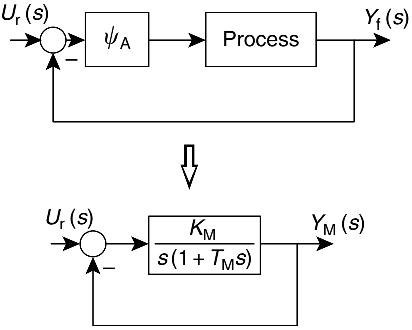


FIGURE 4.37 An adaptive control strategy. (From Kovačić, Z., Bogdan, S., and Punčec, M., *2000 IEEE Intl. Symp. Intell. Ctrl.*, 55–60, 2000. With permission.)

The fuzzy logic controller has two inputs, $e(k) = u_r(k) - y(k)$ and $\Delta e(k) = e(k) - e(k-1)$ and one output calculated according to the center of gravity principle (2.22). The universes of discourse for both input variables are normalized in the range $[-1, 1]$ and split into seven fuzzy sets with triangular, linearly distributed membership functions. The output universe of discourse contains seven linearly distributed singleton sets.

We assume that the process approximation determined in a selected operating point is accurate enough so that a hybrid fuzzy logic controller, designed on the basis of that process approximation, can act on system output response y_f and make it follow reference model response y_M (Figure 4.37).

If we describe a hybrid fuzzy logic controller as a nonlinear mapping function ψ_h, that is, $u(k) = \psi_h[e(k), \Delta e(k)]$, then we are looking for adaptive nonlinear mapping function ψ_{hA}, which would move the closed-loop system dynamics arbitrarily close to the reference model dynamics ($\rho_y \approx \rho_{y_M}$).

The adaptation algorithm is based on model reference-based relations $\Delta K_{\mathrm{f}} = f(\rho_{\mathrm{y}})$ and $\Delta T_{\mathrm{f}} = f(\rho_{\mathrm{y}})$ shown in Figure 4.35, and instead of changing fuzzy logic controller parameters it tunes lead–lag compensator parameters according to tuning law (4.31). The aim of the method is to tune coefficient K_{f} to enforce the open-loop system gain coefficient to converge to reference model gain coefficient K_{M} and to bring up lead-time constant T_{f} to the value that would cancel a dominant process time constant so that system dynamics become primarily determined with reference model time constant T_{M}. Tuning coefficients γ_{K} and γ_{T} are chosen to be equal to one.

The adaptive fuzzy logic controller has been tested by computer simulation in the same linearized model of a PMSM servo system as in Example 4.2. The nominal values of process parameters are the same, only the second-order reference model (4.20) has slightly different given maximum overshoot $\sigma_{\mathrm{m}} = 5\%$ and peak time $t_{\mathrm{m}} = 0.02$ sec. This yields reference model parameter values $K_{\mathrm{M}} = 157.25$ and $T_{\mathrm{M}} = 0.0033$ sec.

Figure 4.38 shows system output and reference model output responses at the beginning of adaptation. The hybrid fuzzy controller parameters are determined for nominal process parameters. As the moment of inertia has been increased five times with respect to its nominal value, the maximal tracking error value (Figure 4.39) at the beginning of the adaptation is very large. System output and reference model output responses obtained after the process of parameter adaptation is finished are shown in Figure 4.40. One can see that the system follows the reference model closely. Tracking error e_{M}, shown in Figure 4.41, is reduced and

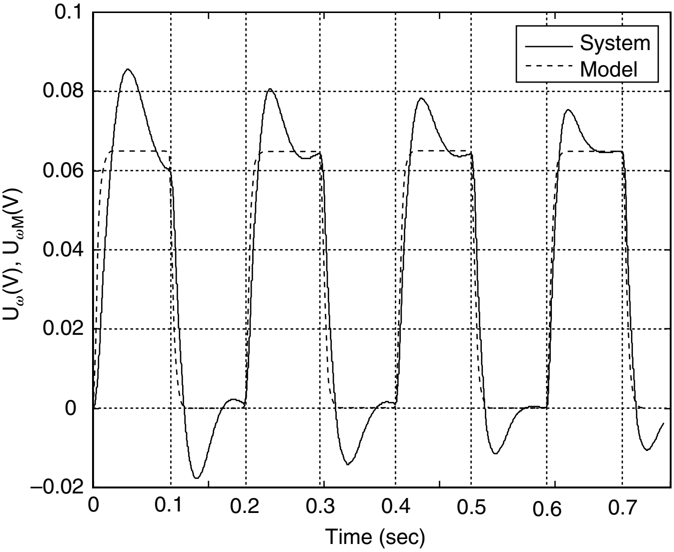


FIGURE 4.38 The process and model responses at the beginning of adaptation, $J = 5 \cdot J_{\mathrm{n}}$. (From Kovačić, Z., Bogdan, S., and Punčec, M., *2000 IEEE Intl. Symp. Intell. Ctrl.*, 55–60, 2000. With permission.)

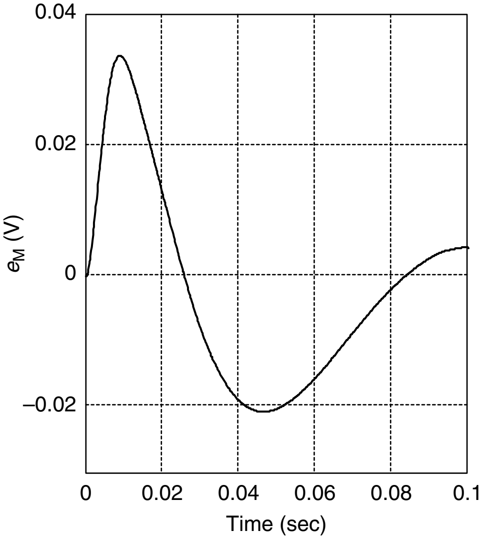


FIGURE 4.39 The tracking error at the beginning of the adaptation, $J = 5 \cdot J_n$. (From Kovačić, Z., Bogdan, S., and Punčec, M., *2000 IEEE Intl. Symp. Intell. Ctrl.*, 55–60, 2000. With permission.)

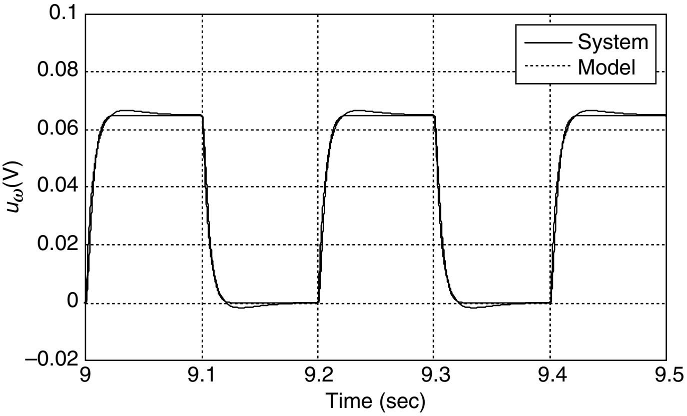


FIGURE 4.40 The process and model responses at the end of the adaptation, $J = 5 \cdot J_n$. (From Kovačić, Z., Bogdan, S., and Punčec, M., *2000 IEEE Intl. Symp. Intell. Ctrl.*, 55–60, 2000. With permission.)

its maximal value is less than 15%. Figure 4.42 and Figure 4.43 show the evolution of controller output from the beginning until the end of adaptation. Convergence of K_f and T_f during the adaptation process is shown in Figure 4.44. Parameters are changing smoothly and they reach stationary values in about seven tuning iterations.

The results obtained confirm efficacy of the integral criterion-based adaptation of the fuzzy controller by adapting parameters of the added lead–lag compensator.

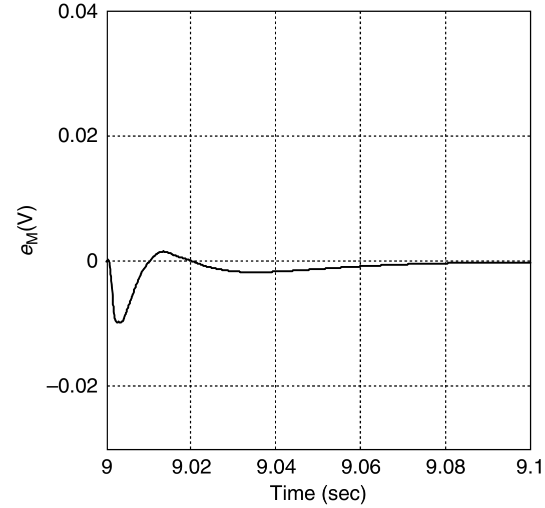


FIGURE 4.41 The tracking error at the end of the adaptation, $J = 5 \cdot J_n$. (From Kovačić, Z., Bogdan, S., and Punčec, M., *2000 IEEE Intl. Symp. Intell. Ctrl.*, 55–60, 2000. With permission.)

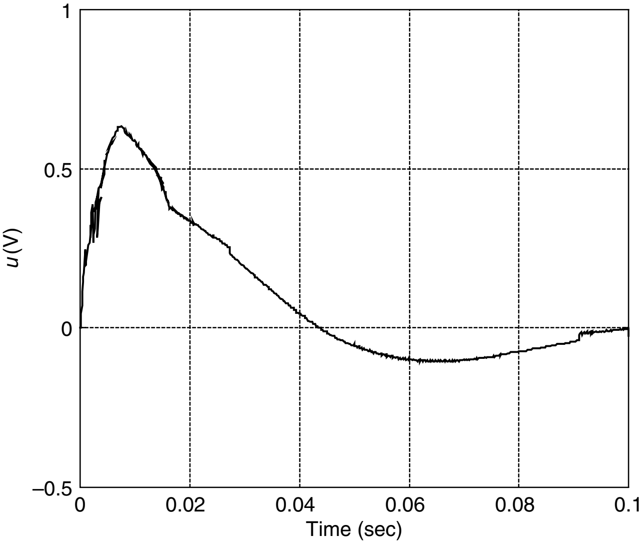


FIGURE 4.42 The controller output at the beginning of the adaptation, $J = 5 \cdot J_n$. (From Kovačić, Z., Bogdan, S., and Punčec, M., *2000 IEEE Intl. Symp. Intell. Ctrl.*, 55–60, 2000. With permission.)

4.2.2.3 Model Reference Adaptive Control with Fuzzy Adaptation

The MRAC schemes compare responses of controlled system variables with responses of accompanying reference model variables and then use the consequent

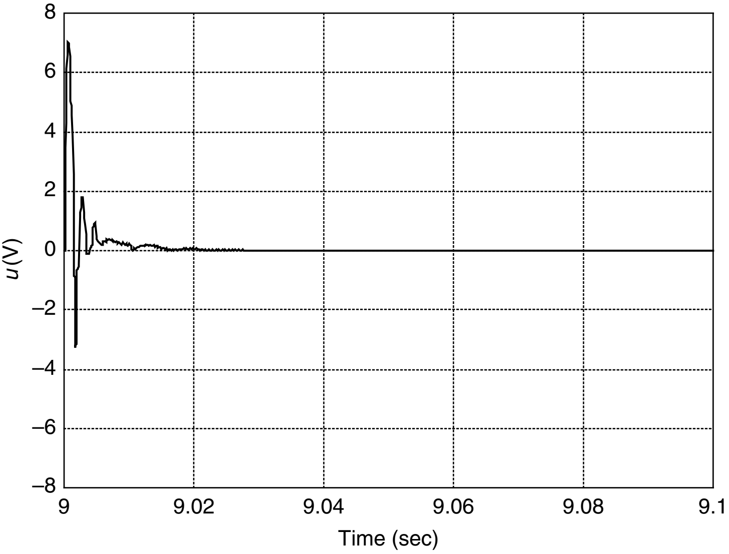


FIGURE 4.43 The controller output at the end of the adaptation, $J = 5 \cdot J_n$. (From Kovačić, Z., Bogdan, S., and Punčec, M., *2000 IEEE Intl. Symp. Intell. Ctrl.*, 55–60, 2000. With permission.)

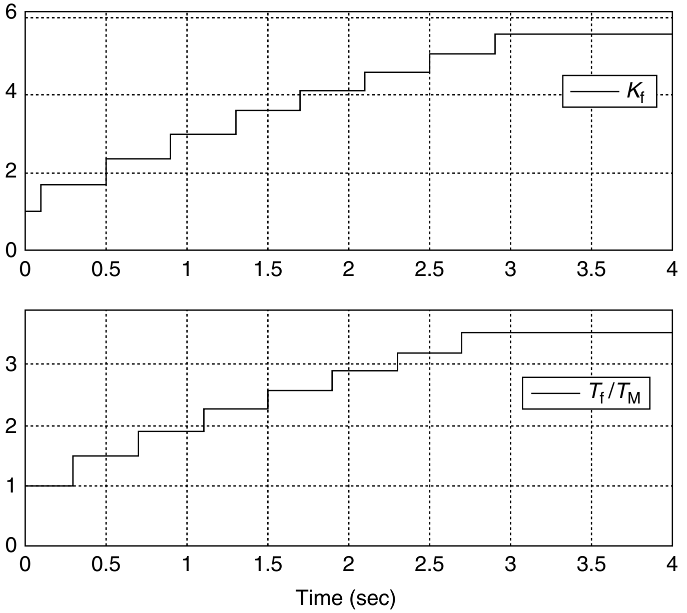


FIGURE 4.44 Convergence of lead–lag compensator gain coefficient K_f and lead-time constant T_f, $J = 5 \cdot J_n$.

error signals as inputs to an adaptation mechanism. Adaptation mechanisms are dominantly nonlinear devices, which may generate an additional signal (signal adaptation) or change basic controller parameters (parameter adaptation). Signal adaptation is preferable because of the high speed of adaptation. On the other hand, signal adaptation causes stationary oscillations in the system response. These oscillations may be eliminated by modifying a signal adaptation algorithm, as described in Reference 58, but such modifications cause a steady-state error.

A full-order reference model can provide the best effectiveness of the adaptation mechanism, but a reduced-order reference model is usually preferred because of its simpler design and implementation. Very often a second-order reference model is used to determine the desired dynamic characteristics of the high-order system. In such cases, unmodelled dynamics and disturbances may have an undesired influence on the convergence of parameters and the stability of the system. This can be solved by modifying the parameter-adaptation algorithms, but such modifications also cause a steady-state error [59].

In this section, we describe a design of a model reference adaptive fuzzy control algorithm, which has a high speed of adaptation and produces no oscillations in the steady state. An adaptation signal is added directly to the feedback controller input (see Figure 4.17).

The fuzzy adaptation mechanism has a PD-type character and its role is to tune the closed-loop control system input so that the system behaves like the reference model. The inputs of the fuzzy adaptation mechanism are model tracking error e_{M} and change of error Δe_{M}. The structure of the fuzzy adaptation algorithm is the same as the structure of a standard fuzzy controller, so adaptation signal u_{A} is obtained in the same manner as the output of a standard fuzzy controller. This means that the design goal is to synthesize fuzzy control rules, which would map observed process response deviation Δy_{f} (equal to e_{M} in the MRAC approach) and its change to adequate changes in the system input:

$$u_{\mathrm{rA}} = u_{\mathrm{r}} + u_{\mathrm{A}} \tag{4.35}$$

where u_{rA} is a modified (adapted) reference input.

The procedure of setting the fuzzy rule table of adaptation algorithm is completely heuristic. This may seem as a drawback, but may also be an advantage. Namely, in many practical situations a control designer knows how certain parameter changes affect the closed-loop system performance. Moreover, if by chance the mathematical model of the process is known, like in the electrical drives and servo control applications described in References 60–64, then using the computer and running simulations can help a lot with fast setup of fuzzy rules. Interpretation of the relation between desired system response changes and necessary system input changes is also referred to as a fuzzy inverse model [65,66]. Here, we refer to it as an auxiliary controller, or simply, an adaptation mechanism.

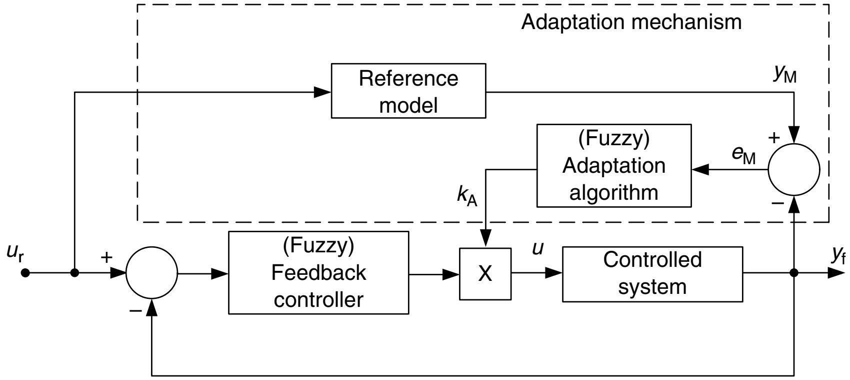


FIGURE 4.45 The structure of a fast parameter adaptation fuzzy MRAC system.

Since the structure of the adaptation algorithm resembles the structure of the standard DISO fuzzy controller, the design steps are also the same. Although a particular design procedure will depend on a particular application, the job to be done will not be any more complicated than the design of a standard fuzzy controller. First, we must define suitable values of input and output scaling factors for e_M, Δe_M, and u_A. Wrong selection of scaling factors may result in a poor performance of the adaptation mechanism or in excessive value of the adaptation signal, which may lead to the saturation of control input. We have mentioned that signal adaptation can be imposed either at the input or at the output of the feedback controller. The place of adaptation signal addition will affect the values of scaling factors, as we normally must account for the gain of the feedback controller.

A special case is adaptation at the output of the feedback controller accomplished by gain multiplication, as shown in Figure 4.45. This structure is actually a hybrid one, featuring fast parameter adaptation in the manner of signal adaptation by direct change of the adaptation gain k_A. Such a concept assumes that the steady-state value of k_A is equal to one. The effectiveness of the fuzzy adaptation mechanism depends on correct estimation of the possible range of variations for each of the parameters, and on the ability of the system to accept the newly formed controller output signal without limitations related to the finite capacity of the energy resources.

The experience with some control systems affected by large parameter variations indicates that the fuzzy rule table of the adaptation algorithm usually assumes a nonsymmetrical form. This stems from the fact that a different amount of signal (parameter) correction is needed to compensate for the same amount of increasing or decreasing parameter variations, especially when more than one parameter varies at the time.

The role of the fuzzy adaptation algorithm is to stay idle in the steady state and get active during system transitions. In order to circumvent steady-state oscillations caused by constant activity of the adaptation signal, the choice of the trapezoidal

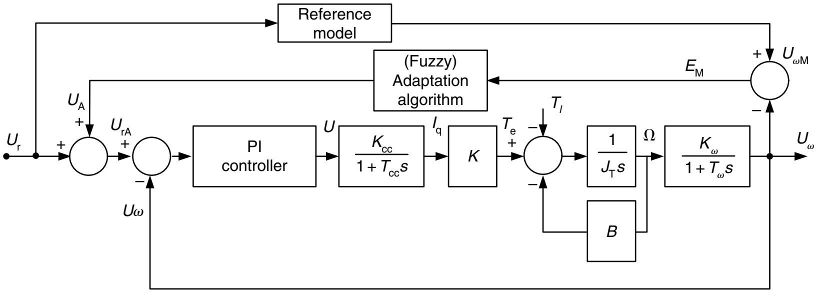


FIGURE 4.46 The block scheme of the studied angular speed fuzzy MRAC system.

form of the most inner fuzzy sets covering small values of e_M and Δe_M is recommended. The size of the nuclei of these fuzzy sets will dictate the width of the dead-zone, which will pacify the adaptation algorithm in the steady state.

Example 4.4 MRAC with fuzzy signal adaptation.

In this example, we describe a design of a fuzzy model reference adaptive control scheme that contains a commonly used PI feedback controller and a fuzzy adaptation mechanism, which generates an adaptation signal added to the input of the PI controller [67]. A second-order reference model is used to describe the desired performance of the closed-loop system. Such a structure can be classified as the one represented in Figure 4.17.

The target system for which the design is carried out is an angular speed control loop of a vector controlled chopper-fed PMSM described in detail in Example 4.1. Instead of using classical design methods (e.g., by using Lyapunov functions) suitable for processes with known models, we demonstrate the heuristic design of a fuzzy adaptation mechanism without knowing the exact model of the process (care must be taken that nominal dynamics defined by the reference model match the feasible dynamics of the controlled process).

In this case the studied system would have a transfer function of a third-order system (4.13). A linearized model of the angular speed control system is shown as a part of the whole fuzzy adaptive control system in Figure 4.46.

As discussed in Example 4.1, torque coefficient K and moment of inertia J_T are parameters of the target control system that vary the most.

The PI controller parameters were specified according to the symmetrical optimum criterion, which gives 40% of overshoot in the output response, all in order to obtain quick compensation of load torque variations. A lower overshoot (in our case 20%) in response to the reference input changes may be achieved by adding an appropriate filter into the reference input signal path. The rated values of linearized model parameters (Figure 4.46) were as follows: $K_{cc} = 1$ A/V, $T_{cc} = 50\ \mu$sec, $K = 0.9837$ Vsec, $J_T = 0.00176\ \text{kg m}^2$, $B = 0.000388$ Nmsec, $T_M = J_T/B = 4.536$ sec, $K_\omega = 0.063$ Vsec, and $T_\omega = 2.5$ msec.

In the case being studied, the desired behavior of the target high-order system is represented by a reference model of a second-order system. The reference model is described by means of a transfer function, which is related to the rated parameter values

$$G_{\mathrm{M}}(s) = \frac{U_{\omega\mathrm{M}}}{U_{\mathrm{r}}(s)} = \frac{1}{(T_{\mathrm{p}}/K_{\mathrm{p}})s^2 + (1/K_{\mathrm{p}}) + 1} \tag{4.36}$$

where $U_{\omega\mathrm{M}}$ is a reference model output (desired feedback signal), $K_{\mathrm{p}} = K_{\mathrm{PI}}K_{\mathrm{cc}}KK_{\omega}/T_{\mathrm{PI}}B$ and $T_{\mathrm{p}} = T_{\omega}$.

The robustness of PI controllers to parameter variations is rather weak, especially in cases of large parameter variations, as it was shown for the target control system in Figure 4.6 (see Example 4.1).

The fuzzy adaptation algorithm has a PD-type character, that is, the current adaptation signal value does not depend on previous output values. The inputs of the fuzzy adaptation algorithm are tracking error, e_{M}, and its change, Δe_{M}. Therefore, the synthesis of the fuzzy adaptation algorithm is very similar to the synthesis of a PD-type DISO fuzzy controller. The sign and magnitude of adaptation signal u_{A} are determined by the signs and magnitudes of inputs e_{M} and Δe_{M} via a fuzzy mapping function.

Seven linguistic subsets are defined for both adaptation algorithm inputs (universes of discourse EM and DEM): NL, NM, NS, Z, PS, PM, and PL. The design goal was to generate adaptation signal u_{A}, which would keep model tracking error e_{M} within ±10% of the imposed change of reference input. Therefore, the maximum error value was set in the range $-0.0063 \leq e_{\mathrm{M}} \leq 0.0063$. The maximum change of error during control interval T_{d} was estimated from the nominal dynamics and it was in the range $-0.0009 \leq \Delta e_{\mathrm{M}} \leq 0.0009$. In general, a nominal system model and some simulation tools such as MATLAB can be very helpful in determination of correct maximal input values.

The idea of the fuzzy adaptation mechanism design is to create a set of fuzzy control rules that would continuously issue the signal required for the adjustment of a PI controller input, in order to ensure that the transient response is minimally affected by parameter variations. Being further away from the nominal dynamics requires stronger adaptation effort. On the other hand, being closer to the nominal dynamics requires finer adaptation. Following that reasoning, we may choose a trapezoidal form of the outer fuzzy sets and a triangular form of the inner sets. An exception is a trapezoidal form of the zero sets, which is there to define a zero zone of u_{A} in the case when both inputs e_{M} and Δe_{M} are very close to zero. Thanks to that zone, the adaptation algorithm will stay passive during steady-state conditions. The normalized distributions of defined membership functions along e_{M} and Δe_{M} axes are shown in Figure 4.47. In the presented design, the distributions are symmetrical with respect to zero sets ZEM and ZDEM and have the same form for both inputs. The way in which the membership functions overlap ensures that not more than 4 out of 49 possible IF–THEN control rules may contribute to the adaptation

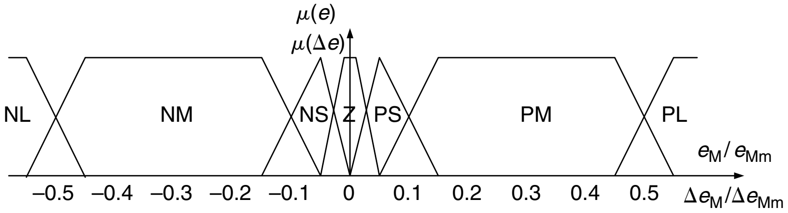


FIGURE 4.47 Normalized distributions of membership functions for both inputs of the fuzzy adaptation algorithm.

TABLE 4.2
The Fuzzy Rule Table of the Fuzzy Adaptation Algorithm

	NLEM	NMEM	NSEM	ZEM	PSEM	PMEM	PLEM
NLDEM	−0.1	−0.06	−0.04	−0.02	−0.015	−0.01	0
NMDEM	−0.06	−0.04	−0.02	−0.015	−0.01	0	0.0075
NSDEM	−0.04	−0.02	−0.015	−0.0075	0	0.0075	0.055
ZDEM	−0.02	−0.015	−0.0075	0	0.0075	0.055	0.15
PSDEM	−0.015	−0.0075	0	0.0075	0.055	0.15	0.3
PMDEM	−0.01	0	0.0075	0.055	0.15	0.3	0.4
PLDEM	0	0.01	0.055	0.15	0.3	0.4	0.8

signal value. The signal value is calculated according to the center of gravity principle (2.22).

The fuzzy rule table of the adaptation algorithm was defined heuristically and the values of singletons are shown in Table 4.2. In the studied case, one may see in Figure 4.48 that the adaptation signal domain contains 15 nonlinearly distributed singletons, with a dominant right-hand side, which enforces eight times higher adaptation effort for the lagging of system dynamics than for the leading. Although exact singleton values are dependent on the studied target system, from their relative values and positions in the universe of discourse, one can get an idea on how to approach other control design problems.

In the case of nominal parameter values, Figure 4.49 shows the angular speed responses of the nonadaptive PMSM drive, adaptive PMSM drive, and a reference model. A certain difference between the closed-loop system and the reference model responses was expected even with nominal parameter values, but the adaptation mechanism was very effective and the adaptive system closely follows the model.

Simulated moderate variations of moment of inertia, $J_T/2$ and $3J_T/2$, were almost fully compensated by the adaptation algorithm, as proved by the responses shown in Figure 4.50 and Figure 4.51, respectively. The tracking error shown in

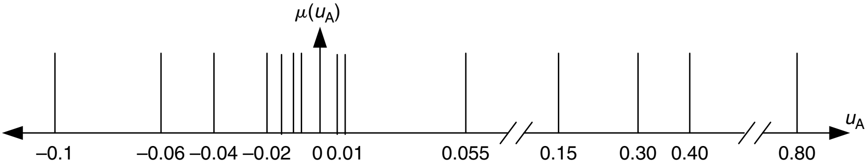


FIGURE 4.48 Distribution of singletons along the adaptation signal universe of discourse.

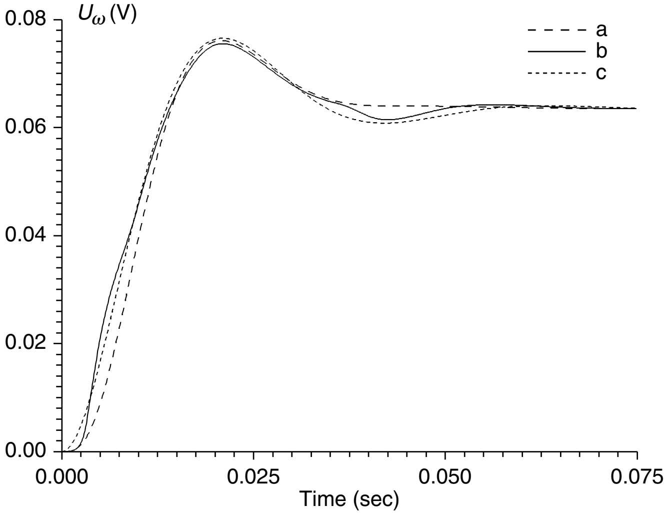


FIGURE 4.49 The measured angular speed responses in case of nominal parameters: nonadaptive PMSM drive (a), adaptive PMSM drive (b), and reference model (c). (From Kovačić, Z., Bogdan, S., and Crnošija, P., *19th Annu. Conf. IEEE Industrial Electr. Soc.*, 1, 207–212, 1993. With permission.)

Figure 4.52 and Figure 4.53 was kept within ±10% of the imposed change of the reference input, thus fulfilling the main design goal. Figure 4.54 shows the waveforms of adaptation signal u_A for both cases. One can notice that the shape and the magnitude of u_A are very suitable for applications of real systems.

Let us check the effectiveness of adaptation for very large variations of J_T equal to $J_T/3$ and $3J_T$. Figure 4.55 and Figure 4.56 show the angular speed responses of the nonadaptive and adaptive PMSM drive, while Figure 4.57 and Figure 4.58 show the tracking error responses for both cases. As may be seen, the adaptation mechanism still noticeably improves the quality of the angular speed responses, but tracking error e_M has exceeded the desired range, thus indicating the imperfection of the designed adaptation algorithm for large parameter variations. This would require intervention into some of the design steps, but it must be noted that the simulated change of process parameter is really very large and therefore rare in most practical control (e.g., servo) applications [68].

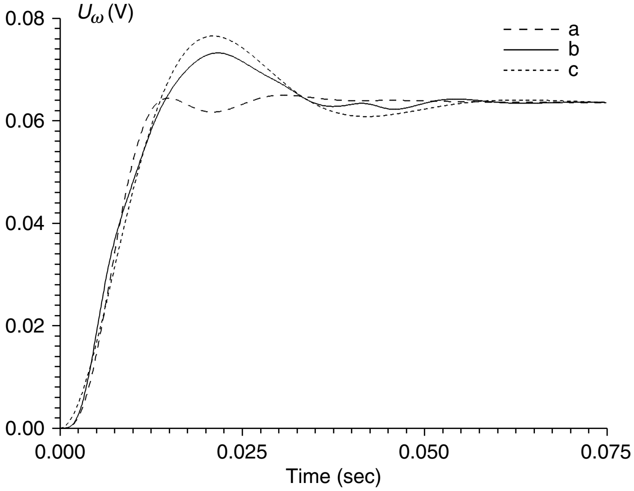


Figure 4.50 The measured angular speed responses in the case of $J_T/2$: nonadaptive PMSM drive (a), adaptive PMSM drive (b), and reference model (c). (From Kovačić, Z., Bogdan, S., and Crnošija, P., *19th Annu. Conf. IEEE Industrial Electr. Soc.*, 1, 207–212, 1993. With permission.)

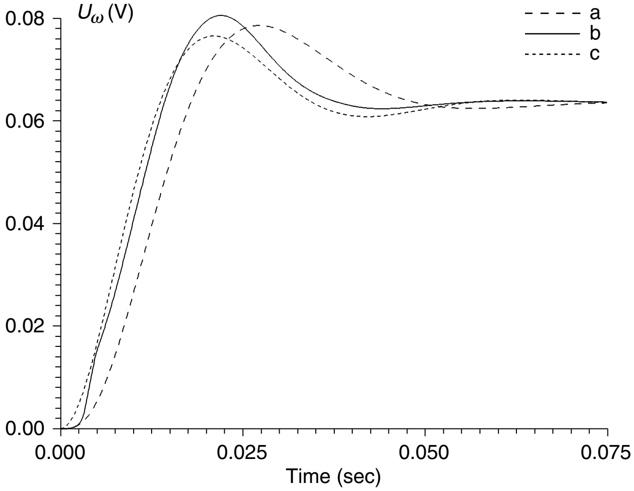


Figure 4.51 The measured angular speed responses in the case of $3J_T/2$: nonadaptive PMSM drive (a), adaptive PMSM drive (b), and reference model (c). (From Kovačić, Z., Bogdan, S., and Crnošija, P., *19th Annu. Conf. IEEE Industrial Electr. Soc.*, 1, 207–212, 1993. With permission.)

Figure 4.59 shows the PMSM drive response in the case of stepwise change of load torque τ_l (active disturbance), $\Delta\tau_l = \tau_n/100$, where τ_n is a nominal motor torque. Although the PI controller parameters were obtained according to the symmetrical optimum criterion, that is, with the best ability to compensate

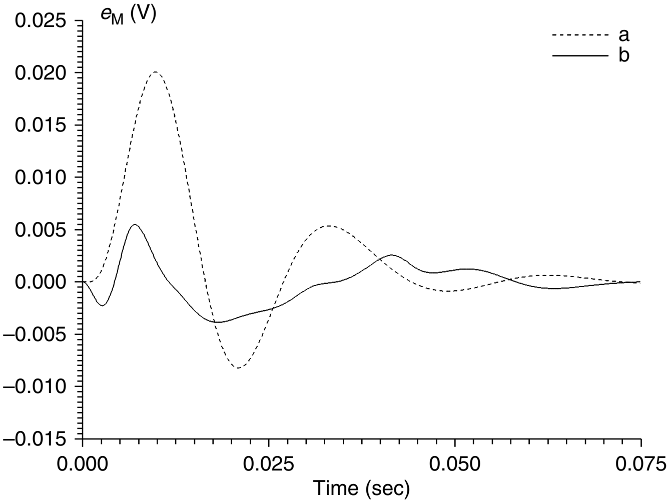


FIGURE 4.52 The tracking error responses in the case of $J_T/2$: nonadaptive PMSM drive (a), adaptive PMSM drive (b). (From Kovačić, Z., Bogdan, S., and Crnošija, P., *19th Annu. Conf. IEEE Industrial Electr. Soc.*, 1, 207–212, 1993. With permission.)

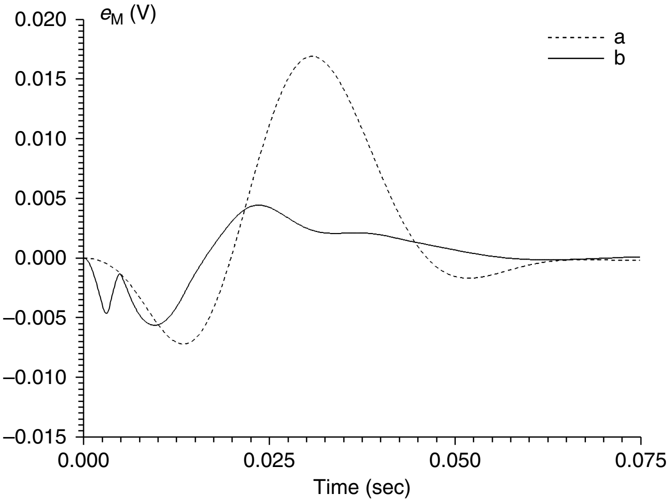


FIGURE 4.53 The tracking error responses in the case of $3J_T/2$: nonadaptive PMSM drive (a), adaptive PMSM drive (b). (From Kovačić, Z., Bogdan, S., and Crnošija, P., *19th Annu. Conf. IEEE Industrial Electr. Soc.*, 1, 207–212, 1993. With permission.)

for disturbance effects, a drop in the angular speed response is much lower in the case of adaptive control. The oscillations in the response are caused by the way of designing the fuzzy rule table for the target system. If the adaptation algorithm is supposed to act equally well in case of the reference input changes

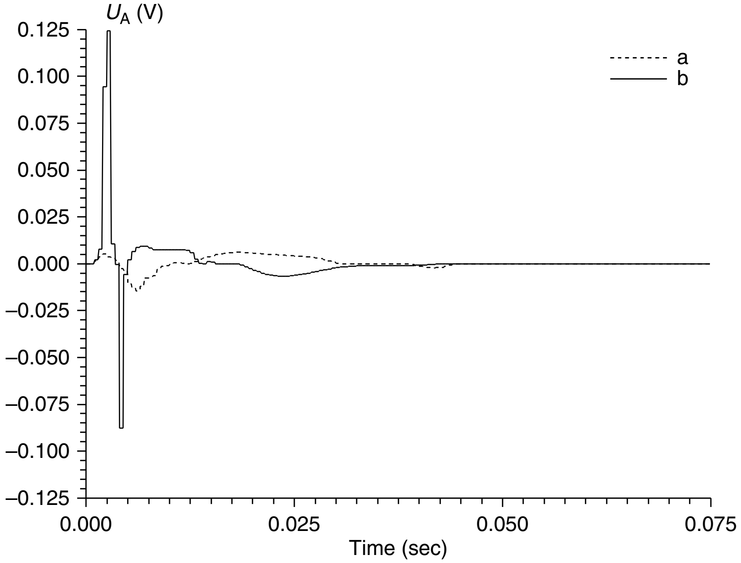


FIGURE 4.54 The adaptation signal responses in the case of $J_T/2$ (a), $3J_T/2$ (b). (From Kovačić, Z., Bogdan, S., and Crnošija, P., *19th Annu. Conf. IEEE Industrial Electr. Soc.*, 1, 207–212, 1993. With permission.)

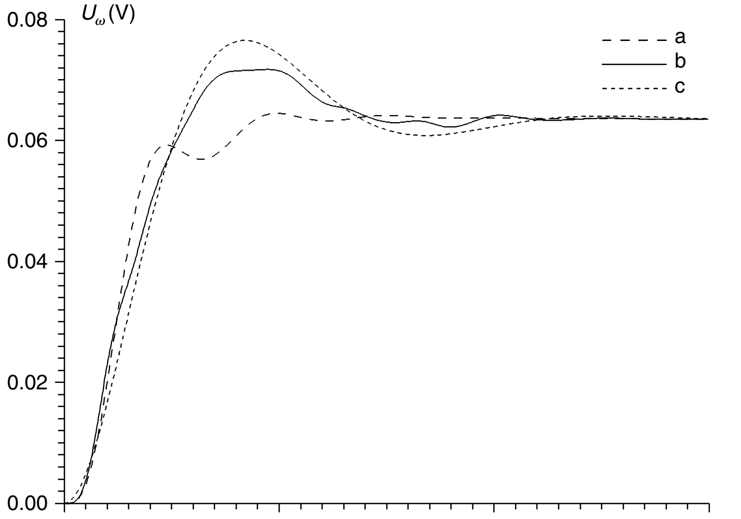


FIGURE 4.55 The measured angular speed responses in the case of $J_T/3$: nonadaptive PMSM drive (a), adaptive PMSM drive (b), and reference model (c). (From Kovačić, Z., Bogdan, S., and Crnošija, P., *19th Annu. Conf. IEEE Industrial Electr. Soc.*, 1, 207–212, 1993. With permission.)

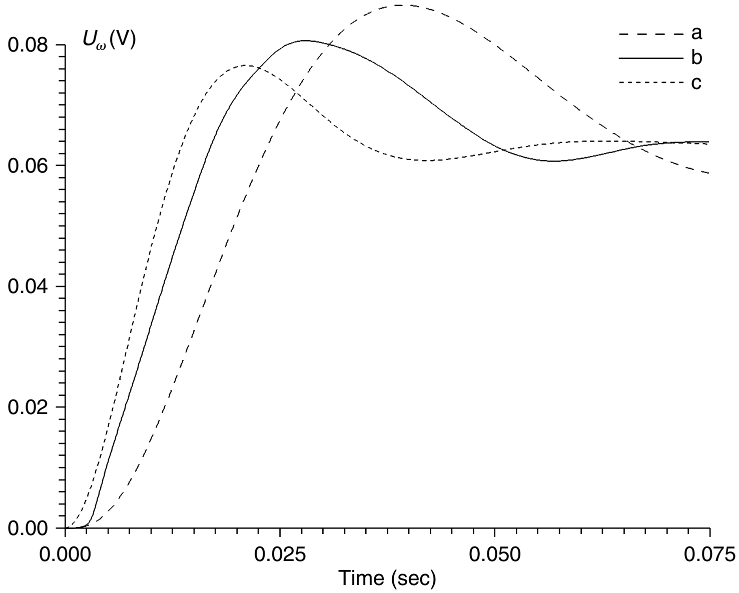


FIGURE 4.56 The measured angular speed responses in the case of $3J_T$: nonadaptive PMSM drive (a), adaptive PMSM drive (b), and reference model (c). (From Kovačić, Z., Bogdan, S., and Crnošija, P., *19th Annu. Conf. IEEE Industrial Electr. Soc.*, 1, 207–212, 1993. With permission.)

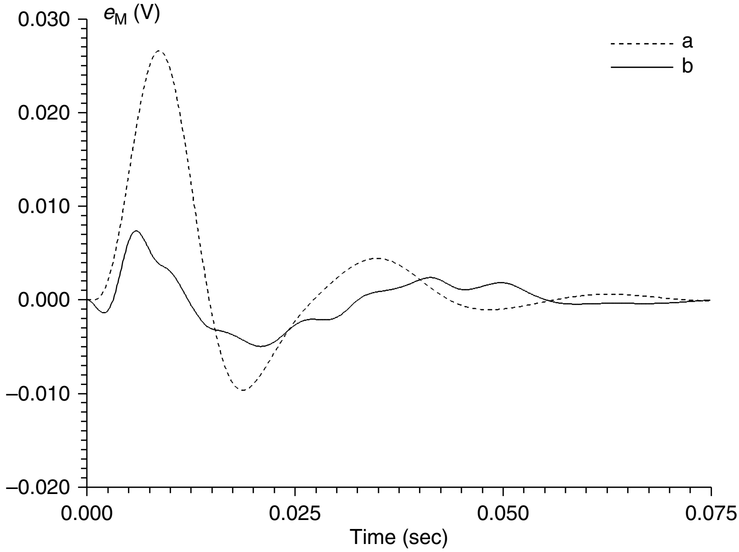


FIGURE 4.57 The tracking error responses in the case of $J_T/3$: nonadaptive PMSM drive (a) and adaptive PMSM drive (b). (From Kovačić, Z., Bogdan, S., and Crnošija, P., *19th Annu. Conf. IEEE Industrial Electr. Soc.*, 1, 207–212, 1993. With permission.)

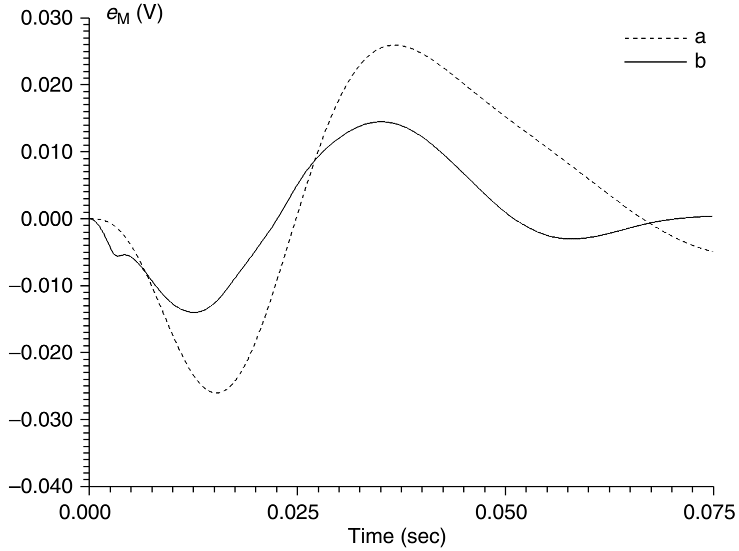


FIGURE 4.58 The tracking error responses in the case of $3J_T$: nonadaptive PMSM drive (a) and adaptive PMSM drive (b). (From Kovačić, Z., Bogdan, S., and Crnošija, P., *19th Annu. Conf. IEEE Industrial Electr. Soc.*, 1, 207–212, 1993. With permission.)

and the disturbance changes, then a sort of trade-off should be included in the design process, thus consciously narrowing the range of good adaptation.

A heuristic design of the fuzzy adaptation algorithm practically ensures the stable performance of the adaptive PMSM drive for all reasonable and predictive system parameter variations. On the other hand, the speed of adaptation is related to selected control interval T_d. In the studied PMSM drive, significantly decreased moments of inertia create unbalanced relation between the adaptation mechanism and the system dynamics, and simulations indicate an unstable system operation for moments of inertia below $J_T/5$. In other words, if parameter variations exceed the expected range of variations, this would finally produce instability of the system. Therefore, a possible range of parameter variations must be properly estimated and thereafter included in the design steps of the fuzzy adaptation algorithm.

The efficacy of adaptation could be determined through assessment of the relative tracking error value during the transient response. Thus, for 50% changes of J_T, it reaches 10%, while for 300% changes of J_T, it reaches 25% of the imposed change of the reference input. For a 300% increase of J_T, the maximal value of adaptation signal u_A is two and a half times higher than the imposed change of the reference input, while the output of the adaptive PI controller is four times larger than in the nonadaptive case.

In the case of very large parameter variations the effectiveness of the fuzzy adaptation algorithm is lower, partly because the range of adaptation signal should be wider and partly because the adaptation signal is transferred to the process through the PI controller, which possesses high sensitivity to parameter variations.

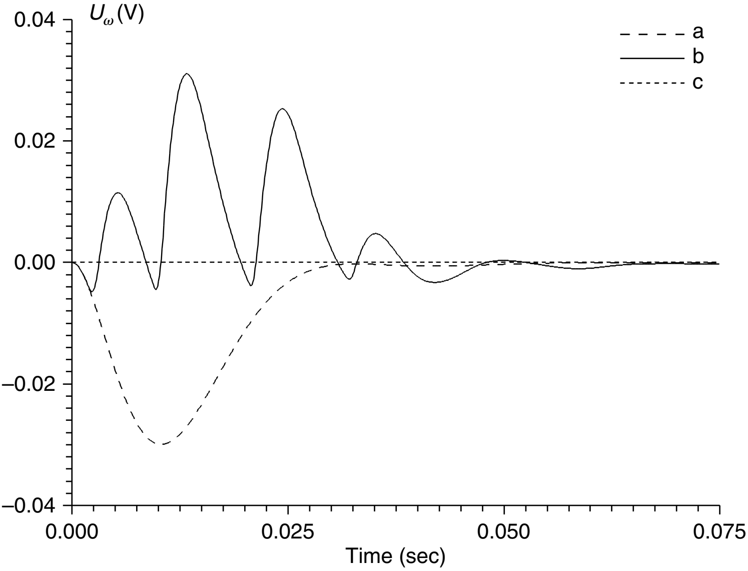


FIGURE 4.59 The PMSM angular speed responses in the case of active disturbance (load torque). (From Kovačić, Z., Bogdan, S., and Crnošija, P., *19th Annu. Conf. IEEE Industrial Electr. Soc.*, 1, 207–212, 1993. With permission.)

It must be noted that simulations have not included the effects of the posed limits which are inevitably present in real control applications.

Example 4.5 MRAC with fuzzy parameter adaptation.

In Example 4.1, we have shown that increased robustness of the hybrid fuzzy controller resulted from the way it has been designed. We have also shown that in the case of large parameter variations, fuzzy controllers, like conventional controllers, cannot provide even system responses without the addition of an adaptation mechanism. Having in mind a completely heuristic fuzzy controller design, the knowledge about the target control system can be very helpful during the development of an adaptive fuzzy control algorithm.

In this example, we describe a design of a MRAC scheme that contains a hybrid fuzzy controller described in Section 4.1, a second-order reference model, and a fuzzy adaptation mechanism [34,69]. The target system is the same as in the previous example, that is, we deal with control of the angular speed of a vector-controlled PMSM drive.

Adaptive tuning is performed by multiplying the hybrid fuzzy controller output with tuning coefficient k_A. Due to the nonintegral character of the fuzzy adaptation algorithm, such a parameter adaptation has the speed of a signal adaptation. Such a control structure shown in Figure 4.60 is referring to the control structure shown in Figure 4.45. This is a nonlinear control scheme because of the nonlinear nature of fuzzy control and adaptation algorithms (they belong to the group of PD-type fuzzy algorithms).

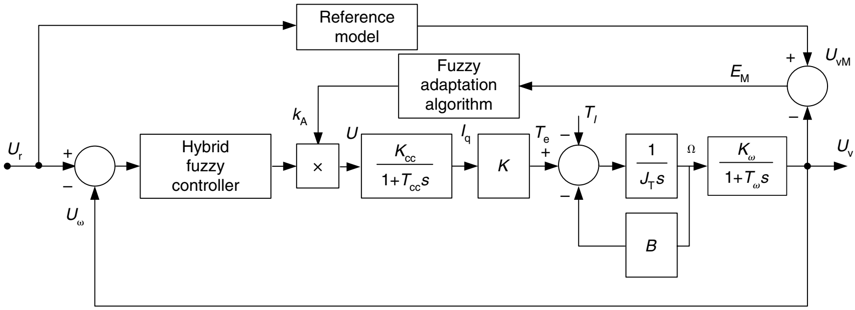


FIGURE 4.60 A block scheme of a studied angular speed fuzzy MRAC system.

In principle, the design of a parameter generating fuzzy adaptation algorithm is very similar to the design of the signal generating fuzzy adaptation algorithm presented in Example 4.4. Here we have to define a set of fuzzy control rules, which would continuously generate the tuning coefficient values required for the adjustment of a hybrid fuzzy controller output. In order to stay passive in the steady state, the steady-state output of the adaptation algorithm should be equal to one, $k_A = 1$.

A fuzzy adaptation mechanism is triggered by its inputs, tracking error e_M and its change Δe_M. The choice of inputs scaling factors depends on correct estimation of the possible range of variations for each of the parameters and desired control quality. In our case we want to keep the tracking error within ±10% of the system output change, that is, $e_M \leq \pm 0.1 \Delta u_\omega$. The effectiveness of the fuzzy adaptation algorithm will depend on the ability of the system to accept the newly formed controller output signal without reaching physical limits.

The fuzzy adaptation algorithm has seven linguistic subsets defined for both inputs: NL, NM, NS, Z, PS, PM and PL. The maximum value of error e_M is estimated to be 0.036, that is, $-0.036 \leq e_M(k) \leq 0.036$. The maximum change of error Δe_M during sampling interval $T_d = 0.5$ msec is estimated to 0.016, that is, $-0.016 \leq \Delta e_M(k) \leq 0.016$. The distribution of membership functions for normalized subsets of e_M and Δe_M are shown in Figure 4.60. The size of the input fuzzy subsets has a direct impact on the characteristics of the adaptation mechanism. In other words, the size of the zero subsets determines the width of the unity tuning coefficient zone, which guarantees steady-state passivity of the adaptation mechanism. As can be seen from Figure 4.61, for both inputs the density of membership functions is increasing toward the origin, thus providing the desired fineness of adaptation. On the other hand, robustness and roughen for cement during the more demanding transient events in the system are achieved by choosing the trapezoidal forms of the membership functions, which are related to the medium and large input values.

The distribution of fuzzy output subsets is nonlinear because this has been directed by the characteristics of the target system. The corresponding centroids have the following normalized values ($\max(k_A) = 7$): −0.043, −0.029, −0.007,

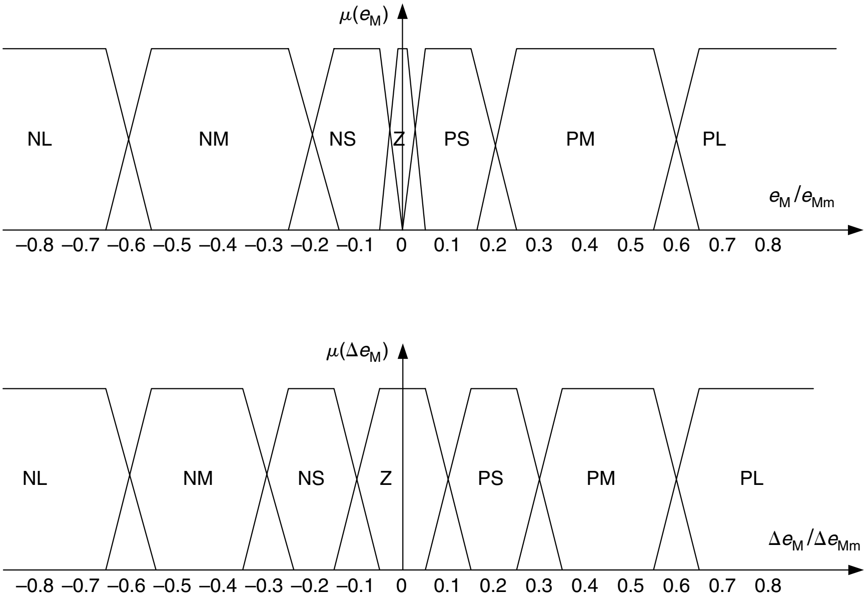


FIGURE 4.61 Distribution of membership functions for a fuzzy adaptation algorithm: (up) $\mu[e_M(k)]$, (down) $\mu[\Delta e_M(k)]$.

TABLE 4.3
The Fuzzy Rule Table of an Adaptation Mechanism

	NLE	NME	NSE	ZE	PSE	PME	PLE
NLDE	1	1	0.667	0.5	0.333	0.0083	0
NMDE	1	1	0.667	0.167	0.0083	−0.0042	−0.0083
NSDE	1	0.667	0.167	0.0084	0.0042	−0.333	−0.5
ZDE	0.833	0.333	0.0083	0	0	−0.333	−0.833
PSDE	0.667	0.333	0	−0.0083	−0.167	−0.667	−1
PMDE	0.0083	−0.0083	−0.167	−0.333	−0.667	−1	−1
PLDE	−0.0083	−0.167	−0.333	−0.5	−0.667	−1	−1

0.007, 0.036, 0.071, 0.107, 0.129, 0.143, 0.157, 0.214, 0.25, 0.321, 0.428, 0.571, 0.714, and 1. The size and distribution of the input and output fuzzy subsets were defined by trial and error in order to keep the tracking error within specified limits: $|e_M| \leq 0.1\Delta u_{\omega}$.

The adaptation mechanism output, that is, tuning coefficient k_A is computed according to the center of gravity principle (2.22). The fuzzy rule table of the adaptation mechanism is shown in Table 4.3.

The proposed adaptive fuzzy controller was tested by simulation experiments for cases of moderate and very large changes of inertial moment J_T. As shown in

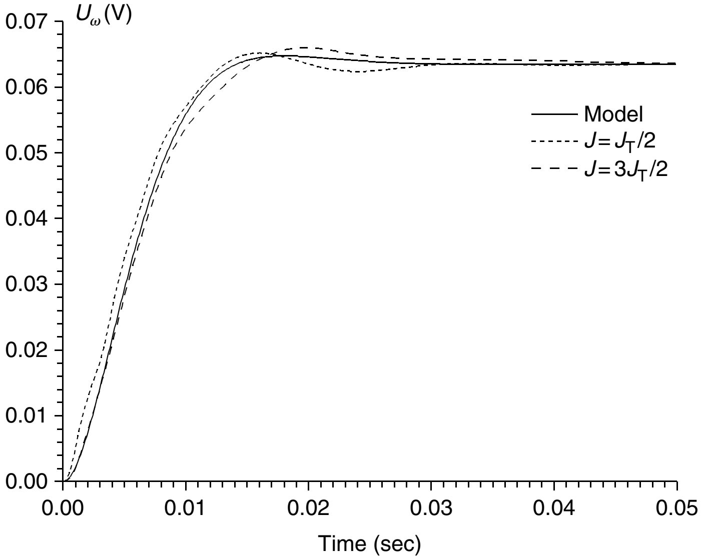


FIGURE 4.62 The measured angular speed responses of the adaptive fuzzy control system for moderate changes of J_T. (From Kovačić, Z. and Bogdan, S., *Eng. Appl. Artif. Intelligence*, 7(5), 501–511, 1994. With permission from Elsevier.)

Figure 4.62, for moderate changes of inertial moment equal to ±50% of the rated value (i.e., $J_T/2$ and $3J_T/2$), the adaptation mechanism has noticeably improved the quality of measured angular speed responses, that is, the quality is much better than in the nonadaptive case (Figure 4.9). In cases of very large changes of the moment of inertia ($J_T/3$, $3J_T$), as shown in Figure 4.63, compared to the nonadaptive case (Figure 4.10), the quality of measured angular speed responses, with regard to basic system dynamics, is kept almost unchanged, while the tracking error is within given limits of 10% of the imposed change of reference input (Figure 4.64).

The responses of other system variables will be given only for large changes of the moment of inertia. Figure 4.65 shows the output of the fuzzy adaptation mechanism, that is, tuning coefficient k_A that multiplies the hybrid fuzzy controller output. For $3J_T$, k_A reaches the maximum, $k_{Am} = 2.8$, while for $J_T/3$, it assumes values less than one. It may be noticed that the tuning coefficient can assume even negative values for the smallest value of the moment of inertia to provide for a possibility of faster braking if it proves necessary. The tuning coefficient assumes a unity value before and after the transient response, thanks to the zero action zone, which guarantees that the adaptation mechanism does not affect steady-state accuracy. This was intentionally built into the fuzzy rule table of the fuzzy adaptation mechanism. The responses of the adaptive fuzzy controller output are shown in Figure 4.66. The fuzzy controller leads in action at the beginning of the transient response, thus creating the initial rate of response. Then the influence of

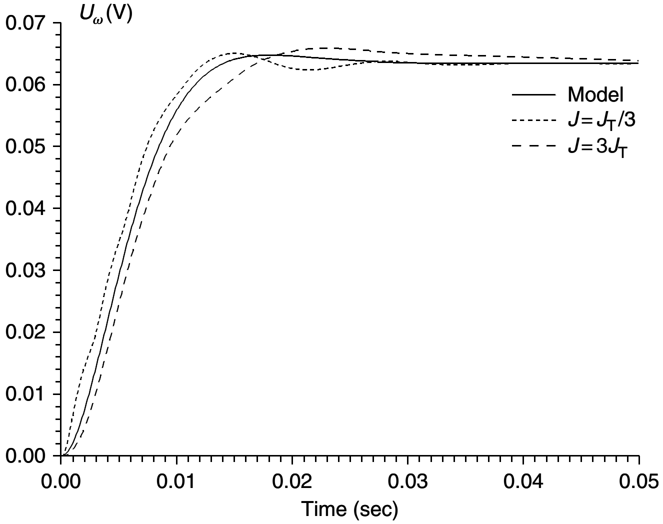


FIGURE 4.63 The measured angular speed responses of the adaptive fuzzy control system for large changes of J_T. (From Kovačić, Z. and Bogdan, S., *Eng. Appl. Artif. Intelligence*, 7(5), 501–511, 1994. With permission from Elsevier.)

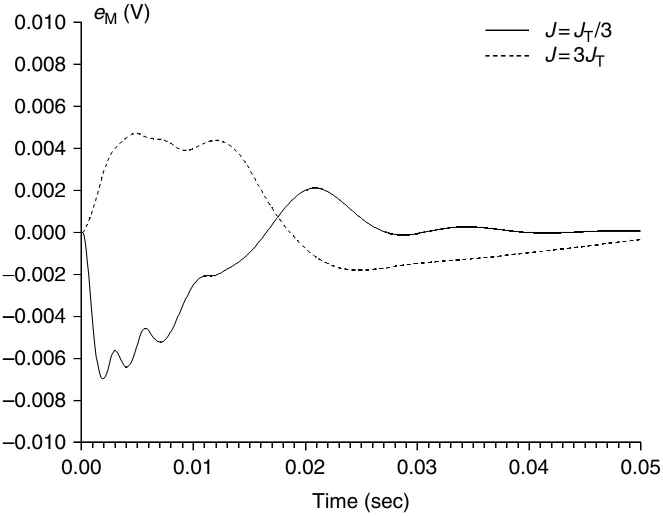


FIGURE 4.64 The angular speed tracking error responses in the case of large changes of the inertial moment. (From Kovačić, Z. and Bogdan, S., *Eng. Appl. Artif. Intelligence*, 7(5), 501–511, 1994. With permission from Elsevier.)

adaptation mechanism becomes dominant ensuring enforcement or relaxation of the control effort depending on tracking error e_M.

Figure 4.67 compares the measured angular speed responses of the PI controlled, nonadaptive, and adaptive fuzzy control system, in cases of load torque

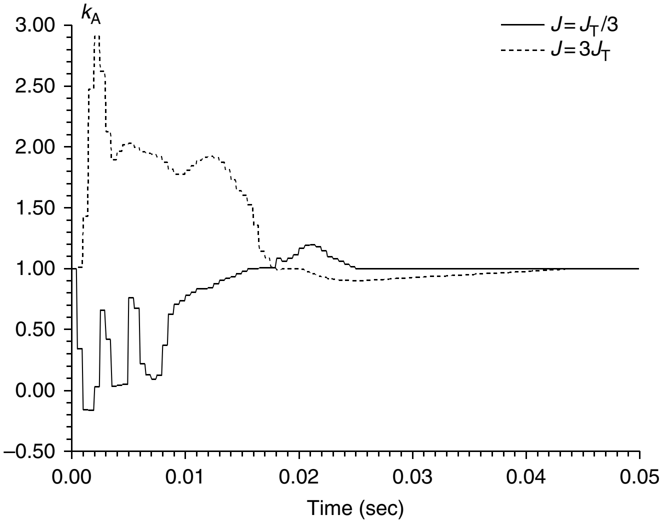


FIGURE 4.65 The fuzzy adaptation mechanism output responses in the case of large changes of the inertial moment. (From Kovačić, Z. and Bogdan, S., *Eng. Appl. Artif. Intelligence*, 7(5), 501–511, 1994. With permission from Elsevier.)

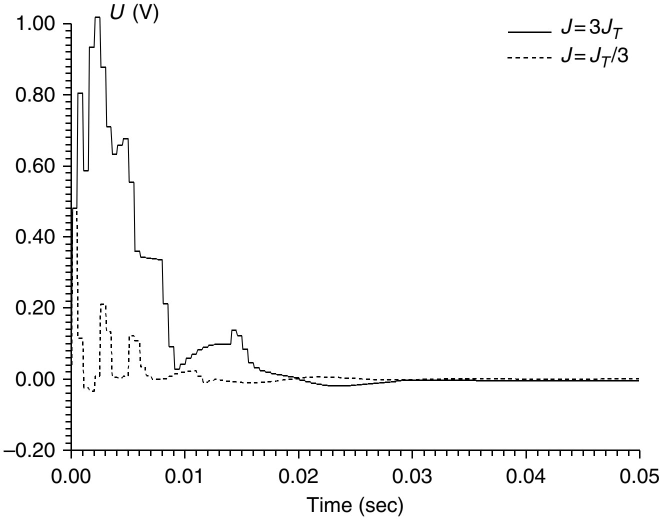


FIGURE 4.66 The adaptive fuzzy controller output responses in the case of large changes of the inertial moment. (From Kovačić, Z. and Bogdan, S., *Eng. Appl. Artif. Intelligence*, 7(5), 501–511, 1994. With permission from Elsevier.)

change $\Delta\tau_l = \tau_n/100$, where τ_n is the nominal motor torque. The drop in speed in the adaptive system is one-third of the drop in the PI controlled system due to the dominant influence of the adaptation mechanism at the beginning of the transient response (k_A reaches values up to 4.8). A noticeable decrease of system error $e(k)$

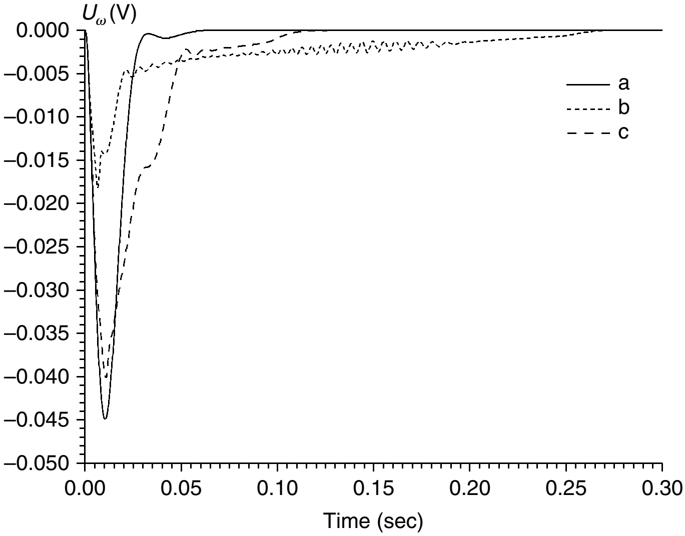


FIGURE 4.67 The measured angular speed responses of: (a) PI controlled, (b) adaptive fuzzy control, and (c) nonadaptive fuzzy control system in the case of a load torque change. (From Kovačić, Z. and Bogdan, S., *Eng. Appl. Artif. Intelligence*, 7(5), 501–511, 1994. With permission from Elsevier.)

slowed down the integration process in the PI algorithm, causing a rather slow approach to the steady state (Figure 4.68). It must be noted that the PI controller parameters were obtained according to the symmetrical optimum; that is, the PI controller was adjusted to react optimally to disturbance effects (load torque). If a particular application requires faster establishment of the steady state, a load torque observer could be used to solve the problem.

The simulation results demonstrated a stable operation of the system within the range of parameter variations being considered. They also proved the high speed of adaptation, typical of signal adaptation algorithms, and smooth steady-state operation, typical of parameter adaptation algorithms. Due to the dead-zone in the adaptation algorithm, which corresponds to the unity value of k_A, the method guarantees that the adaptation mechanism has no influence on the steady-state accuracy. The overall stability of the adaptive fuzzy system was accomplished heuristically, but the applied fuzzy reasoning was strongly based on well-founded knowledge about the target system. In some sense, the design of the hybrid fuzzy controller is a heuristic emulation of the desired controller performance, which would provide the desired system performance as if the rated parameter values were used. For the class of high-order systems that is studied here, it is possible to find a critical value of the open-loop gain. To ensure the stability of the control system, a fuzzy adaptation mechanism should not produce values in excess of this critical value. Due to the limited range of the controller output, the worst case will manifest itself in stationary oscillations.

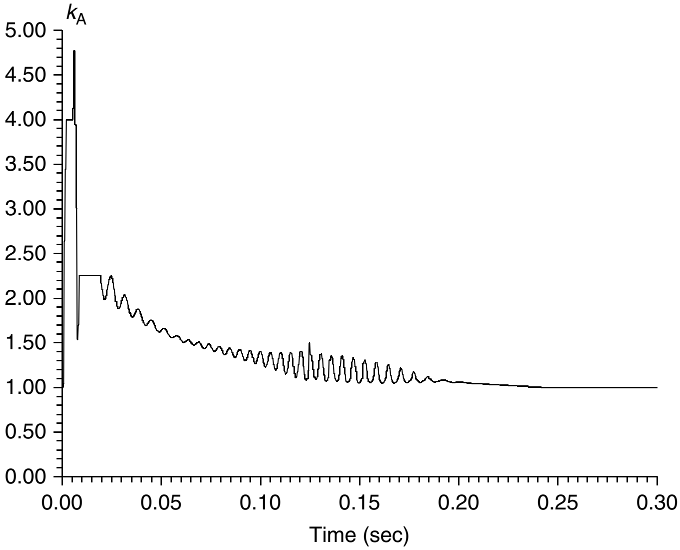


FIGURE 4.68 The adaptation mechanism output response in the case of a load torque change. (From Kovačić, Z. and Bogdan, S., *Eng. Appl. Artif. Intelligence*, 7(5), 501–511, 1994. With permission from Elsevier.)

Since the number of fuzzy subsets and their sizes, as well as the form of fuzzy control rules, depend exclusively on the designer's decisions, there is still plenty of room for further improvement for quality of the achieved system response.

A problem of sufficient resolution of the fuzzy controller and fuzzy adaptation mechanism inputs seems to be very significant. That is, the system being studied has very fast responses and the change in error assumes rather small values immediately after the initial large change. This is even more critical in the implementation of the fuzzy adaptation mechanism, due to a very slow rate of change of the model-tracking error. Practical success can be expected with a 12-bit resolution or better. Regarding the microcomputer-based implementation of angular speed adaptive fuzzy controller and the obtained experimental results, the reader can refer to Section 4.2.5.

4.2.3 Multiple Fuzzy Rule Table-Based Adaptation

A fuzzy controller is designed to achieve the desired control quality in the constrained operating range (region) around the "nominal" operating point. The size of regions is determined by controller's input and output universes of discourse. In the case of controlling a highly nonlinear system, the performance of the fuzzy controller will deteriorate in the operating regions next to or farther from the "nominal" one. In order to cover the operating range in the whole, adaptation to changes of operating points can be accomplished by using multiple fuzzy rule

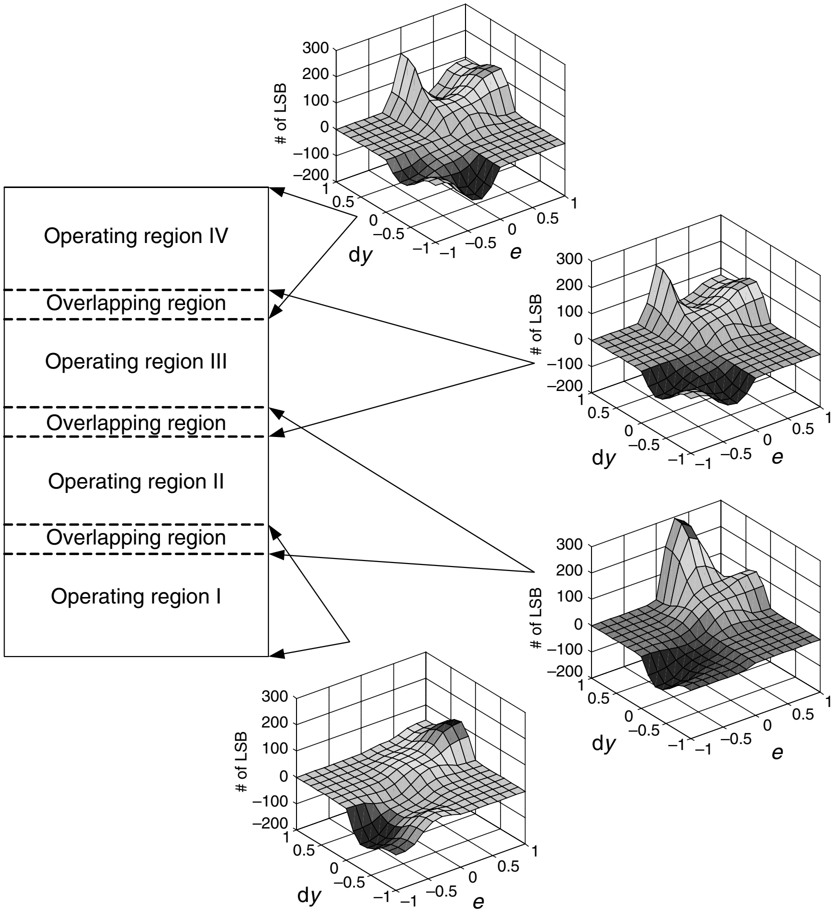


FIGURE 4.69 Adaptation by using multiple fuzzy rule tables.

tables designed for respective operating regions. Figure 4.69 shows the case of splitting the whole operating range into four such regions.

Regarding the design of multiple fuzzy rule table adaptive controllers, if the operating range is split into evenly distributed operating subregions, then all fuzzy controller parameters except output centroids may be the same for all subregions. With a classical heuristic approach, the concept of multiple fuzzy rule tables would be rather questionable from the practical point of view. But by using self-organizing fuzzy controller design methods described in Chapter 5, it is very easy to get as many fuzzy rule tables as needed, which makes this concept so attractive.

The transition from one fuzzy rule table to another should be managed in a smooth and bumpless way, which can be achieved with overlapping of operating regions (see Figure 4.69), and with the hysteresis embedded in the algorithm for switching of adjacent fuzzy rule tables.

Example 5.3.3 describes a method of nonlinear position control by using a self-organizing fuzzy logic controller (SLFLC) with multiple position-dependent fuzzy rule tables [70]. Implementation and experimental verification of the proposed multiple SLFLC structure have confirmed that the superior fuzzy controller characteristics exhibited within one operating region have been extended to the whole operating range.

4.2.4 Fuzzy MRAC Contact Force Control

Most of the presently used machine tools and robotic manipulators are designed as position and orientation controlled mechanisms. Generally, there are two types of possible manoeuvres, unconstrained where the mechanism can move freely, and constrained where restrictions are imposed on the mechanism by the environment. These restrictions are either sudden (collision with obstacles) or planned (assembling, pulling, pushing, and deburring). Using position controlled mechanisms can result in destruction of the mechanism or environment objects, due to expectable errors in task modelling and limitations of the control quality in the position control loop. Grinding, cutting, inserting, and drilling are operations where the contact force between the tool and the workpiece must be constant or must change according to a planned sequence of operations. If the mechanism is only position controlled then some operations like edge following of complex form workpieces may become a problem. If the surface of the manipulated workpiece is very rough, then position control also becomes ineffective. Another problem arises from the progressive wear of the abrasive tool. In that case, it is very difficult to maintain a constant contact force and finally, the tool can lose the contact with the workpiece.

Force control systems have been developed to solve these problems. Force sensors are mounted at the end effector side to measure the contact force. By obtaining information about the exerted force, we get a possibility to prevent a destructive collision with the environment by setting the contact force at the desired value. In spite of some specific approaches as position/force control via sensor programming [71,72], discontinuous control [73] or joint torque control [74,75], most of the control schemes could be divided into two groups (1) hybrid position/force control, (2) impedance control. Hybrid position/force control includes independent force and position control. Force control takes place on the hyperplane normal to the constraint surface and position/velocity control occurs on the tangential hyperplane [76–78].

The example presents position/force control with a completely fuzzified adaptive force control system for the single degree of freedom manipulators [79]. As shown in Figure 4.70, the system contains the outer force control loop and the inner velocity control loop. The velocity control loop contains a standard PD-type fuzzy controller, and the force control loop contains an adaptive PD-type fuzzy controller with a model reference-based PD-type fuzzy algorithm for tuning of the fuzzy force controller output. Adaptive tuning is performed by multiplying the fuzzy force controller output with tuning coefficient k_A, which means that the studied control structure resembles the MRAC control scheme with the fuzzy

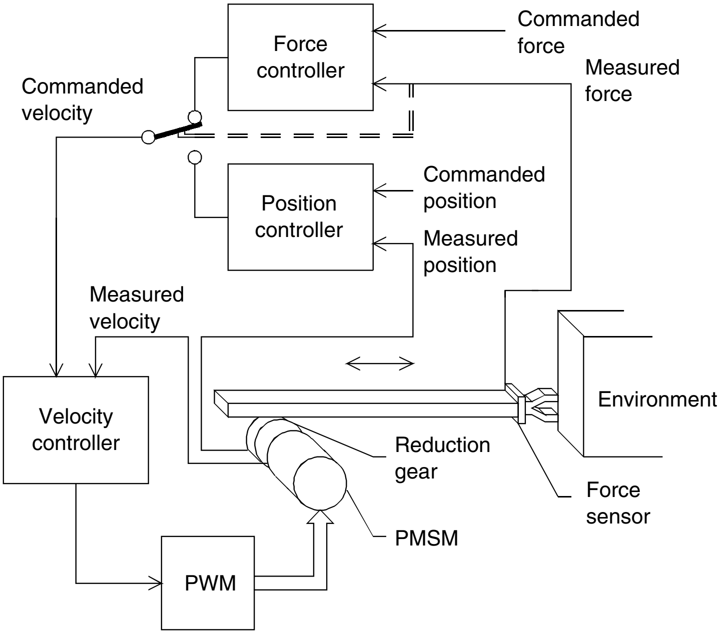


FIGURE 4.70 Force control of a single degree of freedom manipulator. (From Bogdan, S. and Kovačić, Z., *KoREMA Automatika*, 43(3–4), 119–129, 2002. With permission.)

parameter adaptation shown in Figure 4.45. The manipulator's servomechanism is driven by a vector controlled chopper-fed PMSM drive already described in Example 4.1.

The contact between the manipulator and the environment is described with the following differential equations (see Figure 4.71):

$$m_1\ddot{x}_1 + b_s(\dot{x}_1 - \dot{x}_2) + K_s(x_1 - x_2) = F \tag{4.37}$$

$$m_2\ddot{x}_2 + b_s(\dot{x}_2 - \dot{x}_1) + K_s(x_2 - x_1) + K_e(x_2 - x_3) = 0 \tag{4.38}$$

$$\tau_e = F \cdot r \tag{4.39}$$

$$\dot{x}_1 = v_1, \quad \dot{x}_2 = v_2 \tag{4.40}$$

$$v_1 = \omega \cdot r \tag{4.41}$$

where m_1 is the mass of manipulator (without end effector) [kg]; m_2, the mass of end effector [kg]; x_1, the position of manipulator [m]; x_2, the position of end effector [m]; x_3, the position of workpiece in contact with end effector [m]; v_1, the velocity of manipulator [m/sec]; v_2, the velocity of end effector [m/sec]; ω, the angular velocity of motor [rad/sec]; b_s, K_s, the damping and stiffness coefficients of force sensor; K_e, the stiffness coefficient of environment [N/m]; F, the force

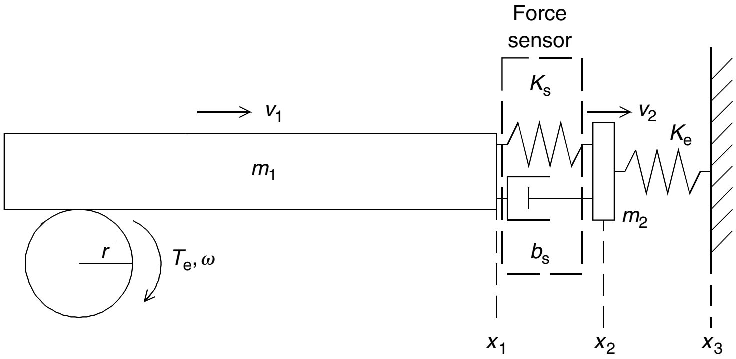


FIGURE 4.71 Model of a contact between the manipulator and the environment. (From Bogdan, S. and Kovačić, Z., *KoREMA Automatika*, 43(3–4), 119–129, 2002. With permission.)

applied to manipulator [N]; τ_e, the torque needed to produce force F [Nm]; and r, the reduction gear radius [m].

Torque equation has a form:

$$\tau_e = J_T\dot{\omega} + B\omega + \tau_l \tag{4.42}$$

On inserting Equations (4.39)–(4.42) into Equations (4.37) and (4.38), we obtain

$$\dot{v}_1 = \frac{1}{J_e}[\tau_e - B_e v_1 + rb_s v_2 - rK_s(x_1 - x_2)] \tag{4.43}$$

$$\dot{v}_2 = \frac{1}{m_2}[b_s(v_1 - v_2) + K_s(x_1 - x_2) - K_e(x_2 - x_3)] \tag{4.44}$$

where $J_e = (J_T/r) + rm_1$, $B_e = (B/r) + rb_s$.

Equations (4.43) and (4.44) include the term $K_s(x_1 - x_2)$, describing the stiffness force component detected by a force sensor, and the term $K_e(x_2 - x_3)$, describing the contact force between the end effector and the workpiece. The force detected by the force sensor is used as a feedback signal and besides the stiffness component it also contains the damping component $b_s(v_1 - v_2)$. In the studied control system, variable x_3 is acting as a disturbance, which may be caused by a rough surface or by a complex form of the workpiece.

The synthesis of the velocity control loop was made under the assumption of unconstrained manipulator motion, that is, under conditions of position control. In this case, the terms in Equations (4.43) and (4.44) caused by the contact, disappear. Masses of the manipulator and the end effector, m_1 and m_2, together with manipulator's viscous friction coefficient b are substituted with equivalent values referred to the motor shaft. Therefore, J_T should be replaced with $J_T + r^2(m_1 + m_2)$, and B with $B + r^2b$.

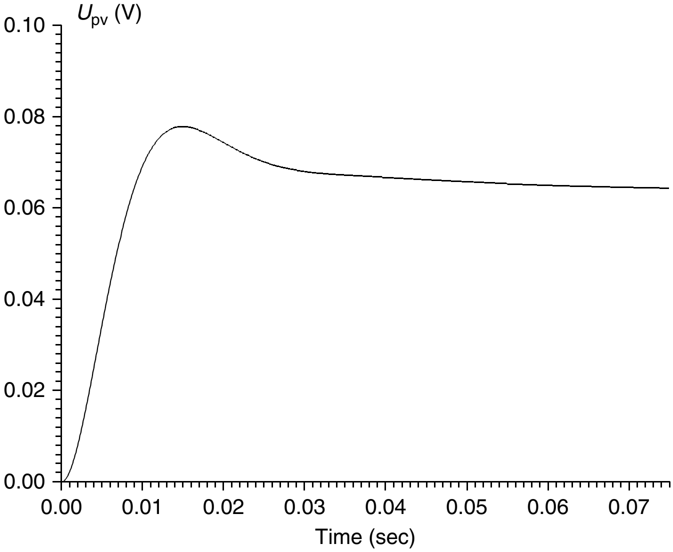


FIGURE 4.72 Measured angular speed responses with a PI controller. (From Bogdan, S. and Kovačić, Z., *KoREMA Automatika*, 43(3–4), 119–129, 2002. With permission.)

The open-loop transfer function of the linearized PMSM angular velocity control system has the form

$$G_{\text{os}}(s) = \frac{U_\omega(s)}{E_\omega(s)} = \frac{K_\text{o}(1 + T_{\text{PI}}s)}{s(1 + T_{\text{cc}}s)(1 + T_\text{M}s)(1 + T_\omega s)} \tag{4.45}$$

where

$K_\text{o} = K_{\text{PI}}K_{\text{cc}}KK_\text{M}K_\omega/T_{\text{PI}}$ open-loop gain
$K_\text{M} = 1/(B + r^2b)$ PMSM gain
$T_\text{M} = [J_\text{T} + r^2(m_1 + m_2)]/(B + r^2b)$ PMSM time constant

In the case of $m_1 = 10$ kg, $m_2 = 1$ kg, $r = 0.02$ m, and $b = 5$ Nsec/m, we obtain $K_\text{M} = 418.76$, $T_\text{M} = 2.579$ sec, and $K_\text{o} = 26.254$.

The parameters of the PI velocity controller were specified in a way to obtain the desired control quality, $K_{\text{PI}} = 15$ and $T_{\text{PI}} = 0.03$ sec, which yielded an overshoot of 20% in the measured angular velocity response (Figure 4.72).

The difference between the commanded velocity and the velocity feedback signal is related to the motor torque in the following way

$$G_1(s) = \frac{T_\text{e}(s)}{E_\omega(s)} = K_{\text{PI}}\frac{1 + T_{\text{PI}}s}{T_{\text{PI}}s}K\frac{K_{\text{cc}}}{1 + T_{\text{cc}}s} \tag{4.46}$$

Let us assume that $x_3 = 0$ (no changes in the workpiece profile). If we transform Equations (4.43) and (4.44) by using the Laplace transformation, we obtain the

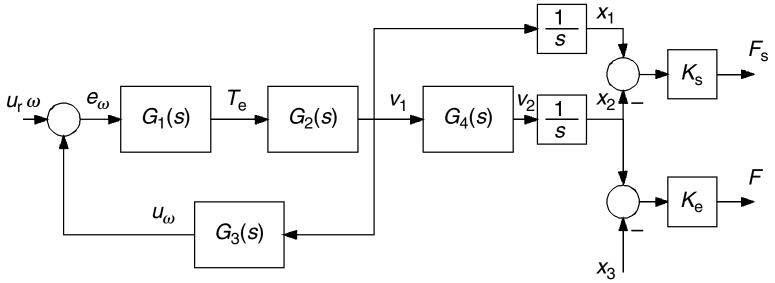


FIGURE 4.73 Open-loop force control system without controller. (From Bogdan, S. and Kovačić, Z., *KoREMA Automatika*, 43(3–4), 119–129, 2002. With permission.)

transfer function, which describes the relation between the manipulator velocity and the motor torque:

$$G_2(s) = \frac{V_1(s)}{T_e(s)} = \frac{s(m_2s^2 + b_s s + K_e + K_s)}{a_4s^4 + a_3s^3 + a_2s^2 + a_1s + a_0} \tag{4.47}$$

where

$$\begin{aligned} a_0 &= rK_sK_e, \quad a_1 = K_s(B_e - rb_s) + B_eK_e \\ a_2 &= K_s(J_e + rm_2) + b_s(B_e - rb_s) + J_eK_e \\ a_3 &= J_eb_s + B_em_2, \quad a_4 = J_em_2 \end{aligned} \tag{4.48}$$

A velocity feedback path transfer function is defined by

$$G_3(s) = \frac{U_\omega(s)}{V_1(s)} = \frac{K_\omega}{r(1 + T_\omega s)} \tag{4.49}$$

The relation between the manipulator and the end effector velocities is described by the following transfer function

$$G_4(s) = \frac{V_2(s)}{V_1(s)} = \frac{b_s s + K_s}{m_2s^2 + b_s s + K_s + K_e} \tag{4.50}$$

The transfer function of the open-loop force control system shown in Figure 4.73 without the force controller has a form

$$G_{oF}(s) = \frac{F_s(s)}{U_{r\omega}(s)} = K_s[1 - G_4(s)]\frac{1}{s}\frac{G_1(s)G_2(s)}{1 + G_1(s)G_2(s)G_3(s)} \tag{4.51}$$

On inserting (4.47)–(4.51), we obtain

$$G_{oF}(s) = \frac{KK_sK_{cc}K_{PI}(1 + T_{PI}s)(1 + T_\omega s)(m_2s^2 + K_e)}{T_{PI}s(b_6s^6 + b_5s^5 + b_4s^4 + b_3s^3 + b_2s^2 + b_1s + b_0)} \tag{4.52}$$

where

$$
\begin{aligned}
b_0 &= a_0 + K_2(K_s + K_e)\\
b_1 &= a_1 + a_0(T_{cc} + T_\omega) + K_2[b_s + (K_s + K_e)T_{PI}]\\
b_2 &= a_2 + a_1(T_{cc} + T_\omega) + a_0 T_{cc} T_\omega + K_2[m_2 + b_s T_{PI}]\\
b_3 &= a_3 + a_2(T_{cc} + T_\omega) + a_1 T_{cc} T_\omega + K_2 m_2 T_{PI}\\
b_4 &= a_4 + a_3(T_{cc} + T_\omega) + a_2 T_{cc} T_\omega\\
b_5 &= a_4 + a_3(T_{cc} + T_\omega)\\
b_6 &= a_4 T_{cc} T_\omega\\
K_2 &= \frac{K_{PI} K_\omega K_{cc} K}{T_{PI} r}
\end{aligned}
$$

We choose to use a proportional force controller with gain coefficient K_F. For the given PMSM drive parameter values (see Example 4.1) and $b_s = 1{,}000$ Nsec/m, $K_s = 100{,}000$ N/m, and $K_e = 6{,}000$ N/m (rubber surface), we obtain: $b_0 = 1.7814 \times 10^8$, $b_1 = 6.7015 \times 10^6$, $b_2 = 81436.04$, $b_3 = 436.403$, $b_4 = 1.064$, $b_5 = 7.585 \times 10^{-4}$, and $b_6 = 3.6 \times 10^{-8}$. A root loci plot for the transfer function (4.52) in the case of $K_F = 0.025$ is shown in Figure 4.74 (zeros = o and poles = x). The closed-loop system poles are marked with rectangles. The

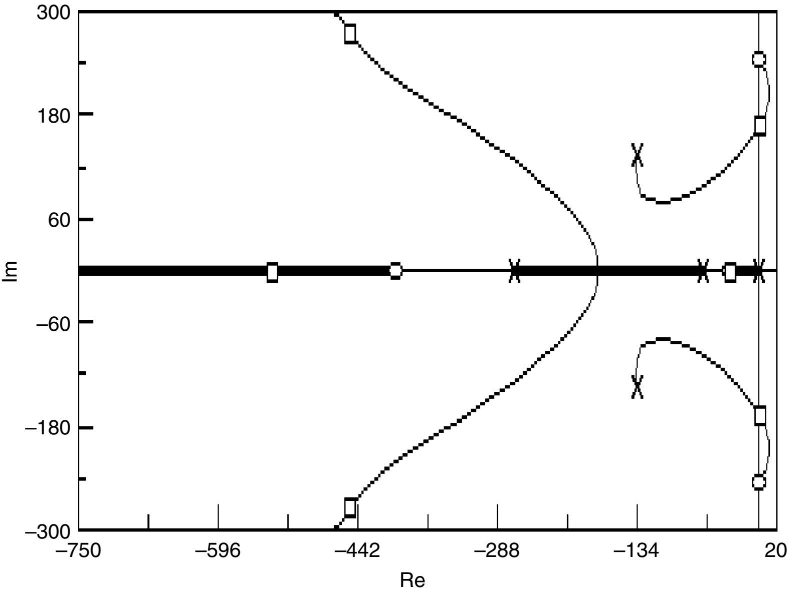


FIGURE 4.74 The force control system root loci plot for the rubber surface. (From Bogdan, S. and Kovačić, Z., *KoREMA Automatika*, 43(3–4), 119–129, 2002. With permission.)

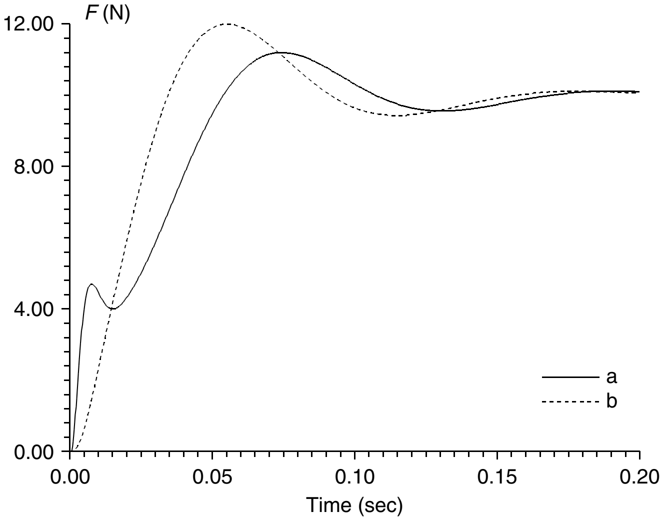


FIGURE 4.75 The responses of the force feedback signal (a) and the contact force (b) (rubber). (From Bogdan, S. and Kovačić, Z., *KoREMA Automatika*, 43(3–4), 119–129, 2002. With permission.)

poles that are far from the origin have not been displayed, since their influence on the system dynamics is negligible.

The responses of the force feedback signal and the contact force are shown in Figure 4.75. As can be seen, an overshoot in the contact force response is approximately 20% with the time of a maximum of 50 msec. Both signals have the same steady-state value of 10 N.

When the environment stiffness varies, system dynamics vary too. Figure 4.76 shows the root loci in the case of contact with an aluminium surface ($K_e = 60000$). Newly obtained values of the parameters are: $b_0 = 3.6 \times 10^8$, $b_1 = 1.2 \times 10^7$, $b_2 = 99041.8$, $b_3 = 990.3$, $b_4 = 1.064$, $b_5 = 7.585 \times 10^{-4}$, and $b_6 = 3.6 \times 10^{-8}$. In the case of $K_F = 0.025$ (i.e., the force controller gain defined for the rubber surface), the poles of the closed-loop force control system are placed in the right-hand side of the s-plane, and the contact force response has become unstable as shown in Figure 4.77.

Apparently, the cascade force control system with a P force controller and a PI angular speed controller cannot cope with large variations of the environment stiffness. The idea is to replace linear controllers with fuzzy controllers in order to achieve increased robustness of the target force control system. First let us substitute the PI angular speed controller with the hybrid fuzzy controller described in Example 4.1. Due to identical angular speed control loops, the hybrid fuzzy controller used here is also identical.

During contact with the rubber surface, the force response obtained with the hybrid fuzzy angular speed controller (see Figure 4.78) is similar to the force response shown in Figure 4.71. The force response obtained during contact with

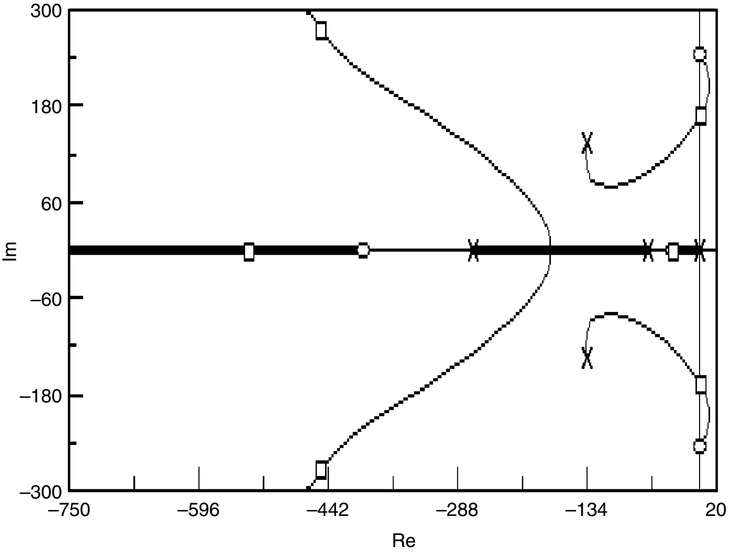


FIGURE 4.76 The force control system root loci plot for the aluminum surface. (From Bogdan, S. and Kovačić, Z., *KoREMA Automatika*, 43(3–4), 119–129, 2002. With permission.)

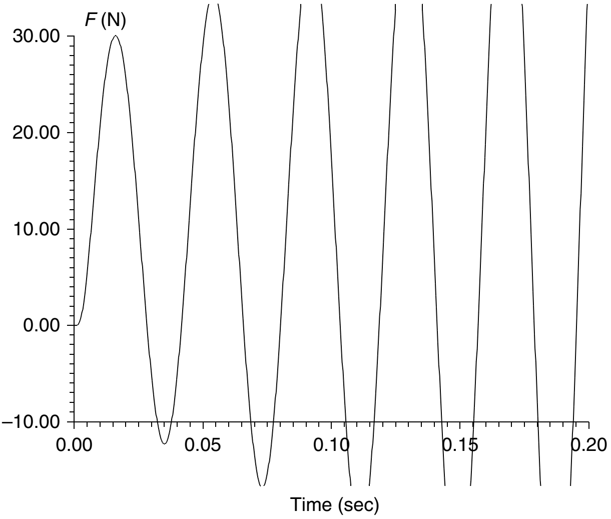


FIGURE 4.77 The contact force response in the case of contact with the aluminum surface. (From Bogdan, S. and Kovačić, Z., *KoREMA Automatika*, 43(3–4), 119–129, 2002. With permission.)

the aluminum surface is also shown in Figure 4.78. Despite noticeable oscillations, the force response is stable. Compared to the unstable response obtained with the PI controller (Figure 4.77), introduction of the hybrid fuzzy controller has contributed to the stabilization of the system.

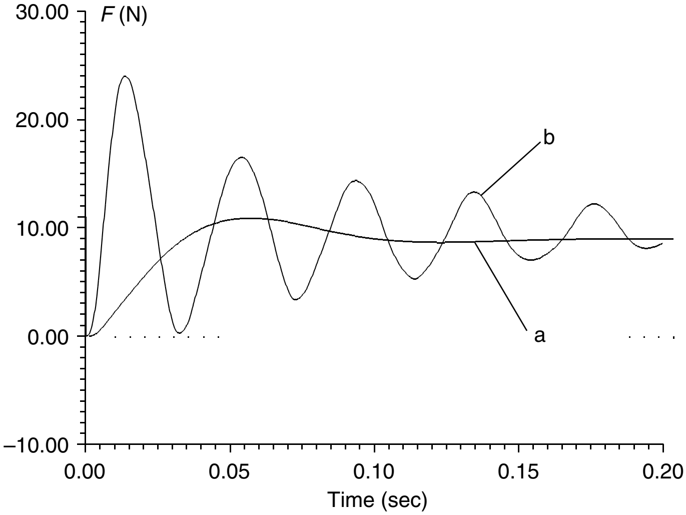


FIGURE 4.78 The contact force responses with the P force controller and the hybrid fuzzy angular speed controller: rubber surface (a) and aluminum surface (b). (From Bogdan, S. and Kovačić, Z., *KoREMA Automatika*, 43(3–4), 119–129, 2002. With permission.)

Further improvement can be achieved by using a fuzzy controller in the force control loop. Let us use the nonintegral fuzzy controller designed for the contact with rubber surface. The maximum change of commanded force was estimated to be 10 N, and therefore the maximum error value was $-10.0 \le e_F \le 10.0$. The maximum error change during control interval $T_d = 0.5$ msec was estimated to be $-0.7 \le \Delta e_F \le 0.7$. The distribution of membership functions along e_F and Δe_F universes of discourse and organization of the fuzzy rule table were the same as of the hybrid fuzzy angular speed controller (Figure 4.7). Only the scaling of singleton values in the fuzzy rule table was different (division by 20), in order to obtain the desired contact force performance. The force controller output value was defined by following the center of gravity principle (2.22).

Figure 4.79 shows the force response in the contact with rubber surface when force and angular speed controllers are fuzzy. The overshoot and peak time are almost the same as they were in the case of using standard controllers (Figure 4.75). In the case of contact with the aluminium surface, the contact force response also shown in Figure 4.79 indicates that the fuzzy force controller has further stabilized the system response. Sensitivity to environment stiffness variations has been noticeably reduced, but there is still the problem of the first maximum, which is too high.

In order to achieve an almost uniform contact force response in the cases of contact with stiffness-varying environment, let us apply a MRAC scheme with fuzzy parameter adaptation that tunes the fuzzy force controller output (Figure 4.80). The fuzzy control scheme is equivalent to the adaptive fuzzy control scheme from Figure 4.45.

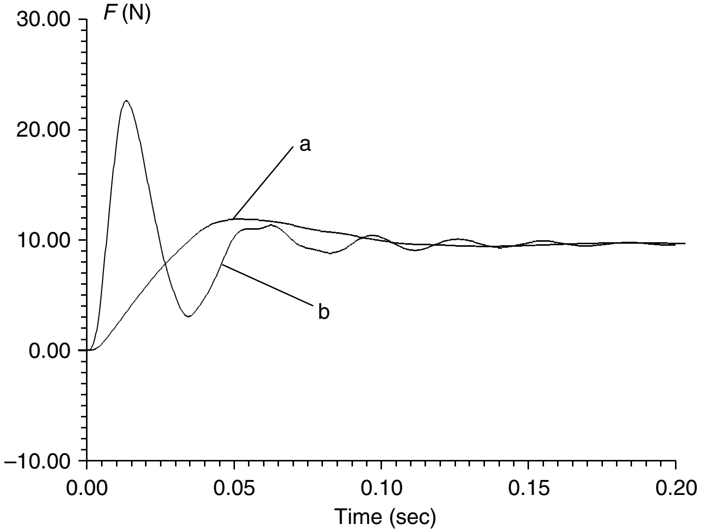


FIGURE 4.79 The contact force responses with a fuzzy force controller and a hybrid fuzzy angular speed controller: rubber surface (a) and aluminum surface (b). (From Bogdan, S. and Kovačić, Z., *KoREMA Automatika*, 43(3–4), 119–129, 2002. With permission.)

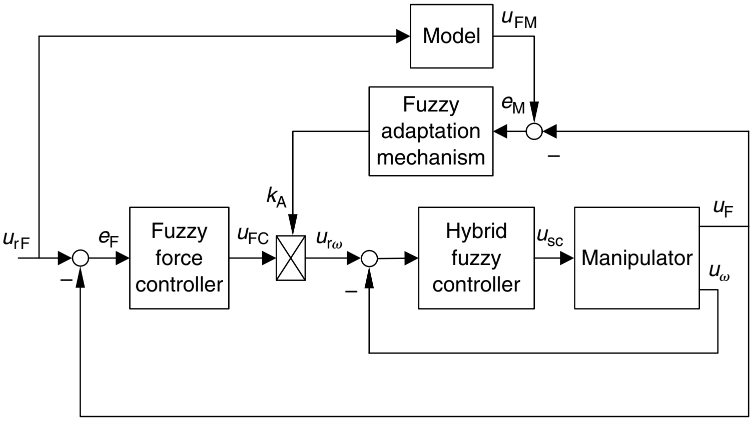


FIGURE 4.80 Fuzzy model reference-based adaptive force control system. (From Bogdan, S. and Kovačić, Z., *KoREMA Automatika*, 43(3–4), 119–129, 2002. With permission.)

The second-order reference model parameters are referred to the contact with rubber surface. The desired reference model performance indices are overshoot in the response, $\sigma_m = 10\%$, and peak time, $t_m = 0.075$ sec. The responses of the reference model and the measured force are shown together in Figure 4.81.

Likewise in Example 4.5, the main design goal was to create the set of rules of fuzzy adaptation mechanism, which would generate tuning coefficient values and adapt the fuzzy force controller output, in order to make the contact force

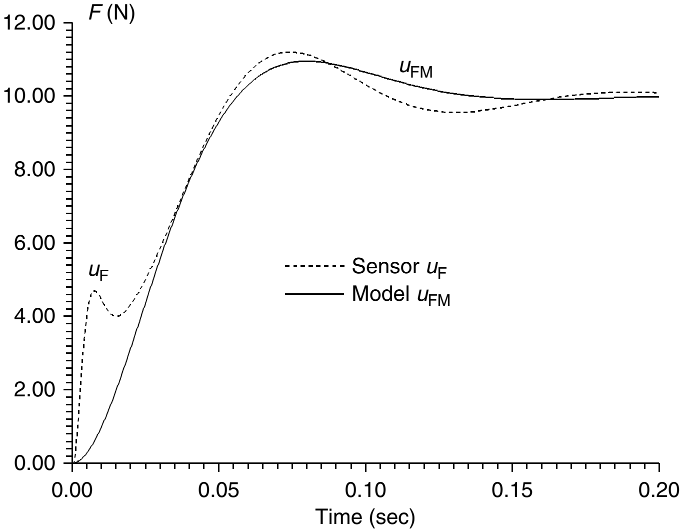


FIGURE 4.81 The reference model and measured force responses in the case of contact with the rubber surface. (From Bogdan, S. and Kovačić, Z., *KoREMA Automatika*, 43(3–4), 119–129, 2002. With permission.)

transient response almost unaffected by environment stiffness variations. The fuzzy adaptation algorithm has a nonintegral character, that is, the current tuning coefficient value is directly determined by the magnitudes of inputs, tracking error $e_M = u_{FM} - u_F$ and change of error Δe_M, where u_F is the force feedback signal, and u_{FM} is the reference model output (desired force feedback signal).

A fuzzy adaptation mechanism has seven linguistic subsets defined for both inputs (universes of discourse EM and DEM): NL, NM, NS, Z, PS, PM, and PL. For the studied force control system, the maximum error value was estimated to be 2.5, that is, $-2.5 \leq e_M(k) \leq 2.5$. The maximum error change during control interval $T_d = 0.5$ msec was estimated to be 0.75, that is, $-0.75 \leq \Delta e_M(k) \leq 0.75$. The distributions of membership functions along e_M and Δe_M universes of discourse are the same as they were in Example 4.4 (see Figure 4.47). The size of input fuzzy sets and output singleton values shown in Table 4.4 were defined in order to keep tracking error e_M within 20% of the imposed change of the force reference input: $|e_M(k)| \leq 0.2\Delta u_{rF}$.

Figure 4.82 shows the contact force responses in the case of contact with rubber ($K_e = 6000$) and aluminum ($K_e = 60000$) surfaces. Figure 4.83 shows the tracking error responses, Figure 4.84 shows the tuning coefficient responses, and Figure 4.85 shows the adapted force controller output responses for both studied cases.

Tuning coefficient k_A is oscillatory in the initial part of the response as the fuzzy-logic tuning algorithm attempts to compensate for the nonmonotonous character of the force feedback sensor response (see Figure 4.75 and Figure 4.81). The tracking error is kept within ±20% of the imposed change of reference

TABLE 4.4
The Fuzzy Rule Table of the Adaptation Mechanism

	NLEM	NMEM	NSEM	ZEM	PSEM	PMEM	PLEM
NLDEM	−1.0	−1.0	−1.0	−0.5	0.75	0.9	1.0
NMDE	−1.0	−0.5	−0.25	0	0.9	1.0	1.1
NSDE	−1.0	−0.25	−0.25	0.25	1.0	1.1	1.15
ZDE	−0.5	0	0.5	1.0	1.1	1.15	1.3
PSDE	0.5	0.75	1.0	1.1	1.15	1.3	2.0
PMDE	0.9	1.0	1.1	1.15	1.3	1.5	2.0
PLDE	1.0	1.1	1.15	1.3	2.0	2.0	2.0

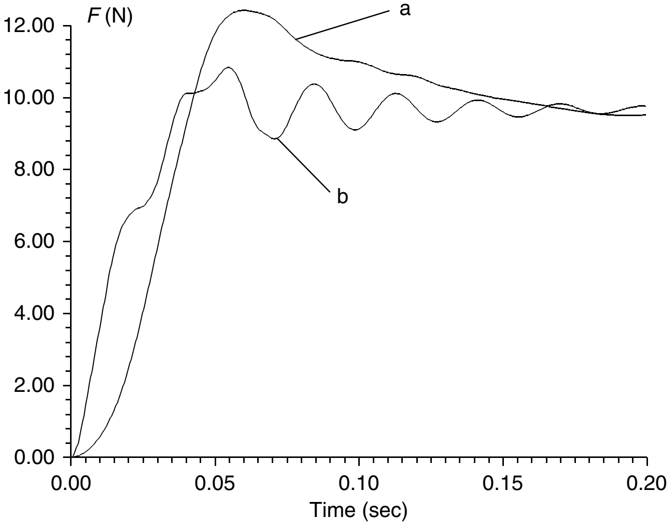


FIGURE 4.82 The contact force responses with the adaptive force controller in the case of contact with rubber (a) and aluminium surface (b). (From Bogdan, S. and Kovačić, Z., *KoREMA Automatika*, 43(3–4), 119–129, 2002. With permission.)

force input ($\Delta u_{\mathrm{rF}} = 10$ N), thus proving a good adaptation in the case of varying environment stiffness. Since the reference model is a second-order system and the linearized model of the studied force control system is a seventh-order system, the simulation results have shown that fuzzy logic has achieved what standard MRAC methods could not be expected to achieve. The experiments have included contact with the rubber and aluminum surfaces, thus showing the ability of force-controlled mechanism to successfully manipulate workpieces made of very different materials.

Let us now suppose that the manipulator follows the edge of a workpiece with a complex profile, as shown in Figure 4.86. The tool travels along the ordinate axis to the right with the velocity of 10 cm/sec and simultaneously acts on the workpiece with a constant contact force of 10 N. In the studied force control system, such

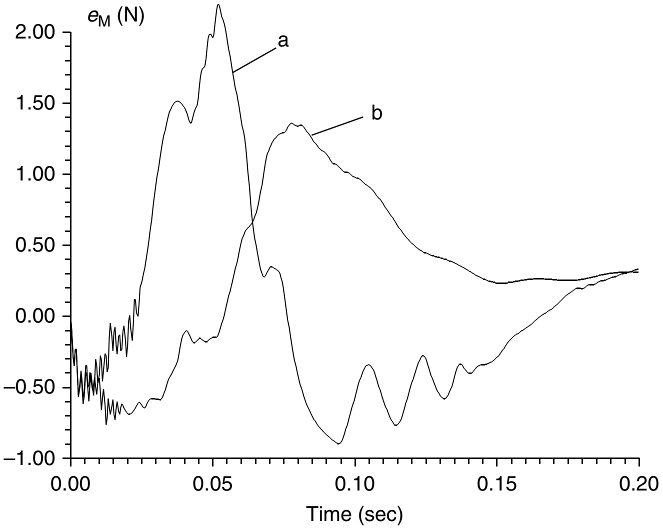


FIGURE 4.83 The tracking error responses with the adaptive force controller in the case of contact with rubber (a) and aluminum surface (b). (From Bogdan, S. and Kovačić, Z., *KoREMA Automatika*, 43(3–4), 119–129, 2002. With permission.)

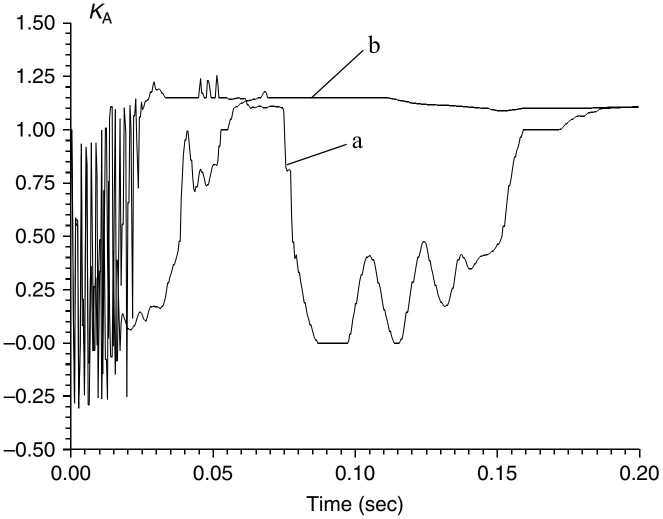


FIGURE 4.84 The tuning coefficient responses in the case of contact with rubber (a) and aluminum surface (b). (From Bogdan, S. and Kovačić, Z., *KoREMA Automatika*, 43(3–4), 119–129, 2002. With permission.)

a task may be described as a test of ability of the systems to act in the presence of disturbance x_3 (Figure 4.76).

Figure 4.87 shows the difference between the reference force value (10 N) and the actual contact force in case of using standard (a) and adaptive fuzzy (b)

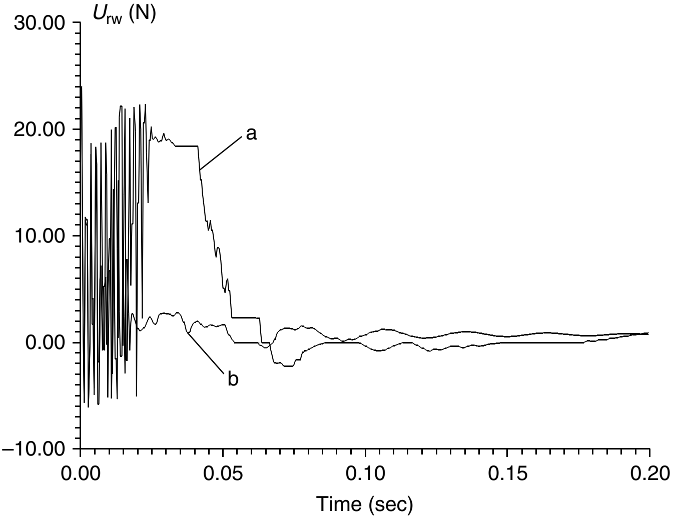


FIGURE 4.85 The adapted force controller output responses in the case of contact with (a) rubber and aluminum surface (b). (From Bogdan, S. and Kovačić, Z., *KoREMA Automatika*, 43(3–4), 119–129, 2002. With permission.)

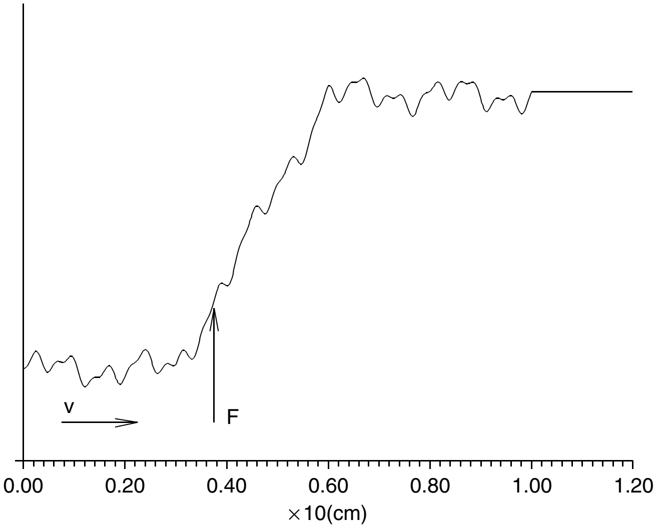


FIGURE 4.86 The profile of manipulated workpiece. (From Bogdan, S. and Kovačić, Z., *KoREMA Automatika*, 43(3–4), 119–129, 2002. With permission.)

controllers for contact with the rubber surface. In both cases the difference is kept below 5% of the given force value. But if the manipulator is in contact with the aluminum surface (Figure 4.88), the adaptive fuzzy control scheme shows superior performance, because the system with standard linear controllers has become

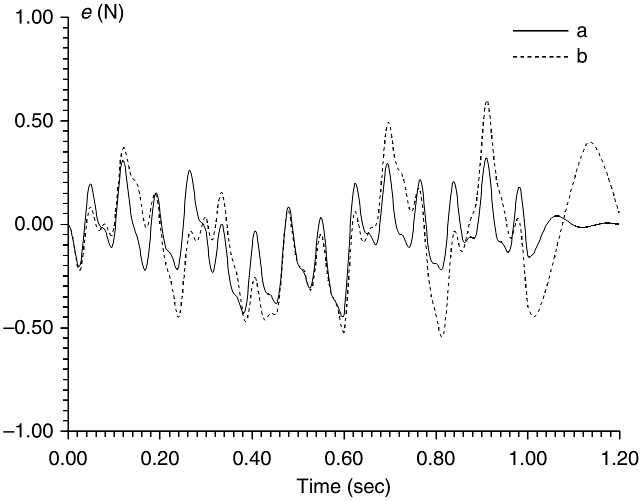


FIGURE 4.87 The force error response in the case of contact with the rubber surface: profile following with linear controllers (a) and adaptive fuzzy controllers (b). (From Bogdan, S. and Kovačić, Z., *KoREMA Automatika*, 43(3–4), 119–129, 2002. With permission.)

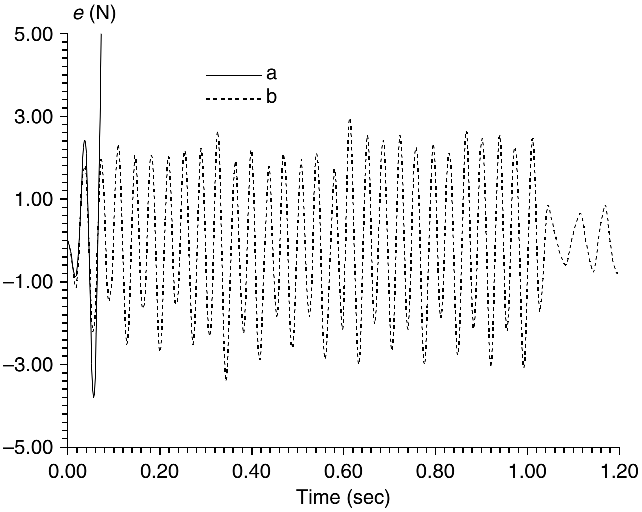


FIGURE 4.88 The force error response in the case of contact with the aluminum surface: profile following with linear controllers (a) and adaptive fuzzy controllers (b). (From Bogdan, S. and Kovačić, Z., *KoREMA Automatika*, 43(3–4), 119–129, 2002. With permission.)

unstable, while the adaptive fuzzy force controller has stabilized the system and kept the difference within 30% of the given force value.

According to the simulation results, a model reference-based fuzzy adaptation mechanism is able to keep the error between the reference model and system output responses within desired limits. The simulation results further indicate a stable performance of the force control system for a wide range of environment stiffness variations. Even though the proposed fuzzy control scheme stabilizes the system in the case of environment parameter variations, a heuristic nature of its design makes a rigorous stability analysis very difficult. The stability mainly depends on designer's experience with determination of fuzzy controller parameters. The adaptive fuzzy force control method has also been effective in the case of contact with a rough surface or a complex form workpiece.

4.2.5 Fuzzy MRAC Angular Speed Control

In this example, we describe a design and a practical implementation of a fuzzy MRAC scheme in the angular speed control loop of a thyristor converter-fed direct current motor (DCM) drive [80]. The adaptive fuzzy controller was implemented in the 32-bit VME-based microcomputer hardware (Motorola 68030). Internal calculations were performed with the floating point precision, while inputs and outputs were handled as 12-bit integers (Figure 4.89). The fuzzy adaptation mechanism is used for gain adjustment of the fuzzy angular speed controller output (see Figure 4.45). The angular speed controller being used is a hybrid fuzzy controller, whose structure is described in detail in Example 4.1.

The rated values of the target system parameters were as follows: $K_{\mathrm{TC}} = 45$ V/V (converter gain), $T_{\mathrm{TC}} = 5$ msec (converter time constant), $K_{\mathrm{a}} = 0.0612$ A/V (armature gain), $T_{\mathrm{a}} = 18.4$ msec (armature time constant), $K = 1.211$ Vsec (torque coefficient), $J_{\mathrm{T}} = 0.0175$ kg m^2 (moment of inertia), $K_{\mathrm{l}} = 0.01957$ Nmsec (load coefficient), $K_{\omega} = 0.065$ Vsec (filter gain), and $T_{\omega} = 25$ msec (filter time constant). For a single phase fully controlled thyristor bridge, the fuzzy MRAC control algorithm has been synchronized with the fundamental frequency of 100 Hz (i.e., a control interval value is $T_{\mathrm{d}} = 10$ msec).

This is a nonlinear control scheme because it contains nonlinear hybrid fuzzy control and fuzzy adaptation algorithms. The DCM drive itself is a nonlinear high-order control object whose dynamic characteristics can be linearized in a selected

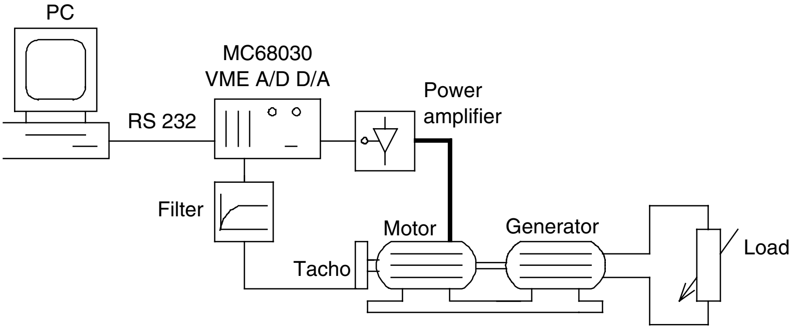


FIGURE 4.89 The setup of a thyristor converter-fed DCM drive. (From Kovačić, Z., Bogdan, S., and Štajdohar, M., *11th IEEE Intl. Symp. Intell. Contr.*, 49–54, 1996. With permission.)

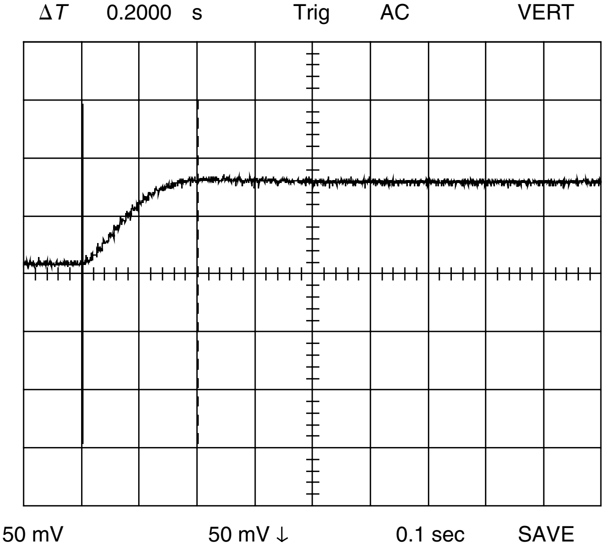


FIGURE 4.90 The transient response of the implemented second-order reference model. (From Kovačić, Z., Bogdan, S., and Štajdohar, M., *11th IEEE Intl. Symp. Intell. Contr.*, 49–54, 1996. With permission.)

operating point. Therefore, for a particular operating point, a reference model has been used to describe a desired dynamic behavior. In the system being studied, the desired behavior is represented by means of second-order reference model transfer function (4.20).

The desired reference model performance indices are the response overshoot, $\sigma_{\mathrm{m}} = 5\%$, and the peak time, $t_{\mathrm{m}} = 0.2$ sec. After discretization ($T_{\mathrm{d}} = 10$ msec), the reference model is described with a recursive equation

$$u_{\omega\mathrm{M}}(k) = 1.7076u_{\omega\mathrm{M}}(k-1) - 0.7426u_{\omega\mathrm{M}}(k-2) + 0.0349u_r(k-2) \quad (4.53)$$

The transient response of reference model (4.53) implemented in the microcontroller software is shown in Figure 4.90.

The heuristic synthesis of the fuzzy controller has the goal to find controller parameter values, which would enforce the closed-loop system to follow the dynamic behavior of reference model (4.53) as close as possible. Hybrid fuzzy controller inputs, system error $e(k)$ and change of error $\Delta e(k)$, form a system error phase plane. The universes of discourse for both inputs and controller output $u_{\mathrm{FC}}(k)$ are directly related to the given resolution of 12-bit bipolar A/D and D/A converters (Figure 4.89).

The error phase plane has been split into several subareas — pages (see Figure 4.91), all in order to provide fine and coarse fuzzy control in a very practical way [81]. More details about the fuzzy controller implementation can be found in the study discussed in Section 7.2.1.

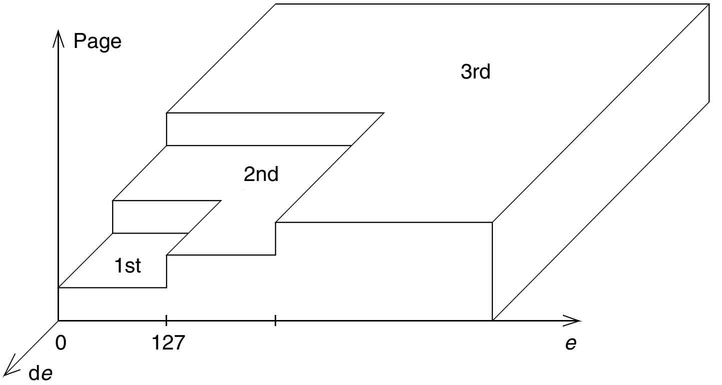


FIGURE 4.91 Division of the first quadrant of the system phase plane into subareas — pages. (From Kovačić, Z., Bogdan, S., and Štajdohar, M., *11th IEEE Intl. Symp. Intell. Contr.*, 49–54, 1996. With permission.)

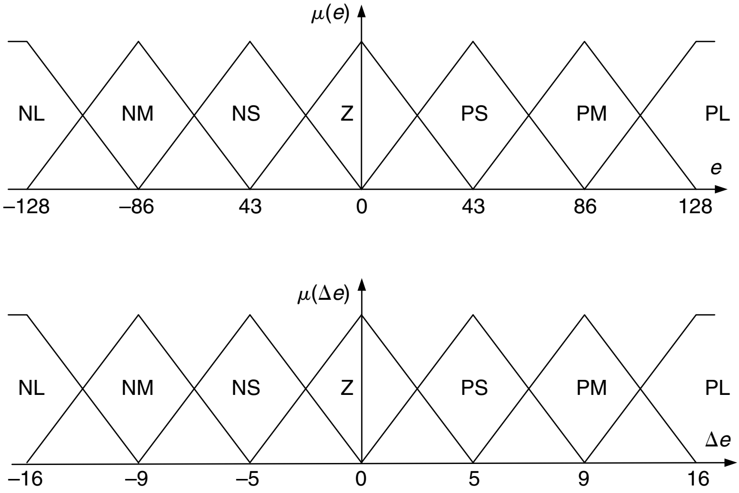


FIGURE 4.92 Distribution of membership functions for a fuzzy controller.

Fine control is primarily required in the surroundings of the phase plane origin (i.e., in the first page). Therefore, the universes of discourse of the first page were divided into seven linguistic subsets, and the triangular forms of their membership functions were adopted (Figure 4.92).

The fuzzy rule table associated with the first page is shown in Table 4.5. It must be noted that all singleton values in the table are expressed as floating point numbers of bit units related to the ±11-bit resolution of a 12-bit bipolar D/A converter (Figure 4.92).

The fuzzy controller output value is computed according to center of gravity principle (2.22). In the discussed control concept, the appearance of larger fuzzy

TABLE 4.5
The Fuzzy Rule Table for Fine Control (First Page)

	NLE	NME	NSE	ZE	PSE	PME	PLE
NLDE	−25.2	−21	16.8	−4.9	−11.2	−8.4	−5.6
NMDE	−16.8	−14	−11.2	−2.8	−5.6	−4.2	−2.8
NSDE	−8.4	−5.6	−4.2	−2.1	−1.75	0	0.35
ZDE	−1.4	−1.05	−0.7	0	0.7	1.05	1.4
PSDE	−0.35	0	1.75	2.1	4.2	5.6	8.4
PMDE	2.8	4.2	5.6	2.8	11.2	14.0	16.8
PLDE	5.8	8.4	11.2	4.9	16.8	21	25.2

controller input values that are not lying in the first page is assumed as coarse fuzzy control. The output of a coarse fuzzy controller is calculated in the following way. First, larger input values are scaled (divided by two, four, etc.) in order to fit into the most inner, that is, the first page. Then, calculation of a corresponding fine control output value is performed. Finally, the coarse control output value is obtained after reverse scaling (multiplication by two, four, etc.) of the corresponding fine control output value. It should be noted that the so-designed fuzzy controller has better disposition to control linear and nonlinear systems, because the impact on the system response caused by magnitude variations of the reference input has been significantly reduced.

Besides its ability to switch smoothly between coarse and fine control, the fuzzy controller is also capable of switching its mode of operation between a PD-type mode and a PI-type mode, depending on the magnitudes of fuzzy controller inputs:

$$\begin{array}{lcll} e(k) \in \mathrm{ZE} & \text{or} & \Delta e(k) \in \mathrm{ZDE} & \rightarrow \text{PI-type mode} \\ e(k) \notin \mathrm{ZE} & \text{and} & \Delta e(k) \notin \mathrm{ZDE} & \rightarrow \text{PD-type mode} \end{array} \tag{4.54}$$

A PD-type fuzzy controller is activated during sufficiently large reference input or system output changes, while a PI-type fuzzy controller, sharing the same fuzzy rule table with the former one (but with an order of magnitude of lower output gain coefficient), supports steady-state accuracy and cancels disturbance effects.

Referring to Figure 4.45, adaptation is achieved by multiplying the controller output with tuning coefficient k_A. A nonintegral fuzzy adaptation mechanism is responding to tracking error $e_\mathrm{M}(k)$ and its change $\Delta e_\mathrm{M}(k)$. In the presented application, it has seven linguistic subsets defined for both inputs in a usual way: NL, NM, NS, Z, PS, PM, and PL. Distributions of membership functions for the subsets of e_M and $\mathrm{d}e_\mathrm{M}$ are shown in Figure 4.93.

The fuzzy rule table of the fuzzy adaptation mechanism is shown in Table 4.6. The values of singletons in the fuzzy rule table were defined by trial and error in order to keep the tracking error within the desired design goal, $|e_\mathrm{M}| \leq 0.1\Delta u_\mathrm{r}$.

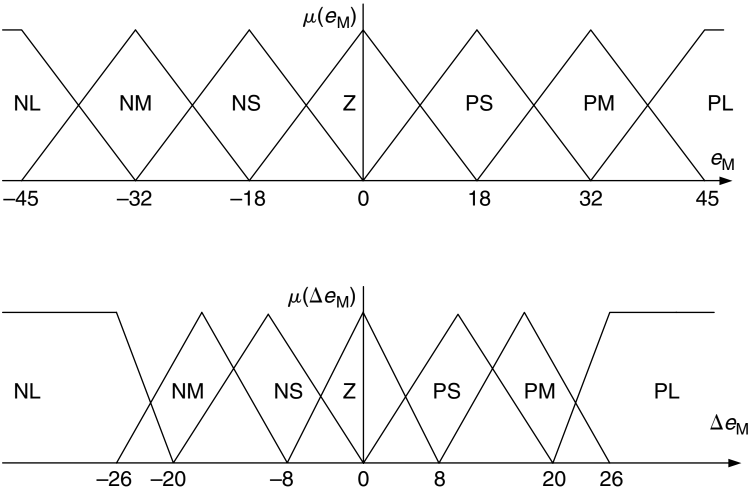


FIGURE 4.93 Distribution of membership functions for a fuzzy adaptation algorithm.

TABLE 4.6
The Fuzzy Rule Table of the Adaptation Mechanism

	NLEM	NMEM	NSEM	ZEM	PSEM	PMEM	PLEM
NLDEM	1.3	1.3	1.2	1.1	1.0	0.95	1.0
NMDEM	1.3	1.2	1.1	1.0	0.95	1.0	1.01
NSDEM	1.2	1.1	1.0	0.95	1.0	1.01	1.05
ZDEM	1.1	1.0	0.95	1.0	1.01	1.05	1.2
PSDEM	1.0	0.95	1.0	1.01	1.05	1.2	1.3
PMDEM	0.95	1.0	1.01	1.05	1.2	1.3	1.4
PLDEM	1.0	1.01	1.05	1.2	1.3	1.4	1.5

Three controllers — a standard PI controller ($K_{\mathrm{PI}} = 0.5$, $T_{\mathrm{PI}} = 0.2$ sec), nonadaptive fuzzy controller, and adaptive fuzzy controller, were tested experimentally in the laboratory setup of a DC motor drive (Figure 4.89). Figure 4.94 compares the measured angular speed and controller output transient responses of the PI-controlled, fuzzy nonadaptively, and adaptively controlled system, in the case of rated parameter values. The parameters that vary most in the system are thyristor converter gain, K_{TC}, and moment of inertia, J_{T}.

The controllers being tested provide almost uniform dynamic behavior of the system. Although the measured angular speed signal contains noise (caused by armature current pulsations), adaptation mechanism remains passive during the steady-state condition, thanks to the zero action zone ($k_{\mathrm{A}} = 1$) in the fuzzy rule table (please refer to Table 4.6).

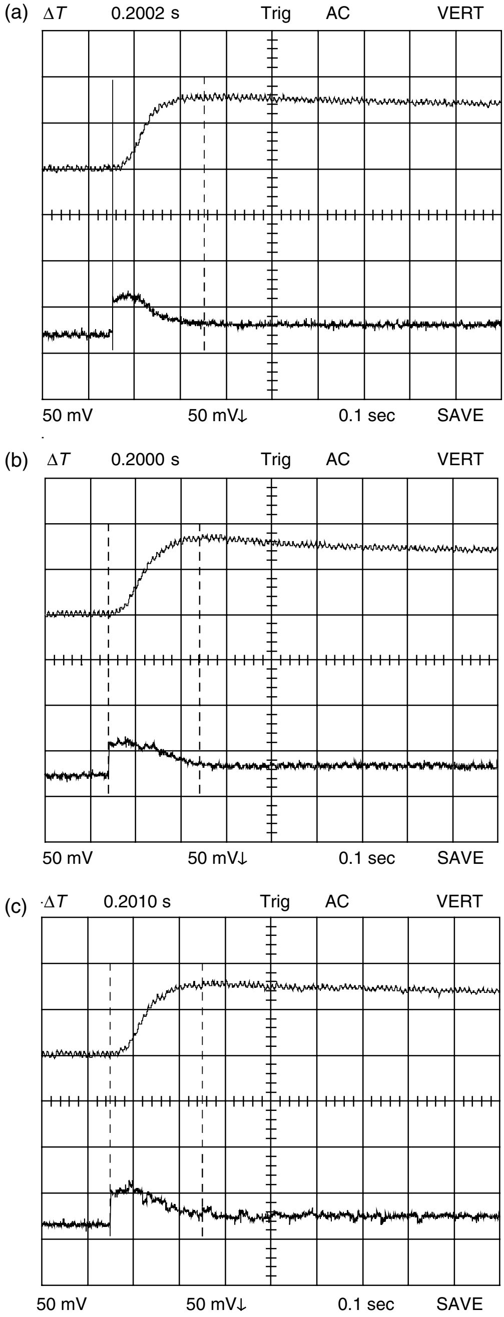


FIGURE 4.94 The measured angular speed and controller output responses for the rated system parameters (a) PI controller, (b) fuzzy controller, and (c) adaptive fuzzy controller. (From Kovačić, Z., Bogdan, S., and Štajdohar, M., *11th IEEE Intl. Symp. Intell. Contr.*, 49–54, 1996. With permission.)

Figure 4.95 compares the measured angular speed and controller output transient responses of the PI-controlled, nonadaptively, and adaptively fuzzy controlled system, in the case of a two times decreased value of the open-loop gain coefficient. The PI and nonadaptively fuzzy controlled systems are affected by parameter variations, because peak time t_m has been significantly increased (from 200 to 560 msec). In the same time, the adaptive fuzzy controller has maintained the desired dynamic behavior, showing a superior performance with respect to other two controllers.

Figure 4.96 compares the measured angular speed and controller output transient responses of the PI-controlled, nonadaptively, and adaptively fuzzy controlled system, in the case of the two times increased value of the open-loop gain coefficient ($K_\mathrm{s} = 2$). PI and nonadaptive fuzzy controllers are affected again by parameter variations, allowing a too high overshoot (PI — 80%, nonadaptive fuzzy — 40%) and a noticeable decrease of the peak time (from 200 to 100 msec). The adaptive fuzzy controller provided the best of the three responses, keeping the response overshoot at 25% and the peak time at 150 msec.

For the rated system parameters, Figure 4.97 compares the measured angular speed and controller output transient responses of the PI-controlled, fuzzy nonadaptively, and adaptively controlled system, in the case of load torque change $\Delta\tau_\mathrm{l}$. The results obtained indicate that the nonadaptive fuzzy controller has provided the worst, while the adaptive fuzzy controller has provided the best angular speed transient response. Apparently, parameter values of the nonadaptive fuzzy controller are the result of compromise, which had to be done in favor of acceptable dynamic behavior when the reference input was changed. On the other hand, these parameters are not optimally adjusted to compensate undesired disturbance effects. The usage of adaptive fuzzy controller solved this problem to a large extent, because the drop of speed caused by load variations was noticeably reduced, but the settling time was slightly extended (PI — 0.9 sec, adaptive fuzzy — 1.1 sec). The problem of slow descent to the steady state has been noticed earlier during simulation experiments in Example 4.5. This occurred because of the dominant influence of adaptation mechanism at the beginning of transient response, which later suppressed the efficiency of the integral mode of operation of the fuzzy controller.

The experimental results demonstrated that desired dynamic characteristics of the angular speed response could be successfully maintained even if system parameter values increased or decreased two times with respect to the rated values. The accomplished speed of adaptation was fast enough to be effective within each system output transition. It should be noted that this was achieved with very simple structures of the fuzzy controller and adaptation algorithms, which had a linear distribution of triangular fuzzy sets for both of their inputs. The simplicity of designed structures was preferred because of easier microcomputer implementation of the adaptive controller. Readers interested in microcomputer implementation of fuzzy control algorithms can find further details and some code writing hints in the study discussed in Section 7.2.1.

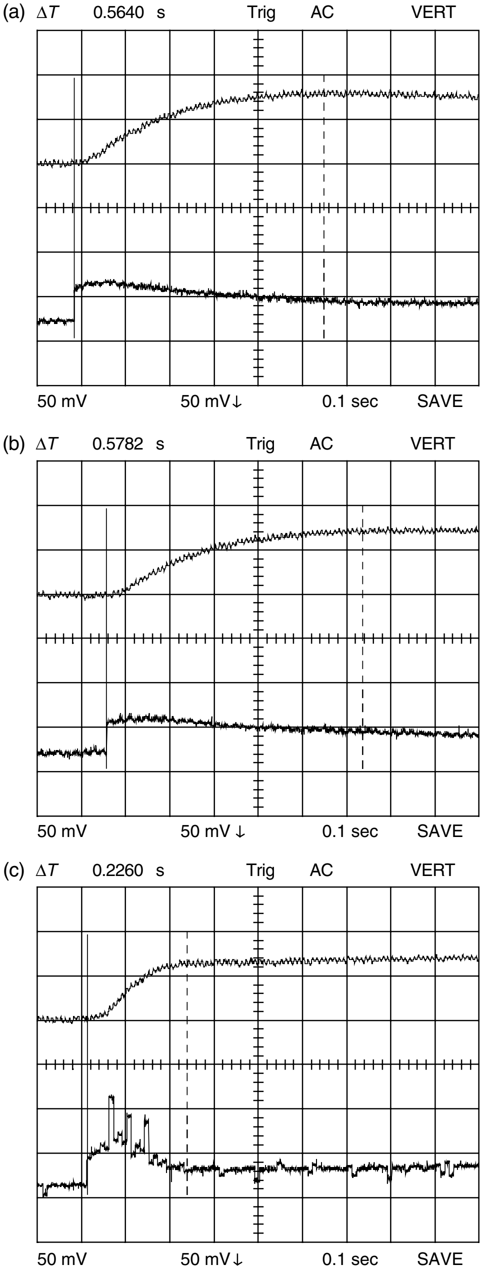


FIGURE 4.95 The measured angular speed and controller output responses for the two times decreased open-loop gain ($K_S = 0.5$) (a) PI controller, (b) fuzzy controller, and (c) adaptive fuzzy controller. (From Kovačić, Z., Bogdan, S., and Štajdohar, M., *11th IEEE Intl. Symp. Intell. Contr.*, 49–54, 1996. With permission.)

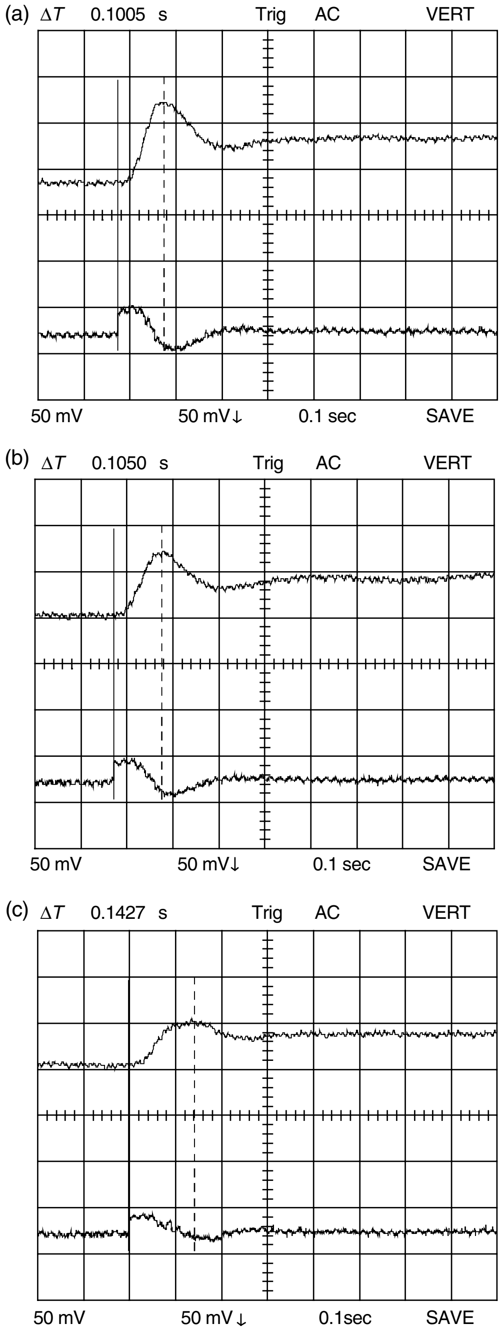


FIGURE 4.96 The measured angular speed and controller output responses for the two times increased open-loop gain ($K_S = 2$) (a) PI controller, (b) fuzzy controller, and (c) adaptive fuzzy controller. (From Kovačić, Z., Bogdan, S., and Štajdohar, M., *11th IEEE Intl. Symp. Intell. Contr.*, 49–54, 1996. With permission.)

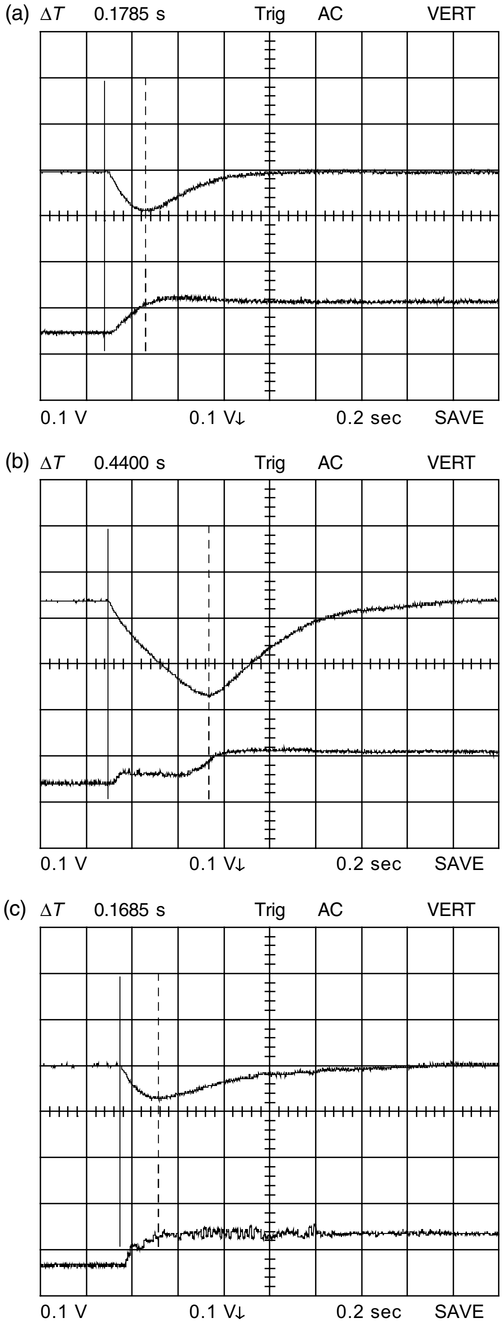


FIGURE 4.97 The measured angular speed and controller output responses for the case of stepwise load disturbance (a) PI controller, (b) fuzzy controller, and (c) adaptive fuzzy controller. (From Kovačić, Z., Bogdan, S., and Štajdohar, M., *11th IEEE Intl. Symp. Intell. Contr.*, 49–54, 1996. With permission.)

References

1. Meng, J.E. and Ya, L.S., "Hybrid fuzzy proportional-integral plus conventional derivative control of robotics systems," in *Autonomous Robotic Systems: Soft Computing and Hard Computing Methodologies and Applications*, Physica-Verlag GmbH, Heidelberg, Germany, pp. 403–427, 2003.
2. Li, W., "Design of a hybrid fuzzy logic proportional plus conventional integral-derivative controller," *IEEE Transactions on Fuzzy Systems*, 6, 449–463, 1998.
3. Akbarzadeh, M.R., Tunstel, E., Kumbla, K., and Jamshidi, M., "Soft computing paradigms for hybrid fuzzy controllers: experiments and applications," *IEEE International Conference on Fuzzy Systems, WCCI'98*, Anchorage, Alaska, pp. 1200–1205, May 1998.
4. Blanco, J.S., "Hybrid self-learning fuzzy PD+I control of unknown linear and nonlinear systems," *Fifth Mexican International Conference in Computer Science (ENC'04)*, Colima, México, pp. 233–240, 2004.
5. Blanco, J.S., Teleperm XP: The Process Control System for Economical Power Plant Control — System Overview, Siemens AG, Erlangen, 1995.
6. Zhao, Z., Tomizuka, M., and Isaka, S., "Fuzzy gain scheduling of PID controllers," *IEEE Transactions on Systems, Man and Cybernetics*, 23, 1392–1398, 1993.
7. Ling, C. and Edgar, T., "A new fuzzy gain scheduling algorithm for process control," *Proceedings of the American Control Conference ACC'92*, Chicago, Vol. 3, pp. 2284–2290, 1992.
8. Schulte, H., "Gain-scheduled control of a servo-pneumatic actuator using Takagi–Sugeno fuzzy models," in *IFAC Workshop Advanced Fuzzy-Neural Control*, Valencia, Spain, pp. 69–74, 2001.
9. Nauck, D., Klawonn, F., and Kruse, R., *Foundations of Neuro-Fuzzy Systems*, Wiley, Chichester, New York, 1997.
10. Fuller, R., *Introduction to Neuro-Fuzzy Systems*, Advances in Soft Computing, Springer-Verlag, Berlin/Heidelberg, 2000.
11. Jang, J.-S.R. and Sun, C.-T., "Neuro-fuzzy modeling and control," *IEEE Proceedings*, 83, 378–406, 1995.
12. Grabowski, P.Z., Kazmierkovski, M.P., Bose, B.K., and Blaabjerg, F., "A simple direct torque neuro-fuzzy control of PWM-inverter-fedinduction motor drive," *IEEE Transactions on Industrial Electronics*, 47, pp. 863–870, 2000.
13. Tunstel, E., Akbarzadeh, M.R., Kumbla, K., and Jamshidi, M., "Hybrid fuzzy control schemes for robotic systems," *10th IEEE International Symposium on Intelligent Control*, Monterey, USA, pp. 171–176, 1995.
14. Kovacevic, R. and Zhang, Y.M., "Neurofuzzy model-based weld fusion state estimation," *IEEE Control Systems Magazine*, 17, 30–42, 1997.
15. Lewis, F., Campos, J., and Selmic, R., *Neuro-Fuzzy Control of Industrial Systems with Actuator Nonlinearities*, SIAM Press, Philadelphia, 2002.
16. Leonhard, W., *Control of Electrical Drives*, Springer-Verlag, Hiedelberg, 1985.
17. Kovačić, Z. and Bogdan, S., "Robust angular speed control of PMSM drive with coordinated performance of nonintegral fuzzy and PI controller," *Automatika (KoREMA)*, 3–4, 99–102, 1993.
18. Vas, P., *Vector Control of AC Machines*, Clarendon Press, Oxford, 1990.
19. Pillay, P., "Application characteristics of permanent magnet synchronous and brushless dc motors for servo drives," *IEEE Transactions on Industry Applications*, 27, 986–996, 1991.

20. Krishnan, R., "Selection criteria for servo motor drives," *IEEE Transactions on Industry Applications*, IA-23, 270–275, 1987.
21. Paul, R.P., *Robot Manipulators: Mathematics, Programming and Control*, MIT Press, Cambridge, MA, 1981.
22. Draper, C.S. and Li, Y.T., *Principles of Optimalizing Control Systems and an Application to the Internal Combustion Engine*, ASME Publications, New York, 1951.
23. Astrom, K.J., "Theory and applications of adaptive control — a survey," *Automatica*, 19, 471–486, 1983.
24. Chalam, V.V., *Adaptive Control Systems — Techniques and Applications*, Marcel Dekker, New York, 1987.
25. Landau, Y.D., *Adaptive Control: The Model Reference Approach*, Marcel Dekker, New York, 1987.
26. Butler, H., *Model Reference Adaptive Control*, Prentice Hall, New York, London, 1992.
27. Whitaker, H.P., Yamron, J., and Kezer, A., *Design of a Model Reference Adaptive System for Aircraft*, R-164, Instrumentation Laboratory, MIT, Cambridge, USA, 1958.
28. Wang, L.-X., *Adaptive Fuzzy Systems and Control: Design and Stability Analysis*, Prentice-Hall, Englewood Cliffs, NJ, 1994.
29. Moore, C.G. and Harris, C.J., "Indirect adaptive fuzzy control," *International Journal of Control*, 56, 441–468, 1992.
30. Kang, H. and Vachtsevanos, G.J., "Model reference fuzzy control," *Proceedings of the 28th IEEE Conference on Decision and Control*, Tampa, pp. 751–756, 1989.
31. Layne, J.R. and Passino, K.M., "Fuzzy model reference learning control," *Journal of Intelligent and Fuzzy Systems*, 4, 33–47, 1996.
32. Yang, Y., Zhou, C., and Ren, J., "Model reference adaptive robust fuzzy control for ship steering autopilot with uncertain nonlinear systems," *Applied Soft Computing*, 3, 305–316, 2003.
33. Cho, J.-W., Park, C.-W., and Park, M., "An indirect model reference adaptive fuzzy control for SISO Takagi–Sugeno model," *Fuzzy Sets and Systems*, 131, 197–215, 2002.
34. Kovačić, Z. and Bogdan, S., "Model reference adaptive fuzzy control of high-order systems," *An International Journal for Engineering Applications of Artificial Intelligence* (Pergamon Press), 7, 501–511, 1994.
35. Kovačić, Z., Bogdan, S., and Balenović, M., "A model reference and sensitivity model-based self-learning fuzzy logic controller as a solution for control of non-linear servo systems," *IEEE Transactions on Energy Conversion*, 13, 1479–1484, 1999.
36. Emelyanov, S.V., *Variable Structure Control Systems (in Russian)*, Nauka, Moscow, U.S.S.R., 1967.
37. Wu, J.C. and Liu, T.S., "A sliding-mode approach to fuzzy control design," *IEEE Transactions on Control Systems Technology*, 4, 141–151, 1996.
38. Fantuzzi, C., "Fuzzy logic — sliding mode hybrid control," *Proceedings of the IFAC/IMACS Workshop on Artificial Intelligence in Real-Time Control AIRTIC '95*, Bled (Slovenia), 1995.
39. Chen, C.L. and Chang, M.H., "Optimal design of fuzzy sliding-mode control: a comparative study," *Fuzzy Sets and Systems*, 93, 37–48, 1998.

40. Chang, W., Park, J.B., Joo, Y.H., and Chen, G., "Design of robust fuzzy-model-based controller with sliding mode control for SISO nonlinear systems," *Fuzzy Sets and Systems*, 125, 1–22, 2002.
41. Ha, Q.P., Nguyen, Q.H., Rye, D.C., and Durrant-White, H.F., "Fuzzy sliding-mode controllers with applications," *IEEE Transactions on Industrial Electronics*, 48, 38–46, 2001.
42. Georgieva, O., Hristozov, I., Pencheva, T., Tzonkov, S., and Hitzmann, B., "Mathematical modelling and variable structure control systems for fed-batch fermentation of *Escherichia coli*," *Chemical and Biochemical Engineering Quarterly*, 17, 293–299, 2003.
43. Bellman, R.E., *Dynamic Programming*, Princeton University Press, Princeton, NJ, 1957.
44. Feldbaum, A.A., *Optimal Control Theory*, Academic Press, New York, 1960.
45. Astrom, K.J. and Wittenmark, B., *Adaptive Control*, Addison-Wesley, New York, 1995.
46. Astrom, K.J. et al., "Theory and applications of self-tuning regulators," *Automatica*, 13, 457–476, 1977.
47. Ziegler, J.G. and Nichols, N.B., "Optimum setting for automatic controllers," *Transactions on ASME*, 64, 759–768, 1942.
48. Takahashi, Y., Chan, C.S., and Auslander, D.M., "Simple discrete control of industrial processes (finite time settling control algorithm for single-loop digital controller)," *Transactions on ASME*, 97, 354–361, 1975.
49. Dahlin, E.B., "Designing and tuning digital controllers, Part I and II," *Instrumentation Control Systems*, 41, 77–83, 87–91, 1968.
50. Ioannou, P.A. and Kokotović, P.V., *Adaptive Systems with Reduced Models*, Springer Verlag, Berlin, 1983.
51. Cruz, J.B., *Feedback Systems*, McGraw-Hill, New York, 1972.
52. Frank, P.M., *Introduction to System Sensitivity Theory*, Academic Press, New York, 1978.
53. Kokotović, P.V., "Method of sensitivity points in the investigation of linear control systems," *Aut. Remote Control (U.S.S.R.)*, 12, 1512–1518, 1964.
54. Crnošija, P., Kovačić, Z., Ban, Ž., and Bogdan, S., "Sensitivity analysis and design of direct-current servo systems," *Proceedings of the XXXV Conference ETAN*, Ohrid (Macedonia), Vol. 8, pp. 43–50, 1991.
55. Crnošija, P., Bogdan, S., and Kovačić, Z., "Sensitivity model and synthesis of dead-beat algorithms in digital servosystems," *Automatica (IFAC)*, 30, 1345–1350, 1994.
56. Bogdan, S., Crnošija, P., Kovačić, Z., and Stajić, D., "Self-tuning time optimal control of servosystems by using a reference model and a sensitivity model," *Proceedings of the 4th IEEE Conference on Control Applications*, Albany, Vol. 1, pp. 730–735, 1995.
57. Kovačić, Z., Bogdan, S., and Punčec, M., "Adaptive fuzzy logic control based on integral criterion," *Proceedings of the 15th IEEE International Symposium on Intelligent Control ISIC'00*, Patras, Greece, pp. 55–60, 2000.
58. Slotine, J.J. and Sastry, S.S., "Tracking control of nonlinear system using sliding surface, with adaptation to robot manipulators," *International Journal of Control*, 38, 465–492, 1983.
59. Ioannou, P.A. and Kokotovic, P.V., "Instability analysis and improvement of robustness of adaptive control," *Automatica*, 20, 583–594, 1984.

60. Kiszska, J.B. et al., "The influence of some parameter on the accuracy of a fuzzy model," in Sugeno, M. (ed.), *Industrial Applications of Fuzzy Control*, Elsevier Science Publisher B.V., 1985.
61. Li, Y.F. and Lau, C.C., "Development of fuzzy algorithms for servo systems," *IEEE Control Systems Magazine*, 65–71, April 1989.
62. Liaw, C.M. and Wang, J.B., "Design and implementation of a fuzzy controller for a high performance induction motor drive," *IEEE Transactions on Systems, Man and Cybernetics*, 21, 921–929, 1991.
63. Quadrado, J.C. and Silva, J.F., "On the elimination of steady-state errors with an 'elastic' fuzzy position controller for motor drives," *Proceedings of the 19th IEEE Annual International Conference on Industrial Electronics, Control and Instrumentation IECON'93*, Lahaina-Maui, Vol. 1, pp. 194–199, 1993.
64. Ko, J.S. et al., "Robust position control of BLDD motors using integral-proportional plus fuzzy logic controller," *Proceedings of the 19th IEEE Annual International Conference on Industrial Electronics, Control and Instrumentation IECON'93*, Lahaina-Maui, Vol. 1, pp. 213–218, 1993.
65. Layne, J.R. and Passino, K.M., "Fuzzy model reference learning control for cargo ship steering," *Proceedings of the 8th IEEE International Symposium on Intelligent Control*, Chicago, pp. 457–462, 1993.
66. Passino, K.M. and Yurkovic, S., *Fuzzy Control*, Addison Wesley, New York, USA, 1998.
67. Kovačić, Z., Bogdan, S., and Crnošija, P., "Fuzzy rule-based model reference adaptive control of permanent magnet synchronous motor drive," *Proceedings of the 19th IEEE Annual International Conference on Industrial Electronics, Control and Instrumentation*, Vol. 1, pp. 207–212, Lahaina-Maui, 1993.
68. Orlowska-Kowalska, T., Szabat, K., and Jaszczak, K., "The influence of parameters and structure of PI-type fuzzy-logic controller on DC drive system dynamics," *Fuzzy Sets and Systems*, 131, 251–264, 2002.
69. Kovačić, Z., "Adaptive fuzzy control with model reference-based fuzzy adaptation mechanism," *Proceedings of the 2nd IEEE Conference on Control Applications*, Vancouver, Vol. 1, pp. 189–195, 1993.
70. Kovačić, Z., Bogdan, S., and Reichenbach, T., "Nonlinear position control by using multiple position dependent self-organizing fuzzy logic controllers," *6th IFAC Symposium on Robot Control SYROCO'00*, Vienna, Austria, pp. 229–233, 2000.
71. Hirzinger, G. and Heindl, J., "Sensor programming — a new way for teaching robot paths and forces/torques simultaneously," *Proceedings of the Third International Conference on Robot Vision and Sensory Controls*, Cambridge, MA, November 1983.
72. Plank, G. and Hirzinger, G., "Controlling a robot's motion speed by a force-torque-sensor for deburring problems," *Proceedings of the 4th IFAC/IFIP Symposium on Information Control Problems in Manufacturing Technologies*, pp. 97–102, October 1982.
73. Mills, J.K. and Lokhorst, D.M., "Stability and control of robotic manipulators during contact/noncontact task transition," *IEEE Transactions on Robotics and Automation*, 9, 335–345, 1993.
74. Luh, J.Y.S., Fisher, W.D., and Paul, R.P.C., "Joint torque control by a direct feedback for industrial robots," *IEEE Transactions on Automatic Control*, AC-28, 153–161, 1983.

75. Wu, C.H. and Paul, R.P., "Manipulator compliance based on joint torque control," *Proceedings of the 20th IEEE Conference on Decision Control*, Vol. 1, pp. 265–270, December 1981.
76. Craig, J.J. and Raibert, M.H., "A systematic method of hybrid position/force control of a manipulator," *Proceedings of the IEEE Computer Software and Applications Conference*, pp. 446–451, November 1979.
77. Raibert, M.H. and Craig, J.J., "Hybrid position/force control of manipulators," *IEEE Transactions on Systems, Man and Cybernetics*, SMC-II, 418–432, 1981.
78. Yoshikawa, T. and Sudou, A., "Dynamic hybrid position/force control of robot manipulators — on-line estimation of unknown constraint," *IEEE Transactions on Robotics and Automation*, 9, 1993.
79. Bogdan, S. and Kovačić, Z., "Fuzzy rule-based adaptive force control of single DOF servo mechanisms," *Automatika (KoREMA)*, 3–4, 119–129, 2002.
80. Kovačić, Z., Bogdan, S., and Štajdohar, M., "Servo application of a model reference-based adaptive fuzzy control," *Proceedings of the 11th IEEE International Symposium on Intelligent Control*, Dearborn, pp. 49–54, 1996.
81. Kovačić, Z., Bogdan, S., and Markić, M., "Realization of a low-cost angular speed fuzzy controller," *Proceedings of the 2nd European Congress on Fuzzy and Intelligent Technologies*, Aachen, pp. 274–277, 1994.
82. Kovačić, Z., Bogdan, S., and Punčec, M., "Adaptive control based on sensitivity model-based adaptation of lead-lag compensator parameters," *The CD-ROM proceedings of the IEEE International Conference on Industrial Technology ICIT'03*, Maribor, pp. 321–326, 2003.

5 Self-Organizing Fuzzy Controllers

In order to ease the burden of heuristic adjustment of fuzzy controller parameters, there are few alternatives to choose from. The first one is to use specialized tools for graphically oriented heuristic development of fuzzy control systems [1,2]. They give very good results when basic fuzzy logic algorithms are considered, but problems may arise when more complex or specific control structures are to be developed. Another alternative is to synthesize fuzzy logic controllers as self-organizing units capable of using given control quality indicators and acquired process data as a basis for iterative self-creation of the algorithm core. An automated fuzzy controller design in the form of fuzzy rule-base self-organization, as introduced in Mamdani's early works [3], has been further elaborated by many authors denoting their self-organizing schemes also as self-tuning, self-learning, adaptive, and expert algorithms or simply as fuzzy logic algorithms with a varying rule base [4–10]. In general, the fuzzy controller can learn either off-line or on-line from identified control system parameters, a given reference model or from selected performance indices [11,12]. The off-line self-organization ends with a static nonlinear mapping function prepared for on-line operation. The on-line self-organization results in dynamic changes of a nonlinear mapping function as directed by the control system performance.

A self-organization of fuzzy controllers can also be accomplished by using neural networks, which have learning capability. Such neuro-fuzzy solutions often use cost functions which need to be optimized. In such a case, stagnation in local optima arises as a typical problem for gradient-based methods [13–16].

In fuzzy model reference learning control (FMRLC) schemes, model tracking error e_{M} is fed into a learning mechanism, which establishes an organized iterative way of changing the core of the fuzzy controller (Figure 5.1). By changing controller's parameters, the self-organization process enforces the fuzzy control system to follow the given reference model dynamics [17–19]. To make a distinction between iterative learning and iterative adaptation control schemes it must be emphasized that the proper work of fuzzy learning schemes does not depend on a mathematical model of a control process, though any knowledge about the control process may be helpful, especially with the determination of the reference model function or with the estimation of normalizing scaling factors of the fuzzy controller (related to the estimated dimensions of the universes of discourse).

A fuzzy learning concept based on the use of a fuzzy inverse model and a knowledge base modifier has been successfully developed and tested by simulation for cargo ship steering [20,21], showing a superior performance of the

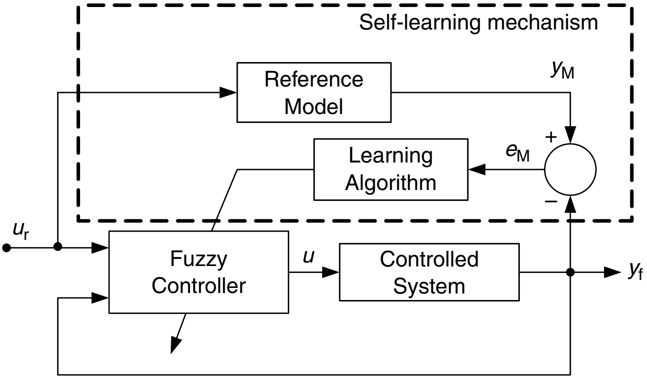


FIGURE 5.1 A structure of a model reference self-learning fuzzy control system.

FMRLC scheme compared to several conventional MRAC schemes. The fuzzy inverse model performs the function of mapping the model tracking error and the changes in the control effort. The knowledge base modifier enforces the changes of the knowledge base, which should provide a required control effort. The structure of the fuzzy inverse model resembles the structure of a standard PD-type fuzzy controller whose parameters are synthesized (optimized) in a heuristic way relying on the rough knowledge about the plant inverse model. Recalling the discussion about PD-type fuzzy adaptation mechanisms in Chapter 4, one may notice that such adaptation mechanisms are actually the synonyms for the fuzzy inverse model of the plant. In Reference 22, a comparative study of several control strategies, which enhance the performance of the FMRLC has been worked out, showing that the so-called "dynamically focused learning" may contribute to more effective solving of demanding control problems such as the magnetic ball suspension system control.

The self-learning fuzzy controller, which utilizes a sensitivity model and a second-order reference model, has been effectively introduced for control of nonlinear control systems [23]. In order to determine a sensitivity function, the function describing the dependence of two variables (or variables and parameters) must be differentiable. As shown in Reference 23, there is a class of fuzzy controllers widely used in practical applications, usually called the Takagi–Sugeno zero-order (or singleton) fuzzy controllers, which can be organized to assume an analytical and differentiable form.

The self-organization of a fuzzy controller should be a stable and fast convergent process. Most of the stability assessment methods developed for nonlinear control systems described by nonlinear differential or difference equations have not been quite appropriate for stability assessment of rule-based nonlinear systems. Since the structure of a model reference-based self-organizing fuzzy controller resembles the structure of standard model reference adaptive controllers, there have been attempts to apply well-known stability assessment approaches.

In Reference 24, the FLC form has been obtained by applying the Lyapunov function-based stability criteria, but learning required data from the fuzzy identification algorithm. Learning based on the on-line identification is too slow for control of systems with high dynamics (e.g., servo systems). In this chapter we will describe three model reference-based self-learning concepts, one based on the direct Lyapunov method, another with a learning mechanism that utilizes a second-order reference model and a polynomial of the model tracking error, and the third one where learning depends on the second-order reference model and a sensitivity model related to the fuzzy controller parameters.

5.1 Self-Organizing Fuzzy Control Based on the Direct Lyapunov Method

Let us consider a SISO process described with the following equation:

$$y(k) = \mathbf{\Lambda}^{\mathrm{T}}\mathbf{z}(k-1) + b_1 u(k-1) \tag{5.1}$$

where $y(k)$ is the process output; $\mathbf{\Lambda}$, the process parameter vector; b_1, the process input gain ($b_1 > 0$); $\mathbf{z}(k)$, the vector containing measurement signals or signals that can be derived from measurement signals; and $u(k)$ is the control signal.

Since elements of $\mathbf{z}(k)$ could be a nonlinear combination of signals, a process described with (5.1) may be nonlinear but it should be linear in respect to the control signal and process parameters.

We define a controller as:

$$u(k) = \boldsymbol{\theta}^{\mathrm{T}}(k)\mathbf{w}(k) \tag{5.2}$$

where $\boldsymbol{\theta}(k)$ is the controller parameter vector and $\mathbf{w}(k)$, the vector containing measurement signals or signals that can be derived from measurement signals.

While the controller structure is predefined and time invariant, controller parameters are changing with time. Vector $\mathbf{w}(k)$ could contain nonlinear combination of process signals, which means that in general, a controller described with (5.2), may be nonlinear.

By including (5.2) into (5.1) one obtains a closed-loop system equation:

$$y(k) = \mathbf{\Lambda}^{\mathrm{T}}\mathbf{z}(k-1) + b_1\boldsymbol{\theta}^{\mathrm{T}}(k-1)\mathbf{w}(k-1) \tag{5.3}$$

As vectors $\mathbf{z}(k)$ and $\mathbf{w}(k)$ could have common parameters let us split them in the following way:

$$\mathbf{z}(k) = \begin{bmatrix} \mathbf{z}_1(k) \\ \mathbf{z}_2(k) \end{bmatrix}, \quad \mathbf{w}(k) = \begin{bmatrix} \mathbf{w}_1(k) \\ \mathbf{w}_2(k) \end{bmatrix}, \quad \mathbf{z}_2(k) = \mathbf{w}_2(k) \tag{5.4}$$

If we also split process parameter vector $\boldsymbol{\Lambda}$ and controller parameter vector $\boldsymbol{\theta}(k)$ as:

$$\boldsymbol{\Lambda}(k) = \begin{bmatrix} \boldsymbol{\Lambda}_1(k) \\ \boldsymbol{\Lambda}_2(k) \end{bmatrix}, \qquad \boldsymbol{\theta}(k) = \begin{bmatrix} \boldsymbol{\theta}_1(k) \\ \boldsymbol{\theta}_2(k) \end{bmatrix} \tag{5.5}$$

then, having in mind that $\mathbf{z}_2(k) = \mathbf{w}_2(k)$, we may describe the closed-loop system as:

$$y(k) = \begin{bmatrix} \boldsymbol{\Lambda}_1^{\mathrm{T}} & [b_1\boldsymbol{\theta}_1(k-1)]^{\mathrm{T}} & [\boldsymbol{\Lambda}_2 + b_1\boldsymbol{\theta}_2(k-1)]^{\mathrm{T}} \end{bmatrix} \begin{bmatrix} \mathbf{z}_1(k-1) \\ \mathbf{w}_1(k-1) \\ \mathbf{w}_2(k-1) \end{bmatrix} \tag{5.6}$$

The goal of an adaptive controller, based on a reference model, is to enforce that tracking error $e_{\mathrm{M}}(k) \to 0$ as $k \to \infty$.

Instead of using a standard reference model, we are using a projection of vectors $\mathbf{z}(k)$ and $\mathbf{w}(k)$ to determine $y_{\mathrm{M}}(k)$:

$$y_{\mathrm{M}}(k) = \begin{bmatrix} \boldsymbol{\Lambda}_{\mathrm{M}1}^{\mathrm{T}} & \boldsymbol{\Lambda}_{\mathrm{M}2}^{\mathrm{T}} & \boldsymbol{\Lambda}_{\mathrm{M}3}^{\mathrm{T}} \end{bmatrix} \begin{bmatrix} \mathbf{z}_1(k-1) \\ \mathbf{w}_1(k-1) \\ \mathbf{w}_2(k-1) \end{bmatrix} \tag{5.7}$$

where $\boldsymbol{\Lambda}_{\mathrm{M}}$ is the vector of reference model parameters.

For the tracking error asymptotically approaching zero the following relations have to be satisfied:

$$\begin{aligned} &\boldsymbol{\Lambda}_1 = \boldsymbol{\Lambda}_{\mathrm{M}1} \\ &b_1\boldsymbol{\theta}_1(k-1) \to b_1\boldsymbol{\theta}_{10} = \boldsymbol{\Lambda}_{\mathrm{M}2} \quad \text{as } k \to \infty \\ &\boldsymbol{\Lambda}_2 + b_1\boldsymbol{\theta}_2(k-1) \to \boldsymbol{\Lambda}_2 + b_1\boldsymbol{\theta}_{20} = \boldsymbol{\Lambda}_{\mathrm{M}3} \quad \text{as } k \to \infty \end{aligned} \tag{5.8}$$

where

$$\boldsymbol{\theta}_0 = \begin{bmatrix} \boldsymbol{\theta}_{10} \\ \boldsymbol{\theta}_{20} \end{bmatrix}$$

represents a controller parameter vector, which enforces the tracking error $e_{\mathrm{M}}(k)$ to vanish.

Having a reference model defined with (5.7) and by using (5.6)–(5.8) to calculate $e_{\mathrm{M}}(k)$, it follows:

$$e_{\mathrm{M}}(k) = b_1[\boldsymbol{\theta}_0 - \boldsymbol{\theta}(k-1)]^{\mathrm{T}}\mathbf{w}(k-1) \tag{5.9}$$

One can see from (5.9) that tracking error $e_{\mathrm{M}}(k)$ will disappear when controller parameter vector $\boldsymbol{\theta}(k)$ will assume value $\boldsymbol{\theta}_0$, so the process will follow the reference model.

A problem related to determination of the controller defined with relation (5.2) lies in the fact that the process parameter vector $\mathbf{\Lambda}$ is generally unknown; hence vector $\boldsymbol{\theta}_0$ cannot be calculated. That is why an adaptation algorithm should be used for controller parameter vector $\boldsymbol{\theta}(k)$ to approach to its final value $\boldsymbol{\theta}_0$.

Now let us consider a learning (tuning) algorithm design. The main objective of the controller design is to guarantee closed-loop stability. For this purpose, let us define a Lyapunov function candidate as:

$$V(k) = [\boldsymbol{\theta}_0 - \boldsymbol{\theta}(k)]^{\mathrm{T}}[\boldsymbol{\theta}_0 - \boldsymbol{\theta}(k)] \tag{5.10}$$

In order to set conditions for $e_{\mathrm{M}}(k)$ to be bounded, we require that $V(k)$ must be bounded:

$$\Delta V(k) = V(k) - V(k-1) \leq 0 \tag{5.11}$$

From (5.10) and (5.11) it follows that:

$$\Delta V(k) = -2[\boldsymbol{\theta}_0 - \boldsymbol{\theta}(k-1)]^{\mathrm{T}}\Delta\boldsymbol{\theta}(k) + \Delta\boldsymbol{\theta}(k)^{\mathrm{T}}\Delta\boldsymbol{\theta}(k) \tag{5.12}$$

If we choose the parameters tuning law to be

$$\Delta\boldsymbol{\theta}(k) = \frac{\gamma e_{\mathrm{M}}(k)}{\alpha + \mathbf{w}(k-1)^{\mathrm{T}}\mathbf{w}(k-1)}\mathbf{w}(k-1) \tag{5.13}$$

then, with relation (5.9) in mind, Equation (5.12) can be written as:

$$\Delta V(k) = \gamma e_{\mathrm{M}}^2(k)\frac{1}{\alpha + \mathbf{w}(k-1)^{\mathrm{T}}\mathbf{w}(k-1)}\left[\gamma\frac{\mathbf{w}(k-1)^{\mathrm{T}}\mathbf{w}(k-1)}{\alpha + \mathbf{w}(k-1)^{\mathrm{T}}\mathbf{w}(k-1)} - \frac{2}{b_1}\right] \tag{5.14}$$

Values for γ and α, which fulfil condition (5.11), can be determined by using Equation (5.14):

$$0 < \gamma < \frac{2}{b_1}, \quad \alpha \geq 0 \tag{5.15}$$

Even though algorithm (5.13), with tuning parameters γ and α defined by (5.15), guarantees the stability of the closed-loop system, there could still be a problem related to signals $\mathbf{w}(k)$ and $e_{\mathrm{M}}(k)$. These signals are obtained by measurement, which means that they may contain a noise of unknown level and spectrum. For a practical implementation of the algorithm, filtering of $\mathbf{w}(k)$ and

$e_{\mathrm{M}}(k)$ is necessary:

$$e_{M_{\mathrm{f}}}(k) = -\sum_{i=1}^{n_f} a_{\mathrm{f}_i} e_{M_{\mathrm{f}}}(k-i) + e_{\mathrm{M}}(k) \tag{5.16}$$

$$\mathbf{w}_{\mathrm{f}}(k) = -\sum_{i=1}^{n_f} a_{\mathrm{f}_i} \mathbf{w}_{\mathrm{f}}(k-i) + \mathbf{w}(k-1) \tag{5.17}$$

After multiplying (5.17) from the left-hand side with $b_1[\boldsymbol{\theta}_0 - \boldsymbol{\theta}(k-1)]^{\mathrm{T}}$, we get:

$$\varepsilon(k) = -\sum_{i=1}^{n_f} a_{\mathrm{f}_i} \varepsilon(k-i) + e_{\mathrm{M}}(k) + \eta(k) \tag{5.18}$$

where

$$\varepsilon(k) = b_1[\boldsymbol{\theta}_0 - \boldsymbol{\theta}(k-1)]^{\mathrm{T}} \mathbf{w}_{\mathrm{f}}(k) \tag{5.19}$$

$$\eta(k) = b_1 \sum_{i=1}^{n_f} a_{\mathrm{f}_i} \sum_{j=1}^{i} \Delta\boldsymbol{\theta}(k-j)^{\mathrm{T}} \mathbf{w}_{\mathrm{f}}(k-i) \tag{5.20}$$

It can be seen that expressions (5.19) and (5.9) are similar in form. If we replace $e_{\mathrm{M}}(k)$ and $\mathbf{w}(k-1)$ in (5.13) with $\varepsilon(k)$ and $\mathbf{w}_{\mathrm{f}}(k)$, the learning law will assume the form

$$\Delta\boldsymbol{\theta}(k) = \frac{\gamma \varepsilon(k)}{\alpha + \mathbf{w}_{\mathrm{f}}(k)^{\mathrm{T}} \mathbf{w}_{\mathrm{f}}(k)} \mathbf{w}_{\mathrm{f}}(k) \tag{5.21}$$

while conditions for determination of tuning parameters γ and α will remain the same.

The main problem arises when calculating $\varepsilon(k)$. From (5.20) we may find that for small changes of controller parameters (small value of $\Delta\boldsymbol{\theta}$), $\eta(k)$ could be neglected, that is, $\eta(k) \approx 0$, which means that $\varepsilon(k)$ could be substituted with $e_{M_{\mathrm{f}}}(k)$. This leads to the final form of the controller parameters learning (tuning) algorithm:

$$\Delta\boldsymbol{\theta}(k) = \frac{\gamma e_{M_{\mathrm{f}}}(k)}{\alpha + \mathbf{w}_{\mathrm{f}}(k)^{\mathrm{T}} \mathbf{w}_{\mathrm{f}}(k)} \mathbf{w}_{\mathrm{f}}(k) \tag{5.22}$$

Example 5.1 Direct Lyapunov method-based self-organizing fuzzy control of a positioning servo system.

Let us show the effectiveness of a direct Lyapunov method-based fuzzy controller design on the problem of controlling a nonlinear positioning servo system affected by friction and nonlinear gravity-dependent load. These nonlinearities are generally considered as difficult to deal with, especially in control of robotic mechanisms [25–31].

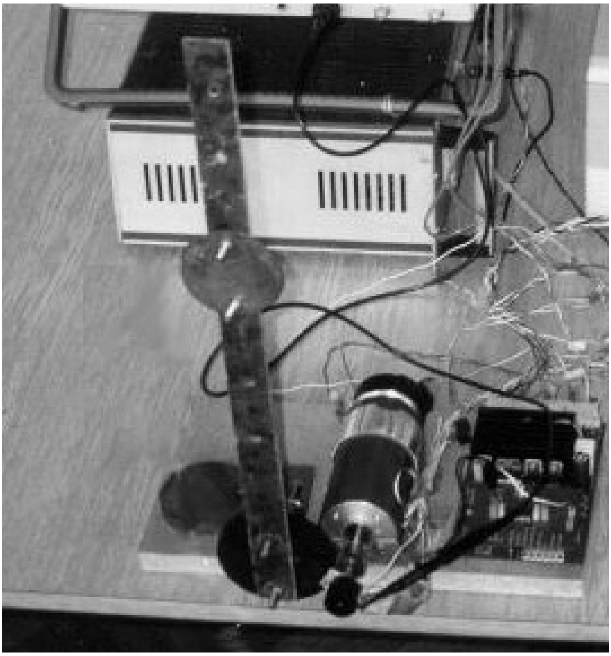


FIGURE 5.2 A DCM positioning servo system.

The target system shown in Figure 5.2 contains a chopper-controlled DC motor, a feedback potentiometer, a nonlinear load, and a personal computer with an A/D–D/A card.

The positioning DCM servo drive can be described with the following set of equations:

$$\begin{aligned}
U_{\mathrm{a}}(s) &= K_{\mathrm{c}}U(s)\\
E(s) &= K\Omega(s)\\
I_{\mathrm{a}}(s) &= K_{\mathrm{a}}[U_{\mathrm{a}}(s) - E(s)]\\
T_{\mathrm{m}}(s) &= KI_{\mathrm{a}}(s)\\
T(s) &= T_{\mathrm{m}}(s) - T_{\mathrm{l}} - T_{\mathrm{fr}}\\
T(s) &= Js\Omega(s)\\
Y_{\mathrm{f}}(s) &= \frac{K_y}{s}\Omega(s)\\
T_{\mathrm{l}} &= f_1(y_{\mathrm{f}})\\
T_{\mathrm{fr}} &= f_2(\omega)
\end{aligned} \tag{5.23}$$

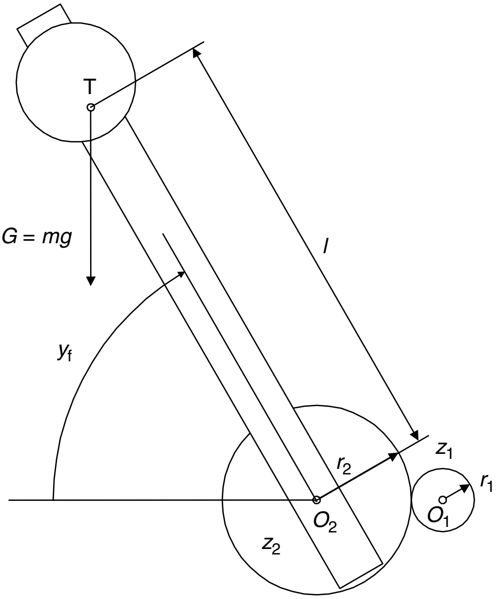


FIGURE 5.3 The nonlinear load.

where $U(s)$ is the controller output [V]; $U_a(s)$, the armature voltage [V]; $I_a(s)$, the armature current [A]; T_m, the motor torque [Nm]; T_l, the load torque [Nm]; T_{fr}, the friction torque [Nm]; T, the shaft torque [Nm], ω; Ω the angular speed [rad/sec]; E, the counter-electromotive force [V]; Y_f, the system output (the bar position) [°]; K_c, the chopper gain [V/V]; K_a, the armature constant [A/V]; K, the motor constant [Nm/A]; J, the moment of inertia [kg m^2]; and $K_y = 180/\pi$.

A nonlinear load (Figure 5.3) is described with:

$$T_l = f_l(y_f) = \frac{mgl}{N}\cos\left(y_f\frac{\pi}{180}\right) = T_{l0}\cos\left(y_f\frac{\pi}{180}\right) \tag{5.24}$$

where m is the bar and burden mass [kg]; g, the gravitational constant [m/sec^2]; l, the distance between the shaft and the center of the bar and burden gravity; N, the gear ratio; and T_{l0}, the maximal load.

Even though the nonlinear load function f_l is deterministic, the parameters m and l usually change in time, especially in the case of robot control systems. That is why we consider f_l as an unknown function.

Same as for the nonlinear load, the friction torque characteristic $T_{fr} = f_2(\omega)$, which is shown in Figure 5.4, is unknown.

In order to obtain the description of the system in the form (5.1), the process is divided on the linear part and the nonlinear part, as shown in Figure 5.5.

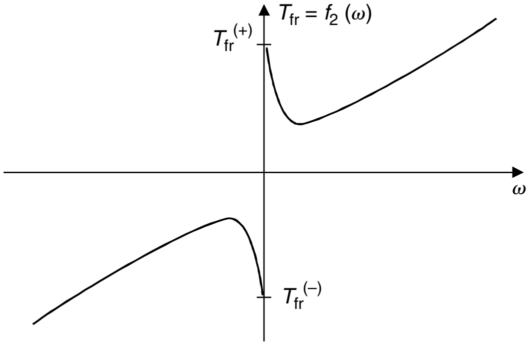


FIGURE 5.4 The friction torque characteristic.

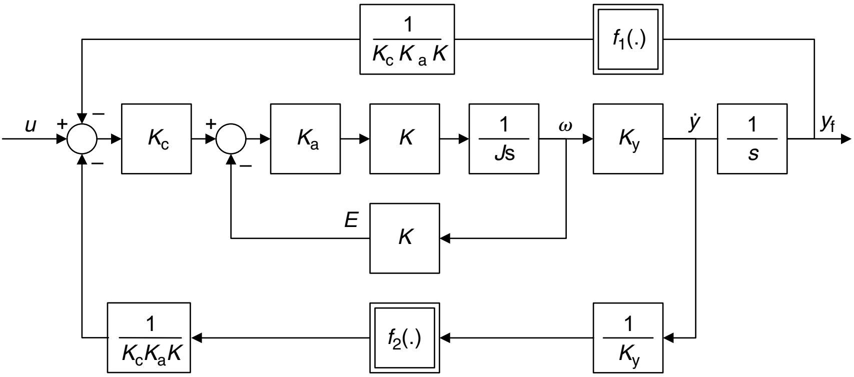


FIGURE 5.5 The block diagram of the modified process.

The process can be described with the following nonlinear differential equation:

$$\tau_1 \frac{\mathrm{d}^2 y_f(t)}{\mathrm{d}t^2} + \frac{\mathrm{d}y_f(t)}{\mathrm{d}t} = K_1 \left\{ u(t) - \frac{1}{K_c K_a K} \left[f_1(y_f(t)) + f_2\left(\frac{1}{K_y} \frac{\mathrm{d}y_f(t)}{\mathrm{d}t} \right) \right] \right\} \tag{5.25}$$

where $K_1 = K_c K_y / K$ and $\tau_1 = J/(K_a K^2)$.

In the case that we know $f_1(\cdot)$ and $f_2(\cdot)$, by choosing a control law

$$u(t) = K_p e(t) - K_d \frac{\mathrm{d}y_f(t)}{\mathrm{d}t} + \frac{1}{K_c K_a K} f_1[y_f(t)] + \frac{1}{K_c K_a K} f_2 \left[\frac{1}{K_y} \frac{\mathrm{d}y_f(t)}{\mathrm{d}t} \right] \tag{5.26}$$

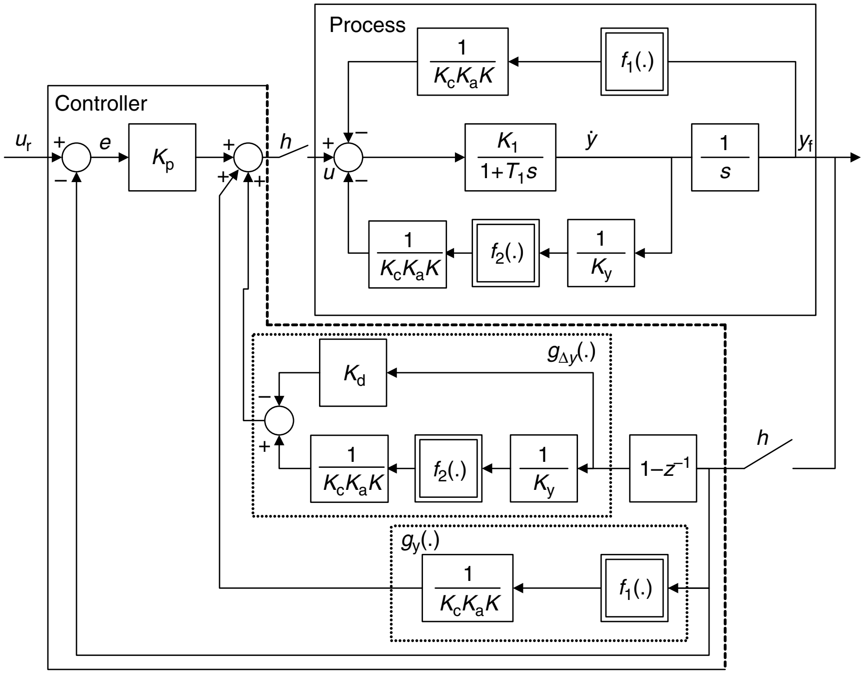


FIGURE 5.6 The structure of the nonlinear controller and process.

where $e(t) = u_r(t) - y_f(t)$, we obtain the closed-loop system

$$\frac{\tau_1}{K_1K_p}\frac{d^2y_f(t)}{dt^2} + \frac{1+K_1K_d}{K_1K_p}\frac{dy_f(t)}{dt} + y_f(t) = u_r(t) \tag{5.27}$$

which allows determination of dynamics by tuning derivative gain K_d and proportional gain K_p. The structure of the nonlinear controller and the process is shown in Figure 5.6.

Since the controller is realized in the discrete technique, instead of continuous signal dy_f/dt we use (T_d is the sampling interval):

$$\left.\frac{dy_f(t)}{dt}\right|_{t=kT_d} = \frac{y_f(k) - y_f(k-1)}{T_d} = \frac{\Delta y_f(k)}{T_d} \tag{5.28}$$

Now we may write a discrete form of the process (5.25)

$$\begin{aligned} y_f(k) = {} & -a_1y_f(k-1) - a_2y_f(k-2) - b_1\frac{1}{K_cK_aK}f_1[y_f(k-1)] \\ & - b_1\frac{1}{K_cK_aK}f_2\left[\frac{1}{K_y}\frac{\Delta y_f(k-1)}{T_d}\right] + b_1u(k-1) \end{aligned} \tag{5.29}$$

where

$$a_1 = -(1 + \mathrm{e}^{T_\mathrm{d}/\tau_1}), \quad a_2 = \mathrm{e}^{T_\mathrm{d}/\tau_1}, \quad b_1 = K_1 T_\mathrm{d}(1 + \mathrm{e}^{T_\mathrm{d}/\tau_1}) \tag{5.30}$$

Same as in the continuous case, by defining a discrete form of the controller (5.26) we may obtain desired closed-loop system behavior by choosing K_p and K_d. The problem is in generally unknown $f_1(\cdot)$ and $f_2(\cdot)$, which could be solved by controller adaptation.

Let us introduce a control law of the following form:

$$u(k) = K_\mathrm{p}(k)e(k) + g_y\{k \mid y_\mathrm{f}(k)\} + g_{\Delta y}\{k \mid \Delta y_\mathrm{f}(k)\} \tag{5.31}$$

where

$$g_y\{k \mid y_\mathrm{f}(k)\} \to \frac{1}{K_\mathrm{c}K_\mathrm{a}K} f_1[y_\mathrm{f}(k)] \quad \text{as } k \to \infty \tag{5.32}$$

$$g_{\Delta y}\{k \mid \Delta y_\mathrm{f}(k)\} \to \frac{1}{K_\mathrm{c}K_\mathrm{a}K} f_2\left[\frac{1}{K_y}\frac{\Delta y_\mathrm{f}(k)}{T_\mathrm{d}}\right] - K_\mathrm{d}\frac{\Delta y_\mathrm{f}(k)}{T_\mathrm{d}} \quad \text{as } k \to \infty \tag{5.33}$$

From (5.33) it may be seen that the derivative gain K_d is included in the system as a part of the function $g_{\Delta y}(\cdot)$ (see Figure 5.6).

Having in mind that fuzzy algorithms can be used as universal approximators [32,33], let us generate functions $g_y(\cdot)$ and $g_{\Delta y}(\cdot)$ by using fuzzy systems:

$$g_y\{y_\mathrm{f}(k)\} = \sum_{i=1}^{n_y} A_{yi}(k)\varphi_{yi}(k), \qquad g_{\Delta y}\{\Delta y_\mathrm{f}(k)\} = \sum_{i=1}^{n_{\Delta y}} A_{\Delta yi}(k)\varphi_{\Delta yi}(k) \tag{5.34}$$

where A_{yi}, $A_{\Delta yi}$ are the output singletons triggered by ith fuzzy rules; n_y, $n_{\Delta y}$, the numbers of rules; and φ_{yi}, $\varphi_{\Delta yi}$, the fuzzy basis functions of ith fuzzy rules.

The distribution of fuzzy membership functions for y_f and Δy_f are shown in Figure 5.7. Membership functions for both variables have a triangular form. While the distribution for y_f is linear, the distribution for Δy_f is nonlinear, that is, finer for smaller values of the variable.

The controller (5.31) can be written in a matrix form (5.2):

$$u(k) = \boldsymbol{\theta}^\mathrm{T}(k)\mathbf{w}(k) = \begin{bmatrix} K_\mathrm{p}(k) & A_y^\mathrm{T}(k) & A_{\Delta y}^\mathrm{T}(k) \end{bmatrix} \begin{bmatrix} e(k) \\ \varphi_y(k) \\ \varphi_{\Delta y}(k) \end{bmatrix} \tag{5.35}$$

From (5.35) it can be seen that the tuning algorithm has to tune proportional gain K_p and output singletons of fuzzy algorithms that emulate nonlinear functions $g_y(\cdot)$ and $g_{\Delta y}(\cdot)$.

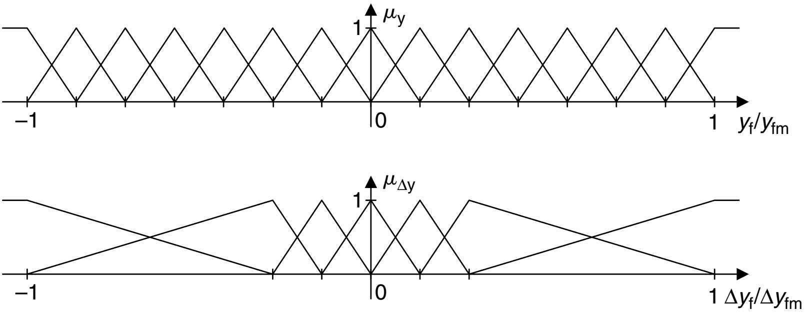


FIGURE 5.7 Distribution of fuzzy membership functions. (From Kovačić, Z., Cupec, R., and Bogdan, S., *IFAC System Structure and Control 2001*, Vol. 2. With permission from Elsevier.)

In order to smooth the signals involved in the tuning algorithm, a discrete second-order Butterworth filter is used:

$$G_f(z) = \frac{1}{1 + a_{f1}z^{-1} + a_{f2}z^{-2}} \tag{5.36}$$

where $a_{f1} = -2e^{-\zeta_f\omega_f T_d}\cos\left(\omega_f T_d\sqrt{1-\zeta_f^2}\right)$, $a_{f2} = e^{-2\zeta_f\omega_f T_d}$, with ζ_f as a desired filter damping ratio and ω_f as a desired filter resonance frequency.

From (5.16), (5.17), and (5.36) we obtain

$$\begin{aligned} e_{M_f}(k) &= -a_{f1}e_{M_f}(k-1) - a_{f2}e_{M_f}(k-2) + e_M(k) \\ e_f(k) &= -a_{f1}e_f(k-1) - a_{f2}e_f(k-2) + e(k) \\ \varphi_{y_f}(k) &= -a_{f1}\varphi_{y_f}(k-1) - a_{f2}\varphi_{y_f}(k-2) + \varphi_y(k) \\ \varphi_{\Delta y_f}(k) &= -a_{f1}\varphi_{\Delta y_f}(k-1) - a_{f2}\varphi_{\Delta y_f}(k-2) + \varphi_{\Delta y}(k) \end{aligned} \tag{5.37}$$

Upon including these variables in vector $\mathbf{w}_f$ and for $\alpha = 0$, we get the final form of tuning (learning) algorithm (5.22):

$$\begin{aligned} K_p(k) &= K_p(k-1) + \frac{\gamma e_{M_f}(k)}{\mathbf{w}_f(k)^T\mathbf{w}_f(k)}e_f(k) \\ A_y(k) &= A_y(k-1) + \frac{\gamma e_{M_f}(k)}{\mathbf{w}_f(k)^T\mathbf{w}_f(k)}\varphi_{y_f}(k) \\ A_{\Delta y}(k) &= A_{\Delta y}(k-1) + \frac{\gamma e_{M_f}(k)}{\mathbf{w}_f(k)^T\mathbf{w}_f(k)}\varphi_{\Delta y_f}(k) \end{aligned} \tag{5.38}$$

where

$$\mathbf{w}_{\mathrm{f}}(k) = \begin{bmatrix} e_{\mathrm{f}}(k) \\ \varphi_{y_{\mathrm{f}}}(k) \\ \varphi_{\Delta y_{\mathrm{f}}}(k) \end{bmatrix} \tag{5.39}$$

The desired closed-loop system behavior is described with a linear second-order reference model, $\zeta_{\mathrm{M}} = 0.8$, $\omega_{\mathrm{M}} = 7\ \mathrm{sec}^{-1}$.

We treat the process parameters as unknown as well as the parameters that define the nonlinear functions $g_y(\cdot)$ and $g_{\Delta y}(\cdot)$. In order to make the learning algorithm (5.38) stable it is necessary to find boundaries of the learning coefficient γ. For that purpose we must identify dynamics of the linear part of the process (5.29), that is, we must identify parameters K_1 and τ_1. From data obtained by implementation of a stepwise signal to the process input we have found that $K_1 = 110°/\mathrm{Vsec}$ and $\tau_1 = 0.12$ sec.

The controller parameters are as follows: $\zeta_{\mathrm{f}} = 0.707$, $\omega_{\mathrm{f}} = 25\ \mathrm{sec}^{-1}$, $y_{\mathrm{f}_m} = 140°$, $\Delta y_{\mathrm{f}_m} = 3.5°\ \mathrm{sec}^{-1}$, $T_{\mathrm{d}} = 0.02$ sec. To keep the learning process in the stable region Equation (5.15) must be fulfilled. Based on the estimated value of parameter b_1, we have chosen $\gamma = 0.5$.

In order to overcome the problem of over-learning, a dead-zone has been introduced into the learning algorithm. Being in a steady state (reference signal u_{r} is not persistent), the learning algorithm will remain idle until variable $e(k)$ becomes larger than threshold value e_{th}. We have adopted $e_{\mathrm{th}} = 0.5°$.

While in learning algorithm (5.38), the fuzzy basis functions obtain values between 0 and 1, variable $e_{\mathrm{f}}(k)$, which is the component of vector $\mathbf{w}_{\mathrm{f}}(k)$, can attain value that is several times larger than 1. This may cause a disproportional tuning effect, which would end up with values of tuneable parameter K_{p} several times larger than the values of fuzzy output singletons $\mathbf{A}_y$ and $\mathbf{A}_{\Delta y}$. If we allow this, the influence of fuzzy algorithms that emulate $g_y(\cdot)$ and $g_{\Delta y}(\cdot)$ would be insignificant and the quality of control would deteriorate. This problem has been solved by scaling variable $e_{\mathrm{f}}(k)$ with respect to its expected maximum value $e_{\mathrm{f}_m} = 20°$.

The experimental results obtained at the beginning and at the end of learning procedure for the set point change from −90 to −60° are shown in Figure 5.8 and Figure 5.9, respectively. As may be seen from Figure 5.8, initial values of parameter vectors $\mathbf{A}_y$ and $\mathbf{A}_{\Delta y}$ and the initial value of parameter K_{p} are set to 0. The system response is stable and the learning algorithm adjusts controller parameters very fast. The system response at the end of the tuning procedure (after 300 sec) is shown in Figure 5.9. The process output follows the reference model closely. The final value of K_{p} is approximately equal to 1.

Very often in robot control applications, when a robot moves from one point to another, moments of inertia of each robot joint are changing. Furthermore, a load carried by the robot could also change during operation execution. In order to investigate the learning algorithm performance in the case of load change, experiments with a load three times larger than the nominal one were conducted.

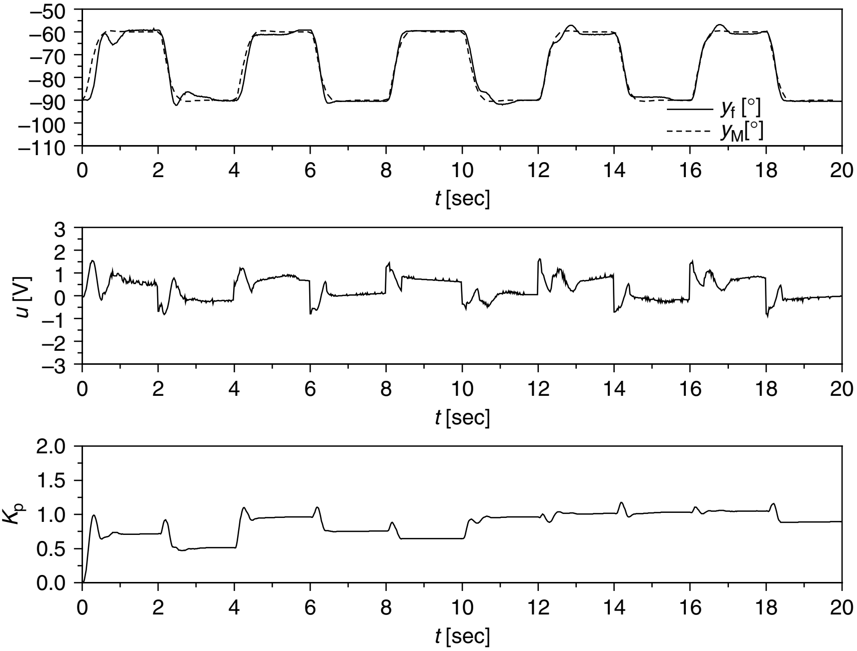


FIGURE 5.8 The system output (y_f), the reference model (y_M), the controller output (u), and the controller proportional gain responses (K_p) in the case of nominal load at the beginning of learning. (From Kovačić, Z., Cupec, R., and Bogdan, S., *IFAC System Structure and Control 2001*, Vol. 2. With permission from Elsevier.)

The structure of the self-learning fuzzy logic controller and the value of learning coefficient γ remain the same as in the case of nominal load. Initial values of parameter vectors $\mathbf{A}_y$ and $\mathbf{A}_{\Delta y}$ and proportional gain K_p are set to 0.

Figure 5.10 and Figure 5.11 show the system response at the beginning and at the end of learning in the case of a large load applied to the motor shaft. Even though at the beginning of learning the controller output has oscillations, the system is stable and the learning is fast. The parameter K_p approaches its steady-state value after only one change of system input. Finally, the tracking error $e_M(k)$ is very small and the system follows the reference model (Figure 5.11). The proportional gain K_p ends with an increased value of 1.75 (for the nominal load that value was 1).

Let us look at the results obtained for an abrupt load change from the three times nominal load to the no load condition. Initially, the parameter vectors $\mathbf{A}_y$ and $\mathbf{A}_{\Delta y}$ and gain K_p have the values obtained for a load three times larger than the nominal one.

From Figure 5.12 it may be seen that the process response has oscillations at the beginning of learning but the system remains stable. The controller output $u(k)$ changes its sign several times, that is, the signal form could be compared with a bang-bang control. The proportional gain K_p oscillates around its steady-state value.

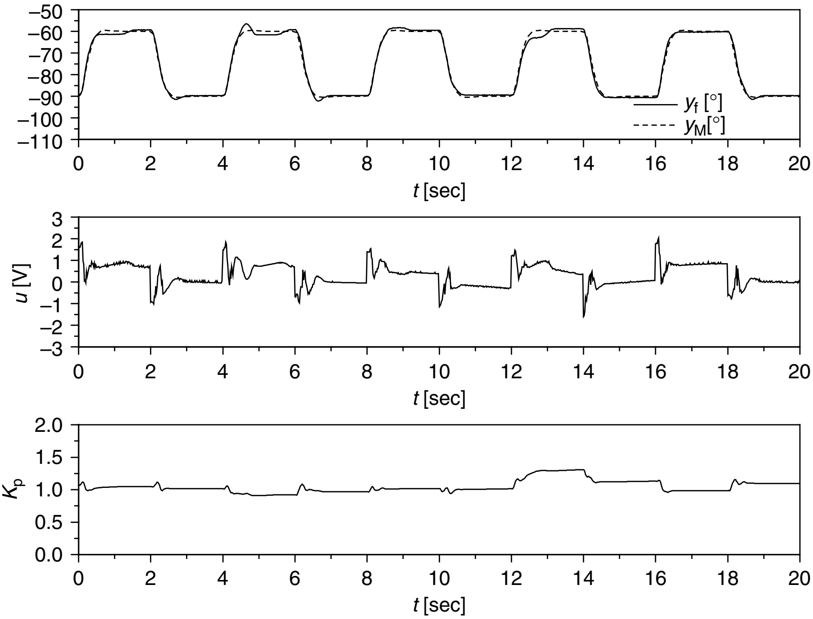


FIGURE 5.9 The system output (y_f), the reference model (y_M), the controller output (u), and the controller proportional gain responses (K_p) in the case of nominal load at the end of learning. (From Kovačić, Z., Cupec, R., and Bogdan, S., *IFAC System Structure and Control 2001*, Vol. 2. With permission from Elsevier.)

The system responses at the end of learning are shown in Figure 5.13. It may be seen that due to the learning effect, oscillations of the controller output are largely reduced. The process response closely follows the reference model. The proportional gain K_p ends with a decreased value of 0.75.

In order to see more precisely the effects of nonlinear functions $g_y(\cdot)$ and $g_{\Delta y}(\cdot)$, a large set point change was applied to the input of the system. Figure 5.14 and Figure 5.15 show the system responses in the case of stepwise set point changes from −90° to +90° with the motor shaft load three times larger than the nominal one. The initial values of parameter vectors $\mathbf{A}_y$ and $\mathbf{A}_{\Delta y}$ and proportional gain K_p were set to 0.

At the beginning of learning (Figure 5.14) the magnitude of tracking error $e_M(k)$ is close to 50% of system input's magnitude. As expected, the controller output has the largest value at the moment when the system set point is 0° since the gravitational torque reaches its maximum. In the set point +90° the controller output contains noise since the fuzzy mapping function responsible for compensation of nonlinear friction torque has not been estimated yet. At the end of learning (Figure 5.15) the tracking error is significantly reduced and the system follows the reference model dynamics closely.

The final forms of fuzzy mapping functions representing nonlinear functions $g_y(\cdot)$ and $g_{\Delta y}(\cdot)$ are shown in Figure 5.16 and Figure 5.17, respectively.

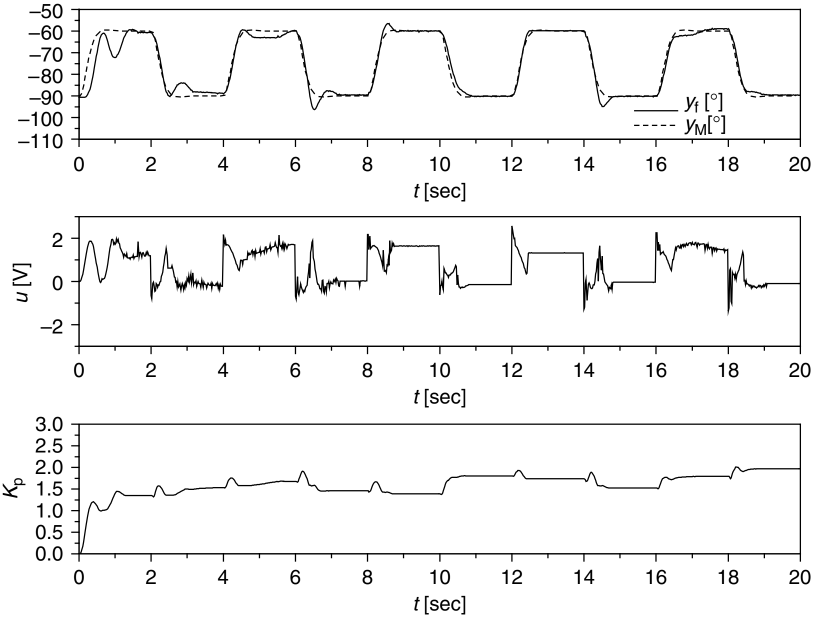


FIGURE 5.10 The system output (y_f), the reference model (y_M), the controller output (u), and the controller proportional gain responses (K_p) in the case of three times larger load than the nominal at the beginning of learning. (From Kovačić, Z., Cupec, R., and Bogdan, S., *IFAC System Structure and Control 2001*, Vol. 2. With permission from Elsevier.)

Fuzzy mapping functions have assumed forms, which were very much alike to inverse functions of friction and nonlinear gravity-dependent load (Figure 5.3 and Figure 5.4).

The example presented a model reference-based self-learning fuzzy logic control scheme suitable for control of nonlinear systems. The system stability was assessed by applying a direct Lyapunov method and stability conditions obtained for a selected Lyapunov function were used for determination of the allowed range of learning coefficient values. The experimental results obtained under various operating conditions confirmed that the analytically founded synthesis of the self-learning (adaptive) fuzzy logic controller headed to a stable adaptation process without steady state errors and with very close following of the reference model dynamics.

5.2 Self-Organizing Fuzzy Control Based on the Hurwitz Stability Criteria

In this chapter, we describe self-organization of a fuzzy rule table by using learning algorithm based on model tracking error $e_M(k)$ and so-called "velocity" and "acceleration" components $e_M(k-1)$ and $e_M(k-2)$. Second-order reference model (3.28) is used for defining the desired closed-loop system dynamics.

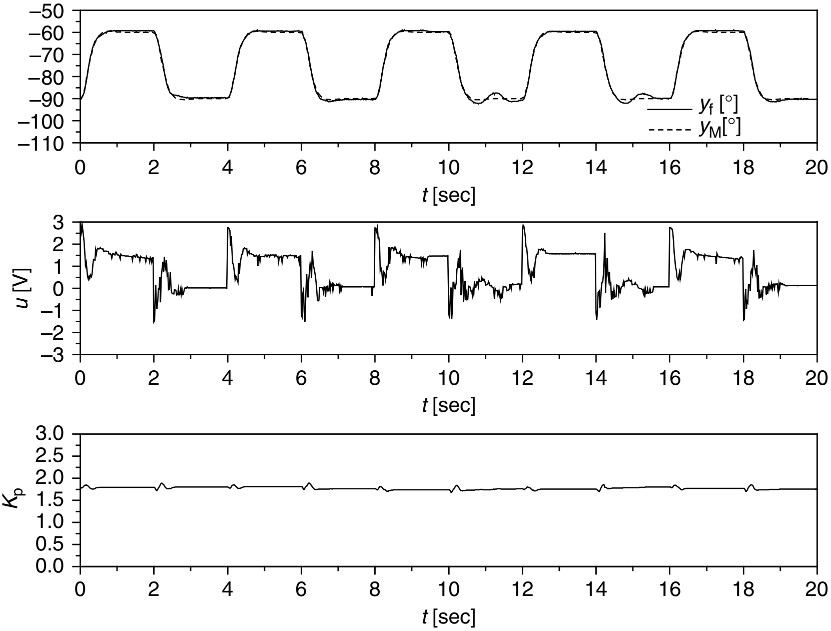


FIGURE 5.11 The system output (y_f), the reference model (y_M), the controller output (u), and the controller proportional gain responses (K_p) in the case of three times larger load than the nominal at the end of learning. (From Kovačić, Z., Cupec, R., and Bogdan, S., *IFAC System Structure and Control 2001*, Vol. 2. With permission from Elsevier.)

A self-organizing fuzzy controller will be designed for an inherently stable linearizable nonlinear SISO control object, which has time-varying parameters. Such a process is given by a set of equations:

$$\begin{aligned} \dot{x}_1 &= x_2 \\ &\vdots \\ \dot{x}_n &= f(\mathbf{x}, \boldsymbol{\lambda}, t) + b(t)u + d(t) \\ y_f &= x_1 \end{aligned} \tag{5.40}$$

where $\mathbf{x}$ is the state vector; f, the unknown nonlinear function; $b > 0$, the unknown process gain; u, the control input; d, the measurement noise; y_f, the output; and $\boldsymbol{\lambda}$, the parameter vector.

Providing that input and output variables are measurable, an approximate linear description of the control object (5.40) in a selected operating point can be obtained by using standard off-line identification methods. A very large number of linearizable systems can be satisfactorily described with a linear second-order approximation (3.29).

In Section 3.2 we have shown that ideal control applied to second-order process approximation (3.29) would synthesize control signal (3.41), which would provide

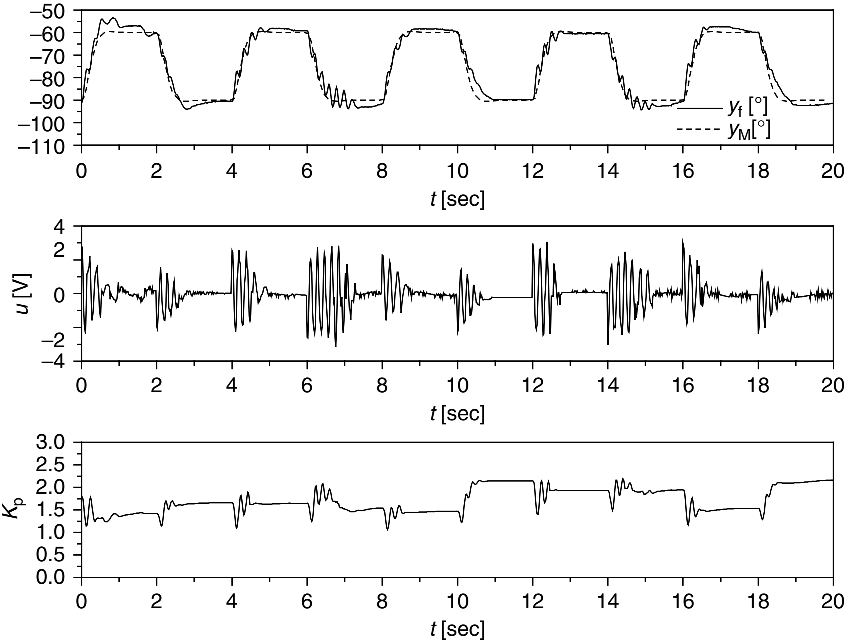


FIGURE 5.12 The system output (y_f), the reference model (y_M), the controller output (u), and the controller proportional gain responses (K_p) in the case of change from the three times nominal load to the no load condition at the beginning of learning.

the equality of process and reference model responses, $y_M(k) = y_A(k)$. Assuming that the reference input signal $u_r(k)$ has a constant value or that it is changing slowly (i.e., $u_r(k) \approx u_r(k-1)$), the controller function becomes a combination of linear controller function Ψ and feedforward part (3.46):

$$\begin{aligned} u(k) &= k_1 e(k) + k_2 \Delta e(k) + k_3 u_r(k) \\ &= \Psi[e(k), \Delta e(k)] + k_3 u_r(k) \end{aligned} \tag{5.41}$$

where k_1, k_2, and k_3 are defined by (3.43).

In the case of controlling nonlinear control objects (5.40), linear controller (5.41) would have to operate with different values of k_1, k_2, and k_3 in different operating points. Additional adjustments of controller parameters would be needed to compensate for continuous system parameter variations. Therefore, instead of using linear form (5.41), it would be more appropriate to use the following nonlinear expression:

$$u(k) = \Gamma[e(k), \Delta e(k) \mid k] + k_3(k) u_r(k) = u_{FC}(k) + u_F(k) \tag{5.42}$$

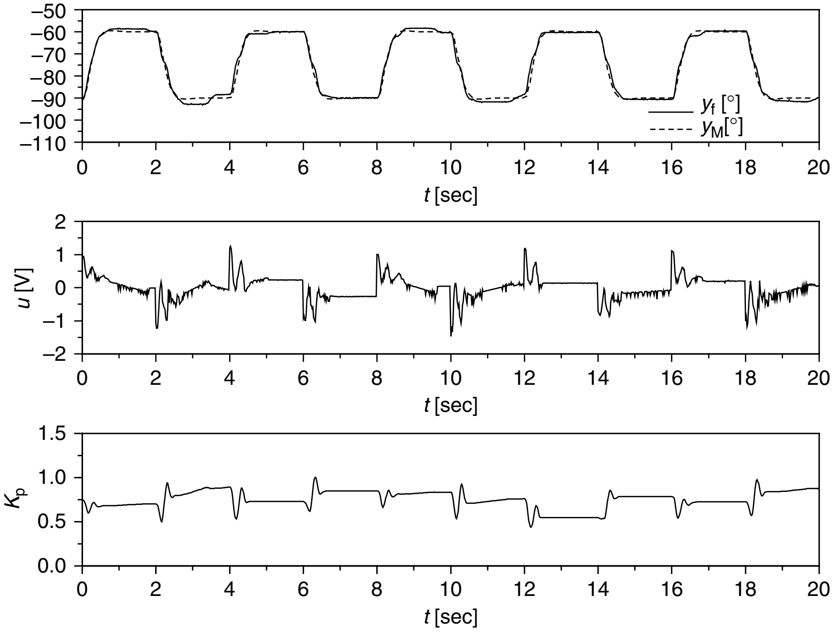


FIGURE 5.13 The system output (y_f), the reference model (y_M), the controller output (u), and the controller proportional gain responses (K_p) in the case of change from the three times nominal load to the no load condition at the end of learning.

where Γ is a nonlinear time-varying fuzzy control function. It must be noted that k_3 in (5.42) is also changing in time, which means that self-organization of a fuzzy controller should affect this parameter, too.

Model tracking error $e_M(k)$ describes the difference between the responses of reference model and closed-loop system:

$$\begin{aligned} e_M(k) &= y_M(k) - y_f(k) \\ &= y_M(k) - \{\Lambda[y_f(k-1), y_f(k-2), \ldots, y_f(k-n) \mid k] + b(k)u(k)\} \end{aligned} \tag{5.43}$$

where $y_f(k)$ is a discrete form of nonlinear system output (5.40).

By combining Equations (3.28), (5.42), and (5.43), $e_M(k)$ attains the form:

$$\begin{aligned} e_M(k) = {} & a_{M1}e_M(k-1) + a_{M2}e_M(k-2) + a_{M1}y_f(k-1) + a_{M2}y_f(k-2) \\ & - \Lambda[y_f(k-1), y_f(k-2), \ldots, y_f(k-n) \mid k] + [b_{M1} - b(k)k_3(k)]u_r(k) \\ & - b(k)\Gamma[e(k-1), \Delta e(k-1) \mid k-1] \end{aligned} \tag{5.44}$$

Considering only initial conditions, $u_r(k) = 0$, and assuming that Λ is changing slowly with respect to changes of controller parameters caused by self-organization

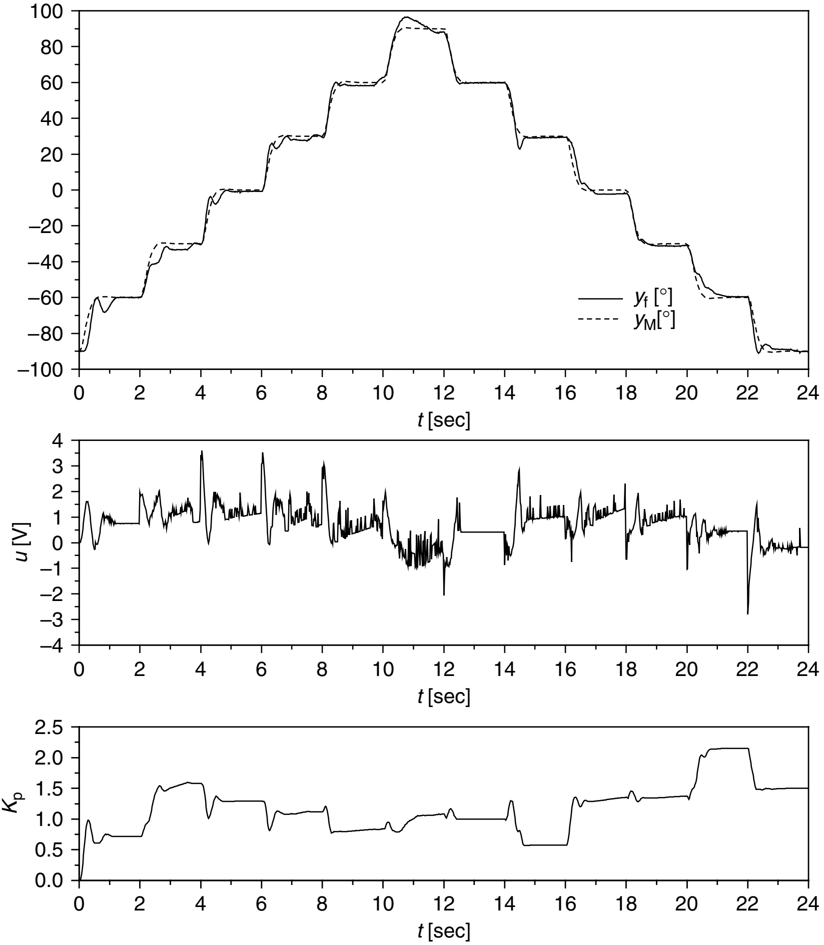


FIGURE 5.14 The system output (y_f), the reference model (y_M), the controller output (u), and the controller proportional gain responses (K_p) in the case of three times nominal load at the beginning of learning.

(thus being time independent), $e_\mathrm{M}(k)$ attains the following form:

$$\begin{aligned} e_\mathrm{M}(k) = a_{\mathrm{M}1}e_\mathrm{M}(k-1) + a_{\mathrm{M}2}e_\mathrm{M}(k-2) + F[y_f(k-1), y_f(k-2), \ldots, y_f(k-n)] \\ - b(k)\Gamma[e(k-1), \Delta e(k-1) \mid k-1] \end{aligned} \tag{5.45}$$

where

$$\begin{aligned} \mathrm{F} &= \mathrm{F}[y_f(k-1), y_f(k-2), \ldots, y_f(k-n)] \\ &= a_{\mathrm{M}1}y_f(k-1) + a_{\mathrm{M}2}y_f(k-2) - \Lambda[y_f(k-1), y_f(k-2), \ldots, y_f(k-n)] \end{aligned} \tag{5.46}$$

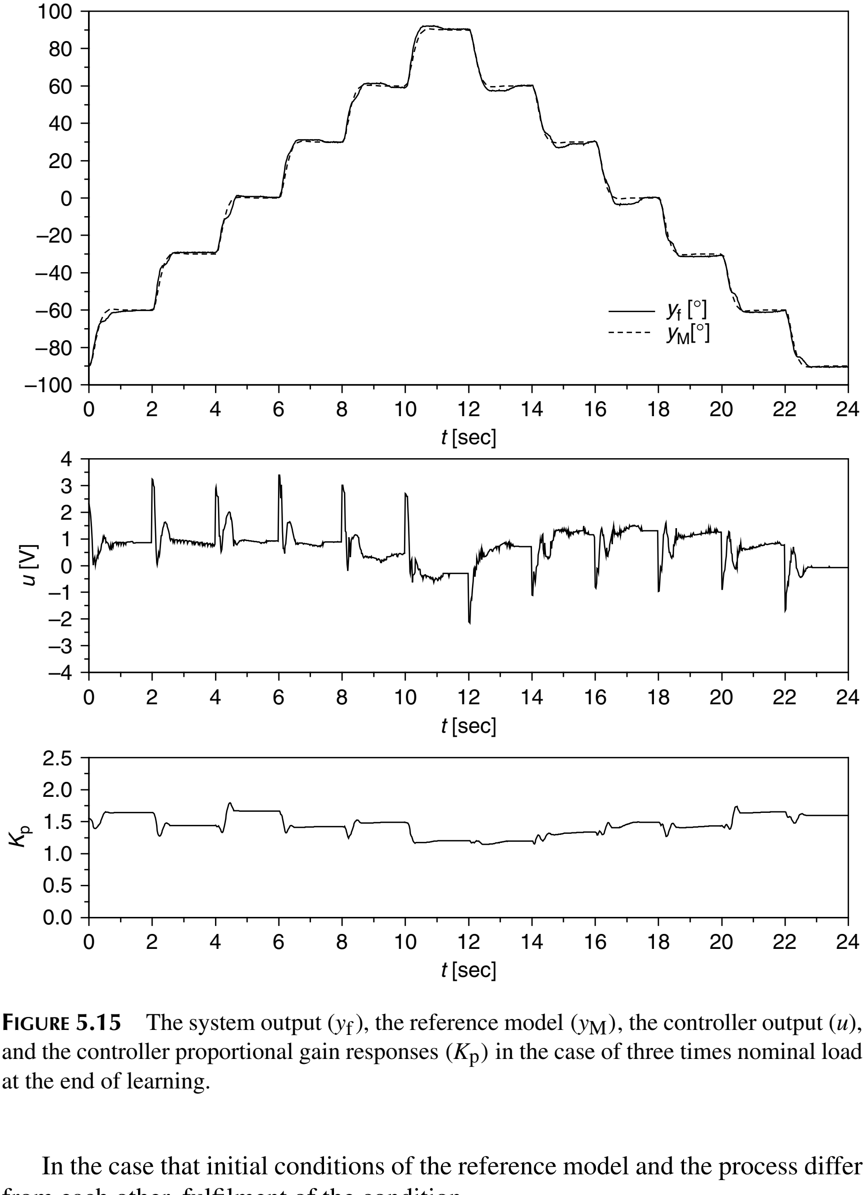


FIGURE 5.15 The system output (y_f), the reference model (y_M), the controller output (u), and the controller proportional gain responses (K_p) in the case of three times nominal load at the end of learning.

In the case that initial conditions of the reference model and the process differ from each other, fulfilment of the condition

$$\Gamma[e(k-1), \Delta e(k-1) \mid k-1] = \frac{1}{b(k)} F[y_f(k-1), y_f(k-2), \ldots, y_f(k-n)] \tag{5.47}$$

will enforce the model tracking error to diminish with dynamics of the reference model. In other words, the goal of self-organization is to modify the fuzzy control

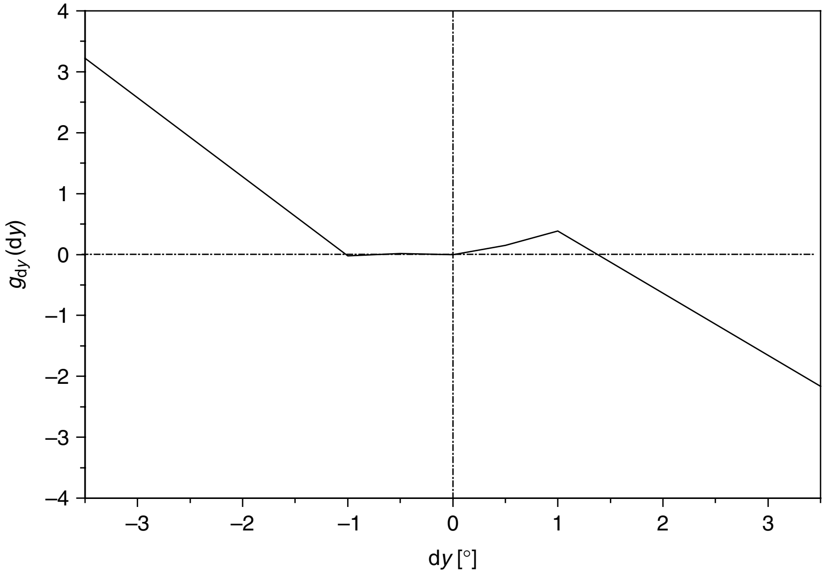


FIGURE 5.16 The shape of a nonlinear function $g_{\Delta y}(\cdot)$ obtained at the end of learning. (From Kovačić, Z., Cupec, R., and Bogdan, S., *IFAC System Structure and Control 2001*, Vol. 2. With permission from Elsevier.)

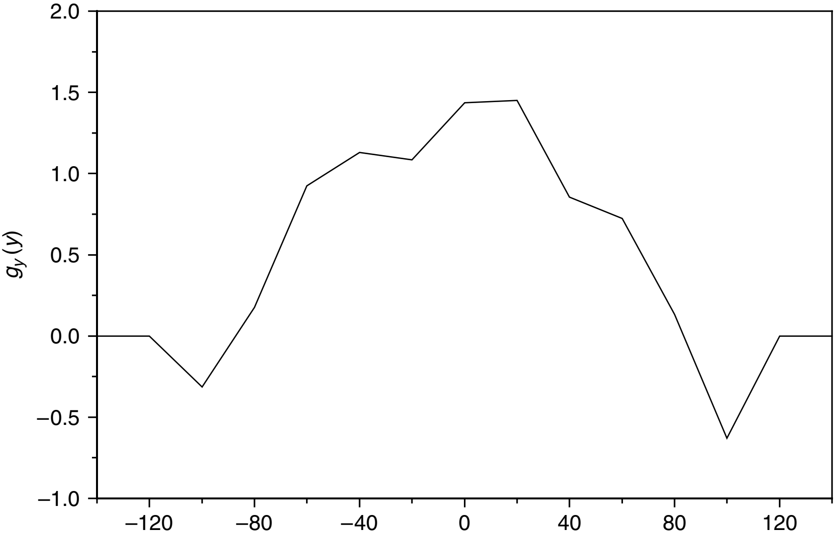


FIGURE 5.17 The shape of a nonlinear function $g_y(\cdot)$ obtained at the end of learning. (From Kovačić, Z., Cupec, R., and Bogdan, S., *IFAC System Structure and Control 2001*, Vol. 2. With permission from Elsevier.)

surface Γ in a way, which will enforce the last term in (5.45) to asymptotically approach nonlinear function F, thus providing a stable approach of Γ to its steady-state form, let us designate it Γ^*.

We assume that all output fuzzy sets are represented by singletons. Because of simplicity and good interpolation features, a fuzzy controller output is computed according to the center of gravity principle (2.22). Referring to (2.23), a particular singleton A_{ij} will contribute to a crisp controller output value $u_{\mathrm{FC}}(k)$ depending on the degree of contribution described with the fuzzy basis function φ_{ij}:

$$u_{\mathrm{FC}}(k) = \Gamma[e(k), \Delta e(k) \mid k] = \sum_{i,j} A_{ij}(k)\varphi_{ij}[e(k), \Delta e(k)] \tag{5.48}$$

where $A_{ij}(k)$ is the output singleton activated by (i, j)th fuzzy rule.

Assuming that input membership functions are time-invariant, the intention of the learning algorithm is to compensate all variations of process parameters by modifying the fuzzy control surface by varying singletons $A_{ij}(k)$. At this point, time-dependent variations of system parameters and their impact on feedforward coefficient $k_3(k)$ will be neglected.

After determination of all universes of discourse and after selection of the number and size of fuzzy input sets (including the shape of membership functions), the self-organization of the fuzzy rule-table is accomplished by using a learning algorithm, which utilizes the third-degree model tracking error polynomial as a measure of control quality and modifies singleton values according to the degree of their contribution to a crisp controller output:

$$\begin{aligned}\Delta A_{ij}(k) &= [\gamma_1 e_{\mathrm{M}}(k) + \gamma_2 e_{\mathrm{M}}(k-1) + \gamma_3 e_{\mathrm{M}}(k-2)]\varphi_{ij}[e(k), \Delta e(k)] \\ &= \Delta A(k)\varphi_{ij}[e(k), \Delta e(k)] \end{aligned} \tag{5.49}$$

where $\gamma_1, \gamma_2, \gamma_3$ are the learning coefficients and $\Delta A(k)$ is the learning mechanism output.

The considered self-organizing fuzzy control scheme contains a feedforward element, a fuzzy controller, a reference model, and a model reference-based learning mechanism (Figure 5.18). In order to establish steady-state accuracy and to cancel disturbance effects, an integral element is added. The integral element is activated only when both fuzzy controller input values are close to the phase plane origin (i.e., when they belong to their zero subsets).

It is of practical interest to find values of learning coefficients $\gamma_1, \gamma_2, \gamma_3$, which would ensure convergence of the learning process and provide a stable performance of the control system. Referring to (5.44), let us introduce a steady-state form of fuzzy controller (5.48), which corresponds to the linear control function Ψ in (5.41):

$$\Gamma^*[e(k), \Delta e(k) \mid k] = \sum_{i,j} A^*_{ij}\varphi_{ij}[e(k), \Delta e(k)] \tag{5.50}$$

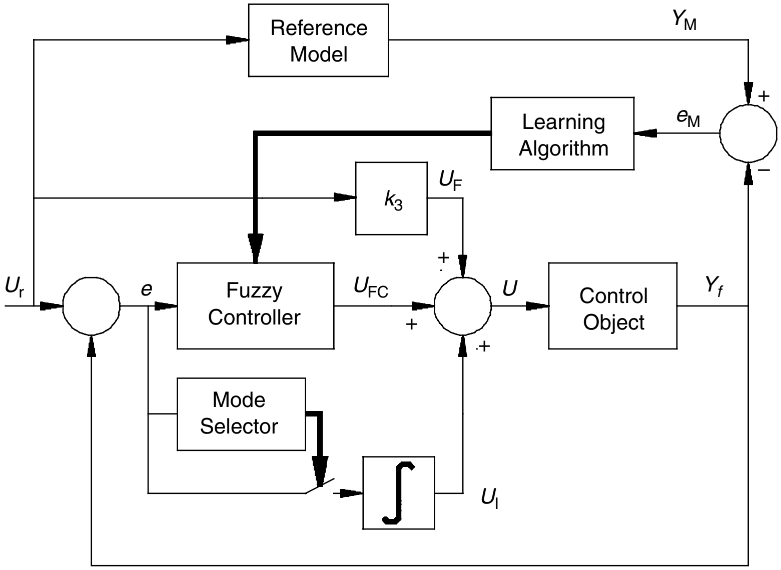


FIGURE 5.18 The structure of self-organizing fuzzy control scheme. (From Kovačić, Z., Bogdan, S., and Crnosija, P., *10th IEEE Intl. Symp. Intell. Contr.*, 389–394, 1995. With permission from Elsevier.)

where A_{ij}^* is a steady-state value of the output singleton activated by (i,j)th fuzzy rule.

Having in mind that in the end of learning $b(k)\Gamma^*$ should emulate F (please refer to [5.47]), the tracking error $e_M(k)$ is determined by insertion of (5.48) and (5.50) into (5.45) as follows:

$$\begin{aligned} e_M(k) &= a_{M1}e_M(k-1) + a_{M2}e_M(k-2) \\ &\quad + \sum_{i,j}[A_{ij}^* - b(k)A_{ij}(k-1)]\varphi_{ij}[e(k-1), \Delta e(k-1)] \end{aligned} \tag{5.51}$$

Assuming that condition $\varphi_{ij}[e(k-1), \Delta e(k-1)] \approx \varphi_{ij}[e(k-2), \Delta e(k-2)]$ always holds for sufficiently short control intervals, the change in error has a form:

$$\begin{aligned} \Delta e_M(k) &= e_M(k) - e_M(k-1) = a_{M1}e_M(k-1) + (a_{M2} - a_{M1})e_M(k-2) \\ &\quad - a_{M2}e_M(k-3) - \sum_{i,j} b(k)[A_{ij}(k-1) - A_{ij}(k-2)] \\ &\quad \times \varphi_{ij}[e(k-1), \Delta e(k-1)] \end{aligned} \tag{5.52}$$

Difference $A_{ij}(k-1) - A_{ij}(k-2)$ is actually the change of a singleton, which must be accomplished by using the learning algorithm (5.49). After insertion of

(5.49) into (5.52), a recursive equation for the model tracking error is obtained:

$$\begin{aligned} e_{\mathrm{M}}(k) = &\left\{1 + a_{\mathrm{M}1} - b(k)\gamma_1 \sum_{i,j} \{\varphi_{ij}[e(k-1), \Delta e(k-1)]\}^2\right\} e_{\mathrm{M}}(k-1) \\ &+ \left\{a_{\mathrm{M}2} - a_{\mathrm{M}1} - b(k)\gamma_2 \sum_{i,j} \{\varphi_{ij}[e(k-1), \Delta e(k-1)]\}^2\right\} e_{\mathrm{M}}(k-2) \\ &- \left\{a_{\mathrm{M}2} + b(k)\gamma_3 \sum_{i,j} \{\varphi_{ij}[e(k-1), \Delta e(k-1)]\}^2\right\} e_{\mathrm{M}}(k-3) \end{aligned} \tag{5.53}$$

The characteristic equation in the z-domain is obtained from (5.53):

$$R(z) = r_3 z^3 + r_2 z^2 + r_1 z + r_0 = 0 \tag{5.54}$$

where $r_3 = 1$, and r_2, r_1, and r_0 are time-varying coefficients of the following form:

$$\begin{aligned} r_0 &= a_{\mathrm{M}2} + b(k)\gamma_3 \sum_{i,j} \{\varphi_{ij}[e(k-1), \Delta e(k-1)]\}^2 \\ r_1 &= a_{\mathrm{M}1} - a_{\mathrm{M}2} + b(k)\gamma_2 \sum_{i,j} \{\varphi_{ij}[e(k-1), \Delta e(k-1)]\}^2 \\ r_2 &= -1 - a_{\mathrm{M}1} + b(k)\gamma_1 \sum_{i,j} \{\varphi_{ij}[e(k-1), \Delta e(k-1)]\}^2 \end{aligned} \tag{5.55}$$

The conditions for the stability of a closed-loop fuzzy control system with characteristic equation (5.54) can be found by applying the Hurwitz stability criterion. According to the Hurwitz criterion of stability, four conditions must be fulfilled in order to ensure the absolute stability of e_{M}:

1. $R(1) > 0 \Rightarrow \gamma_1 + \gamma_2 + \gamma_3 > 0$
2. $R(-1) < 0 \Rightarrow \gamma_1 - \gamma_2 + \gamma_3 < \dfrac{2(1 + a_{\mathrm{M}1} - a_{\mathrm{M}2})}{b(k)\sum_{i,j}\{\varphi_{ij}[e(k-1), \Delta e(k-1)]\}^2}$
3. $|r_0| < r_3 \Rightarrow \left| b(k)\gamma_3 \sum_{i,j} \{\varphi_{ij}[e(k-1), \Delta e(k-1)]\}^2 + a_{\mathrm{M}2} \right| < 1$
4. $r_0^2 - r_3^2 < r_0 r_2 - r_1 r_3$

(5.56)

These conditions can be further used for determination of learning coefficient values γ_1, γ_2, and γ_3, but they do not give a unique solution. Different values of γ_1, γ_2, and γ_3 can still provide a stable learning process. Therefore, the designer must choose γ_1, γ_2, and γ_3 while paying attention to a desired tracking error response and the speed of learning. As found in (5.56), characteristics of a learning process depend on values of process coefficient $b(k)$ and the sum of squared fuzzy basis functions.

Most often, gain coefficient $b(k)$ is replaced by an off-line identified approximate value b_0. In order to ensure a stable learning process, it is very important to estimate a possible range of parameter variations that have an impact on $b(k)$. From the practical point of view, the worst case will happen for the maximal value of $b(k)$, which will give the minimum values of learning coefficients. The maximum value of the sum of squared fuzzy basis functions, which depends on the shape of membership functions and the number of input and output variables, can also be estimated.

Incorrect determination of feedforward coefficient k_3 will cause a static error unless this coefficient is also adjusted by the self-organization algorithm. The learning law can resemble in form to the learning law (5.49):

$$\Delta k_3(k) = [\gamma_1 e_{\mathrm{M}}(k) + \gamma_2 e_{\mathrm{M}}(k-1) + \gamma_3 e_{\mathrm{M}}(k-2)]u_{\mathrm{r}}(k) = \Delta A(k) u_{\mathrm{r}}(k) \tag{5.57}$$

Under assumption that the reference input is changing much slower than fuzzy controller parameters ($u_{\mathrm{r}}(k) \approx u_{\mathrm{r}}(k-1)$), the tracking error dynamics assumes the form:

$$\begin{aligned} e_{\mathrm{M}}(k) =& \left\{1 + a_{\mathrm{M1}} - b(k)\gamma_1 \sum_{i,j} \{\varphi_{ij}[e(k-1), \Delta e(k-1)]\}^2 + u_{\mathrm{r}}^2(k)\right\} e_{\mathrm{M}}(k-1) \\ &+ \left\{a_{\mathrm{M2}} - a_{\mathrm{M1}} - b(k)\gamma_2 \sum_{i,j} \{\varphi_{ij}[e(k-1), \Delta e(k-1)]\}^2 + u_{\mathrm{r}}^2(k)\right\} e_{\mathrm{M}}(k-2) \\ &- \left\{a_{\mathrm{M2}} + b(k)\gamma_3 \sum_{i,j} \{\varphi_{ij}[e(k-1), \Delta e(k-1)]\}^2 + u_{\mathrm{r}}^2(k)\right\} e_{\mathrm{M}}(k-3) \end{aligned} \tag{5.58}$$

The form of (5.58) is similar to the form of (5.53). The difference between expressions is in appearance of the new term $u_{\mathrm{r}}^2(k)$. Consequently, stability criteria (5.56) used for determination of learning coefficients assumes the following form:

1. $R(1) > 0 \;\Rightarrow\; \gamma_1 + \gamma_2 + \gamma_3 > 0$
2. $R(-1) < 0 \;\Rightarrow\; \gamma_1 - \gamma_2 + \gamma_3 < \dfrac{2(1 + a_{\mathrm{M1}} - a_{\mathrm{M2}})}{b(k)\left[\sum_{i,j} \{\varphi_{ij}[e(k-1), \Delta e(k-1)]\}^2 + u_{\mathrm{r}}^2(k)\right]}$
3. $|r_0| < r_3 \;\Rightarrow\; \left| b(k)\gamma_3 \left[\sum_{i,j} \{\varphi_{ij}[e(k-1), \Delta e(k-1)]\}^2 + u_{\mathrm{r}}^2(k)\right] + a_{\mathrm{M2}} \right| < 1$
4. $r_0^2 - r_3^2 < r_0 r_2 - r_1 r_3$

(5.59)

The important result is that learning stability will also depend on the maximal change of the reference input, which is usually known to the designer.

Special caution has to be taken to avoid the overlearning effect. It must be noted that self-organization should end when the predefined learning indicator (threshold) is reached (e.g., an indicator related to the average square model tracking error). The introduction of learning indicator elegantly solves the overlearning problem, declining any need for optimization of appropriate cost functions, which would otherwise inevitably lead to the problem of stagnation in local optima.

Example 5.2 Hurwitz stability criteria-based self-organizing fuzzy control.

In this example, we shall describe a design and a practical implementation of a Hurwitz criterion-based fuzzy controller scheme in the angular speed control loop of a thyristor converter-fed DCM drive [17]. Please note that the same control object was used in Section 4.2.5 to demonstrate the design and practical implementation of a fuzzy MRAC scheme.

The identification of DCM drive parameters in a selected operating point has resulted in the following parameter values: $\omega = 115$ rad/sec: $K_{\mathrm{TC}} = 108$ V/V, $T_{\mathrm{TC}} = 5$ msec, $K_{\mathrm{a}} = 0.05$ A/V, $T_{\mathrm{a}} = 15$ msec, $K = 0.755$ Vsec, $J_{\mathrm{T}} = 0.157\ \mathrm{kg\,m}^2$, $K_l = 0.0098$ Nmsec, $K_\omega = 0.065$ Vsec, $T_\omega = 25$ msec, $T_{\mathrm{d}} = 10$ msec. A linearized block scheme of the control loop is shown in Figure 5.19.

Seven linguistic subsets were defined for both fuzzy controller inputs (universes of discourse E and DE): NL, NM, NS, Z, PS, PM, PL. Based on the knowledge about angular speed control loop characteristics, maximum values for both inputs and the output of the fuzzy controller were estimated. Since only two adjacent input membership functions are overlapping, maximally four out of forty nine possible IF–THEN fuzzy control rules can contribute to the crisp controller output. If we arrange that adjacent membership functions intersect at the halves of the maximum values ($\mu_i = \mu_{i+1} = 0.5$), then the computation of fuzzy rules' contribution factors is even simpler (Figure 5.20).

The desired reference model performance indices are overshoot in the response, $\sigma_{\mathrm{m}} = 5\%$ and peak time, $t_{\mathrm{m}} = 0.2$ sec. Having a control interval $T_{\mathrm{d}} = 10$ msec,

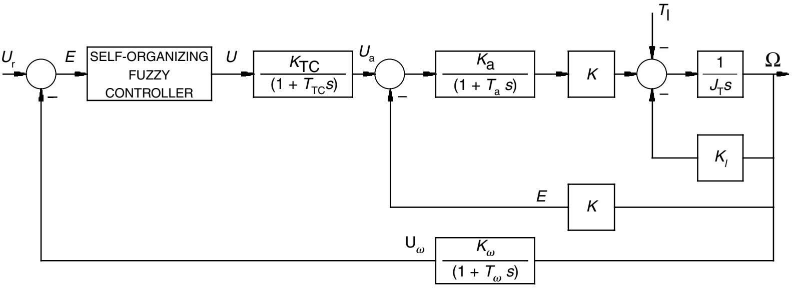

U_r
E
SELF-ORGANIZING FUZZY CONTROLLER
U
$\frac{K_{TC}}{(1 + T_{TC}s)}$
U_a
$\frac{K_a}{(1 + T_a s)}$
K
T_l
$\frac{1}{J_T s}$
Ω
K_l
E
K
U_ω
$\frac{K_\omega}{(1 + T_\omega s)}$
–
–
–
–
–

FIGURE 5.19 The structure of the DCM servo drive angular speed control loop. (From Kovačić, Z., Bogdan, S., and Crnosija, P., *10th IEEE Intl. Symp. Intell. Contr.*, 389–394, 1995. With permission from Elsevier.)

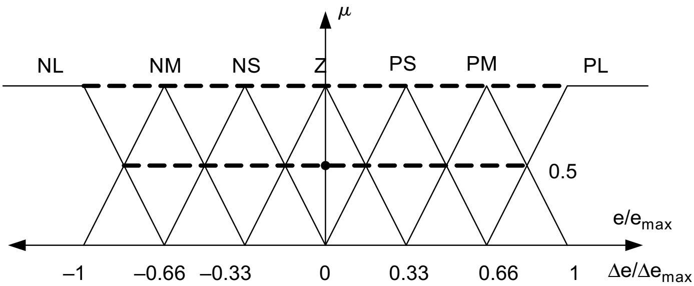


FIGURE 5.20 The distribution of fuzzy controller's input membership functions. (From Kovačić, Z., Bogdan, S., and Crnosija, P., *10th IEEE Intl. Symp. Intell. Contr.*, 389–394, 1995. With permission from Elsevier.)

the second-order reference model is described with the recursive equation (4.53). The integral gain coefficient (see Figure 5.18) has been set to $K_I = 0.002$.

An off-line identification of the studied DCM servo drive in the case of a stepwise change of the controller output has resulted in the second-order process approximation:

$$y_A(k) = 1.7225y_A(k-1) - 0.7293y_A(k-2) + 0.0416u(k) \tag{5.60}$$

Referring to Equations (3.43), (4.53), and (5.60), coefficients k_1, k_2, and k_3 attain the following values: $k_1 = 0.6775$, $k_2 = -0.3197$, $k_3 = 0.1629$. The impact of operating point-dependent and time-dependent variations of system parameters on feedforward coefficient k_3 has been ignored.

As mentioned in Section 3.2, if k_1 and k_2 can be estimated fairly well, then the self-organization can start from an initially preset fuzzy rule-table. If for some practical reason such identification is not accomplishable, learning can start from a blank fuzzy rule-table, provided that at least a control object gain coefficient b_0 is known.

Special caution has to be taken to avoid the overlearning effect. For this purpose, the following learning indicator related to the average square tracking error has been introduced:

$$I_e(k) = \frac{\sum_{i=0}^{N-1} e_M^2(k-N+i)}{N} \tag{5.61}$$

For a selected number of samples N, learning is supposed to stop after the value of learning indicator I_e drops below a predetermined threshold value. At that moment the learning mechanism is deactivated, remaining idle until triggered again. In the studied case, for $N = 300$, the threshold value of I_e has been set to 0.0017.

First, let us test the Hurwitz criterion-based fuzzy control method by a series of computer simulations for the case of learning with initially preset singleton values calculated according to the model reference-based presetting algorithm (3.47).

TABLE 5.1
Preset (Shaded) and Modified by Learning (Unshaded) Fuzzy Rule-Table (Simulation)

	NLE	NME	NSE	ZE	PSE	PME	PLE
NLDE	0	0.05	0.01	0.15	0.2	0.25	0.3
	0	0.05	0.01	0.15	0.2	0.24	0.3
NMDE	−0.05	0	0.05	0.1	0.15	0.2	0.25
	−0.05	0	0.05	0.07	0.15	0.26	0.27
NSDE	−0.1	−0.04	0	0.06	0.11	0.16	0.21
	−0.09	−0.04	0	0	0.11	0.18	0.25
ZDE	−0.15	−0.1	−0.05	0	0.05	0.1	0.15
	−0.2	−0.11	−0.05	0	0.05	0.11	0.2
PSDE	−0.21	−0.16	−0.11	−0.06	0	0.04	0.1
	−0.26	−0.2	−0.14	−0.05	0	0.04	0.1
PMDE	−0.25	−0.2	−0.15	−0.1	−0.05	0	0.05
	−0.28	−0.26	−0.19	−0.1	−0.05	0	0.05
PLDE	−0.3	−0.25	−0.2	−0.15	−0.1	−0.05	0
	−0.3	−0.25	−0.2	−0.15	−0.1	−0.05	0

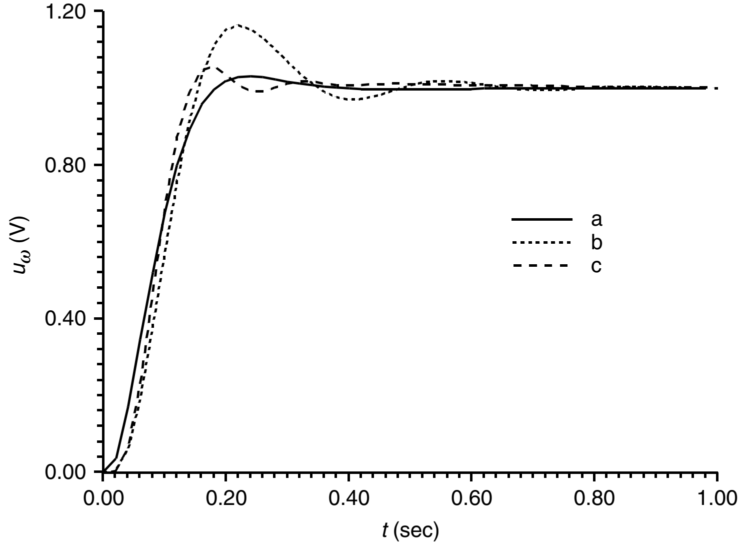


FIGURE 5.21 The reference model response (a) and the measured angular speed responses (preset fuzzy rule-table): the 1st run (b), the 15th run (c). (From Kovačić, Z., Bogdan, S., and Crnosija, P., *10th IEEE Intl. Symp. Intell. Contr.*, 389–394, 1995. With permission from Elsevier.)

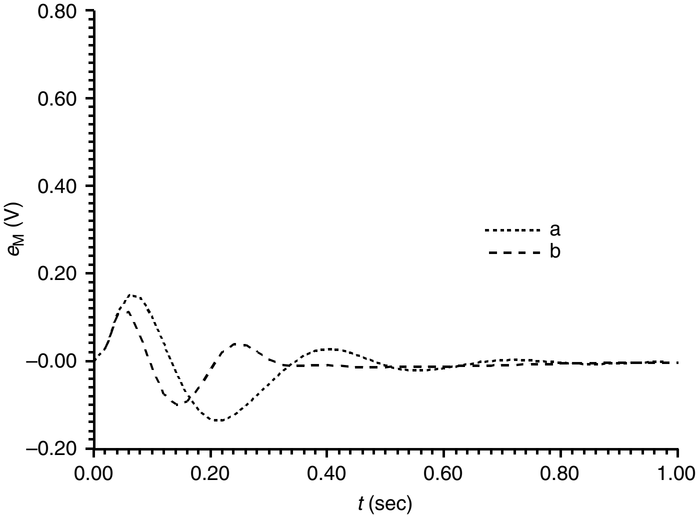


FIGURE 5.22 The tracking error responses (preset fuzzy rule-table): the 1st run (a), the 15th run (b).

These values are displayed shaded in Table 5.1. Learning coefficients in learning algorithms (5.49) and (5.57) have the following values: $\gamma_1 = 0.2$, $\gamma_2 = -0.18$, and $\gamma_3 = 0.018$. Figure 5.21 shows reference model and measured angular speed responses obtained after the first and after fifteen runs of the system (i.e., 30 runs in both directions). In the first run, the dynamic behavior is determined primarily by the preset controller dynamics and feedforward control signal u_F. The system response differs from the reference model response ($\sigma_m = 15\%$), but the difference is tolerable. After 15 runs, the system follows the reference model closely, and the maximum tracking error value decreases from initial 15 to 11% of imposed change of the reference input $\Delta u_r = 1$ V (Figure 5.22). Output of the learning mechanism ΔA, as expected, follows the shape of the tracking error waveform (Figure 5.23). It may be noticed that peak values of ΔA are in both characteristic runs almost the same, that is, $\Delta A \approx 0.01$. Figure 5.24 shows a very acceptable nonoscillatory form of fuzzy controller output responses in both characteristic runs, what is essential for practical control applications.

After 15 runs, the learning process has eventually resulted in the modified fuzzy rule-table with singleton values displayed unshaded in Table 5.1. Since the simulated trajectories have not reached the corners of fuzzy rule-table, those singleton values have remained almost unchanged, while the other values have changed up to 50% of the preset values.

The aim of the next group of simulation experiments was to test the effectiveness of learning starting from a blank fuzzy rule-table. A prerequisite for such experimenting was the ability to identify the gain coefficient b_0, which in turn enabled the calculation of feedforward coefficient k_3. Figure 5.25 shows the reference model response and the measured angular speed responses obtained after

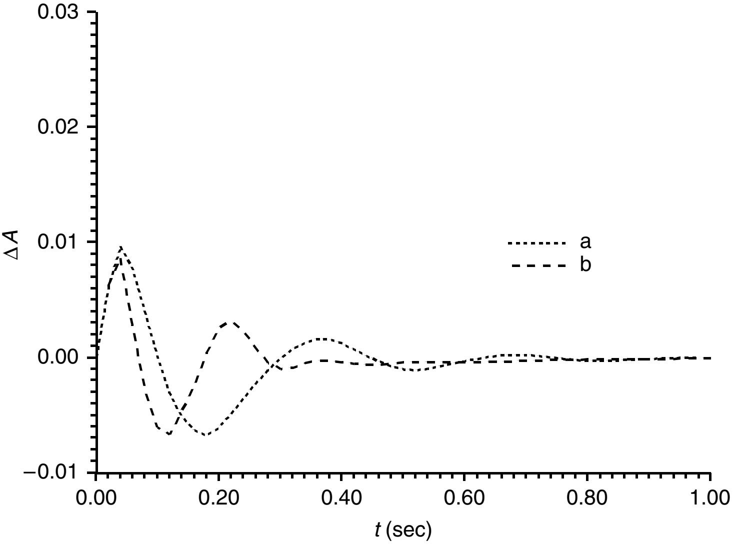


FIGURE 5.23 The learning mechanism output responses (preset fuzzy rule-table): the 1st run (a), the 15th run (b).

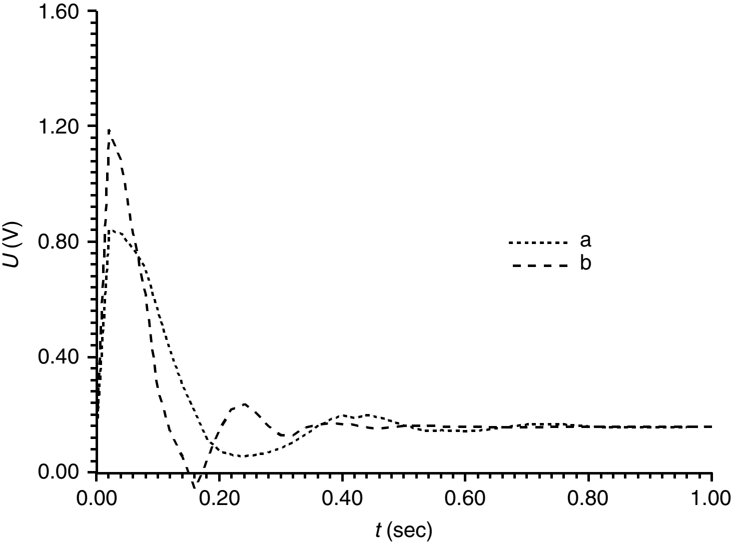


FIGURE 5.24 The self-organizing fuzzy controller output responses (preset fuzzy rule-table): the 1st run (a), the 15th run (b). (From Kovačić, Z., Bogdan, S., and Crnosija, P., *10th IEEE Intl. Symp. Intell. Contr.*, 389–394, 1995. With permission from Elsevier.)

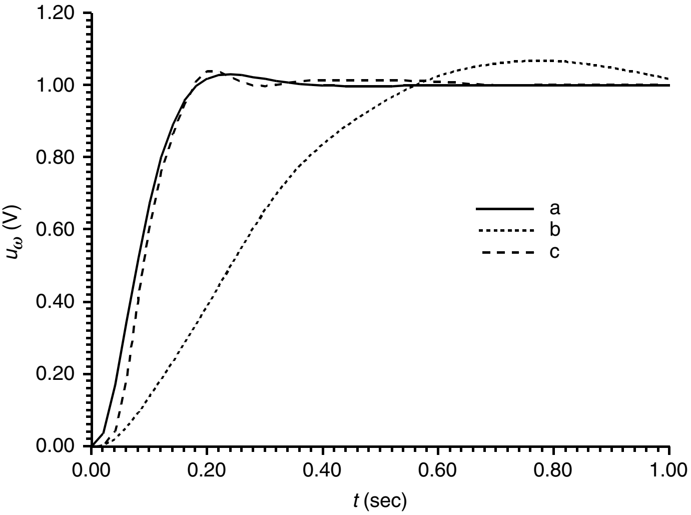


FIGURE 5.25 The reference model response (a) and the measured angular speed responses (blank fuzzy rule-table): the 1st run (b), the 30th run (c). (From Kovačić, Z., Bogdan, S., and Crnosija, P., *10th IEEE Intl. Symp. Intell. Contr.*, 389–394, 1995. With permission from Elsevier.)

the first and after thirty runs of the system. In the first run, the dynamic behavior is primarily determined by feedforward control signal u_F. As may be seen, after thirty runs the maximum tracking error value stepped down from initial 66 to 15% of imposed change of the reference input $\Delta u_r = 1$ V (Figure 5.26). The output of learning mechanism ΔA follows the shape of tracking error (Figure 5.27). The peak value after 30 runs is almost three times smaller than at the beginning of learning process and it is comparable in magnitude with the preset controller output values (Figure 5.23). Figure 5.28 shows the fuzzy controller output responses in two characteristic runs indicating again a very acceptable form of the control signal for practical applications.

The centroid values obtained after 15 runs of the learning process are displayed in Table 5.2. Similarity with the values obtained in the previous case is obvious. Exceptions are the values in the fuzzy rule-table corners, which have not been reached by the simulated trajectories. This has caused a nonmonotonous increase or decrease of values in the fuzzy rule-table.

The selection of other learning coefficient values, which satisfy stability conditions (5.53) has an influence on the tracking error dynamics and the speed of learning. Figure 5.29 shows the transitions of learning indicator I_e (please refer to [5.61]) obtained for different sets of learning coefficient values. In general, the higher the values of learning coefficients, the faster will the indicator I_e drop below a predetermined triggering value. On the other hand, since one must usually deal with unknown and time varying control object parameters, it is convenient to choose smaller values of learning coefficients in order to provide reliable learning under all operating circumstances. Figure 5.29 also shows the transition of I_e in the

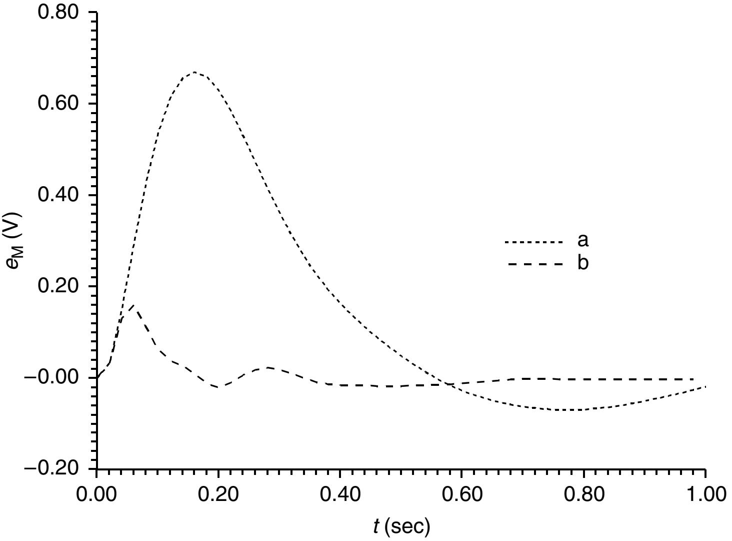


FIGURE 5.26 The tracking error responses (blank fuzzy rule-table): the 1st run (a), the 30th run (b).

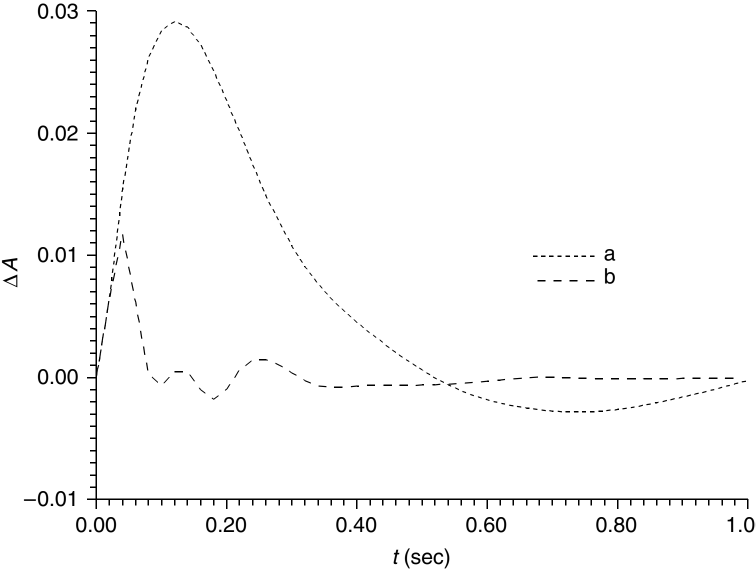


FIGURE 5.27 The learning mechanism output responses (blank fuzzy rule-table): the 1st run (a), the 30th run (b).

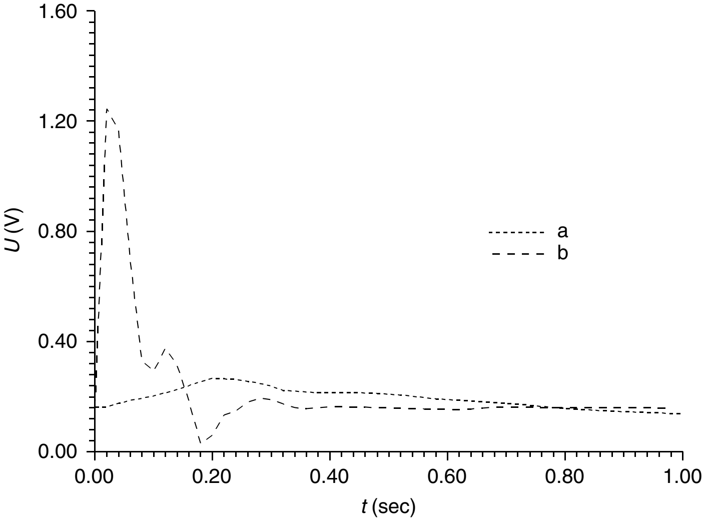


FIGURE 5.28 The self-organizing fuzzy controller output responses (blank fuzzy rule-table): the 1st run (a), the 15th run (b). (From Kovačić, Z., Bogdan, S., and Crnosija, P., *10th IEEE Intl. Symp. Intell. Contr.*, 389–394, 1995. With permission from Elsevier.)

TABLE 5.2
The Fuzzy Rule-Table Obtained after Learning (Simulation)

	NLE	NME	NSE	ZE	PSE	PME	PLE
NLDE	0	0	0	0	0	0	0
NMDE	0	0	0	0	0.02	0.08	0.03
NSDE	0	0	0	0	0.14	0.28	0.09
ZDE	−0.22	−0.18	−0.11	0	0.1	0.16	0.22
PSDE	−0.1	−0.29	−0.14	−0.02	0	0	0
PMDE	−0.02	−0.06	−0.02	0	0	0	0
PLDE	0	0	0	0	0	0	0

case of using the preset fuzzy rule-table. The descent to the threshold level is very fast, the transition starts from markedly lower initial values and, therefore, a much better performance of the control system is provided in the initial phase of learning.

Figure 5.30 and Figure 5.31 illustrate the robustness of the self-organizing fuzzy controller in the case of large parameter variations. After completion of learning for nominal parameter values, the stepwise changes of the moment of inertia in the range from $3J_T$ (Figure 5.30) to $J_T/3$ (Figure 5.31) were simulated. These changes were large enough to provoke a new start of self-organization, which was fully completed in both extreme cases after only several iterations, thus showing a fairly high robustness to large parameter variations.

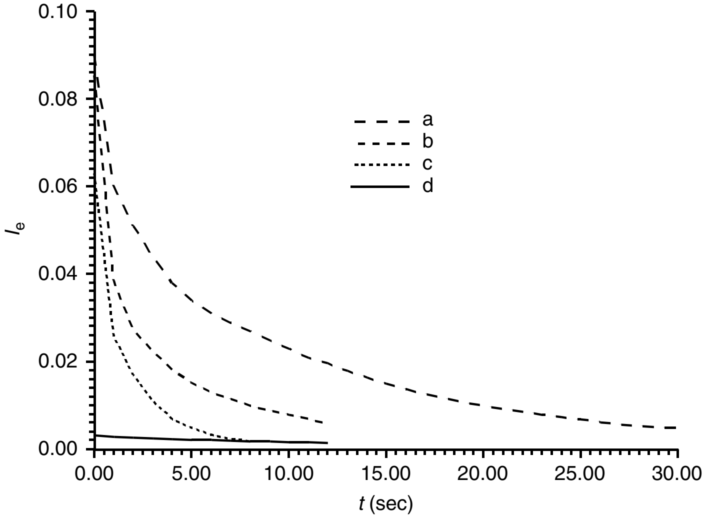


FIGURE 5.29 The transitions of learning indicator I_e: $\gamma_1 = 0.1$, $\gamma_2 = -0.085$, $\gamma_3 = 0.007$ (a), $\gamma_1 = 0.2$, $\gamma_2 = -0.18$, $\gamma_3 = 0.018$ (b), $\gamma_1 = 0.8$, $\gamma_2 = -0.72$, $\gamma_3 = 0.072$ (c), $\gamma_1 = 0.2$, $\gamma_2 = -0.18$, $\gamma_3 = 0.018$ with a preset fuzzy rule-table (d). (From Kovačić, Z., Bogdan, S., and Crnosija, P., *10th IEEE Intl. Symp. Intell. Contr.*, 389–394, 1995. With permission from Elsevier.)

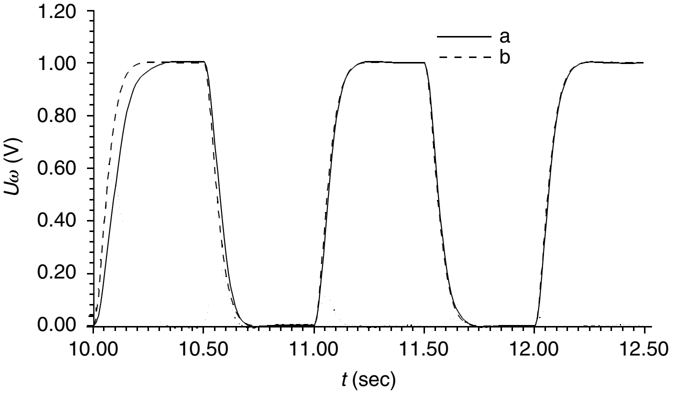


FIGURE 5.30 The measured angular speed responses in the case of restarted learning after a stepwise change of moment inertia $J = 3J_T$.

All practical experiments were made on the laboratory setup of a thyristor converter-fed DC servo drive (Figure 4.89). An analysis of the angular speed response to a stepwise change of the controller output $\Delta u = 0.151$ V yielded the following second-order approximation of the control object:

$$y_A(k) = 1.5499y_A(k-1) - 0.5642y_A(k-2) + 0.0939u(k) \tag{5.62}$$

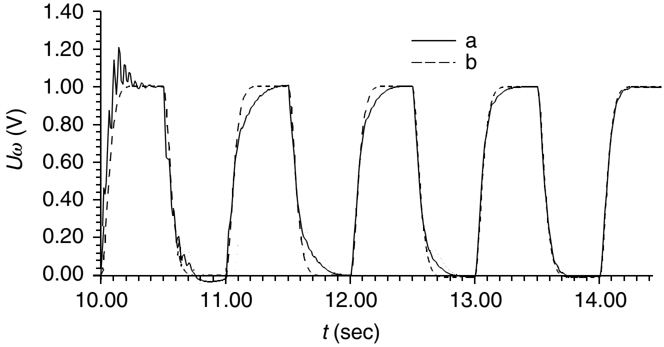


FIGURE 5.31 The measured angular speed responses in the case of restarted learning after a stepwise change of moment inertia $J = J_T/3$.

Experimentally obtained values of k_1, k_2, and k_3 were as follows: $k_1 = 0.2206$, $k_2 = -1.9000$, $k_3 = 0.151$. Because of using a single-phase thyristor converter, the angular speed response was noisy and, therefore, it had to be additionally filtered by calculating the mean value of the last two samples of measured angular speed u_ω. In order to cope with a new situation in the control system, reference input $u_r(k)$ in reference model Equation (4.53) was replaced in the algorithm by the delayed reference input signal $u_r(k-1)$.

The distribution of membership functions remained the same as in the simulation experiments (see Figure 5.20). Only zero subsets ZE and ZDE have changed their triangular form into trapezoidal in order to reduce the impact of the inherent "noise" in the angular speed feedback signal, which could provoke continuous triggering of the fuzzy controller output and thus lead to the overlearning effect.

Indicator I_e (see expression [5.61]) had a role to stop the learning process when its value (obtained for $N = 300$) had dropped below predetermined threshold value $I_e < 0.0017$, which in case of using a bipolar (±10 V) 12-bit A/D converter corresponded to 81 LSB.

The first group of experiments had the goal to test the performance of the fuzzy controller when learning had started with a blank fuzzy rule-table. The learning coefficients had the same values as in simulations: $\gamma_1 = 0.2$, $\gamma_2 = -0.18$, and $\gamma_3 = 0.018$. The operating point was set to $n_0 = 1000\ \text{min}^{-1}$. The reference model and the measured angular speed responses obtained in the first run and after ten runs in both directions are shown in Figure 5.32. In the first run, feedforward control signal u_F has a dominant influence on the system dynamics, while a self-organizing fuzzy controller prevails in the successive runs, enforcing the system to follow the reference model dynamics. The corresponding fuzzy controller output waveforms shown in Figure 5.33 are nonoscillatory and, therefore, suitable for smooth real-time operation. The final result of the self-organization is a fuzzy rule-table with singleton values displayed in Table 5.3. The values in the table corners remained at very low values because of very weak influence of induced system trajectories on them.

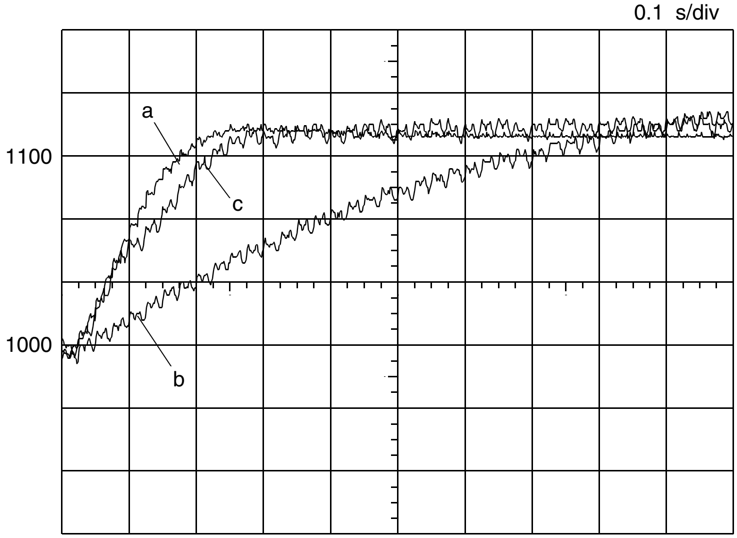


FIGURE 5.32 The measured angular speed responses (blank fuzzy rule-table): reference model (a), the 1st run (b), and the 10th run (c). (From Kovačić, Z., Bogdan, S., and Crnosija, P., *10th IEEE Intl. Symp. Intell. Contr.*, 389–394, 1995. With permission from Elsevier.)

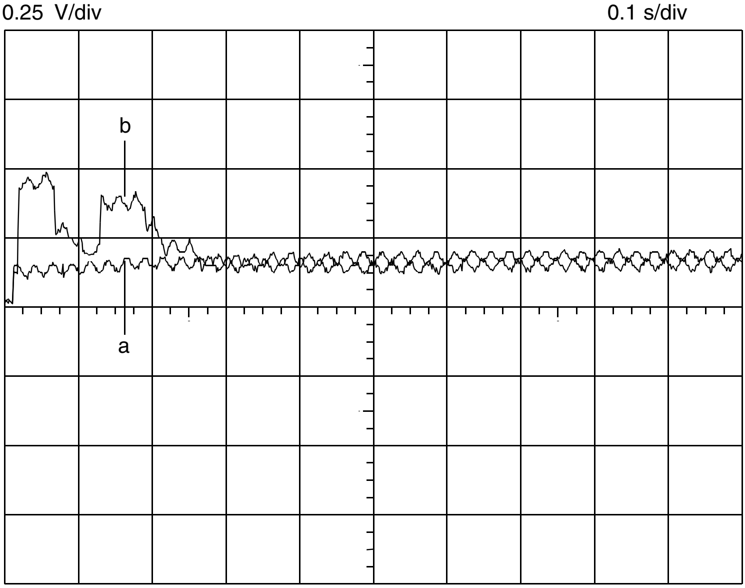


FIGURE 5.33 The fuzzy controller output responses (blank fuzzy rule-table): 1st run (a) and the 10th run (b).

TABLE 5.3
The Fuzzy Rule-Table Obtained after Learning (Experiment)

	NLE	NME	NSE	ZE	PSE	PME	PLE
NLDE	0	0	0	0	0.06	0.17	0.03
NMDE	0	0	0	0.02	0.33	0.61	0.23
NSDE	0	0	−0.05	−0.21	0.31	0.66	0.86
ZDE	−1.14	−0.22	−0.21	0	0.16	0.18	1.08
PSDE	−0.88	−0.75	−0.43	0.17	0.04	0	0
PMDE	−0.23	−0.6	−0.36	−0.03	0	0	0
PLDE	−0.03	−0.17	−0.06	0	0	0	0

TABLE 5.4
Preset (Shaded) and Modified by Learning (Unshaded) Fuzzy Rule-Table (Experiment)

	NLE	NME	NSE	ZE	PSE	PME	PLE
NLDE	−0.65	−0.42	−0.19	0.03	0.25	0.48	0.71
	−0.64	−0.42	−0.19	−0.06	−0.76	0.42	0.73
NMDE	−0.66	−0.43	−0.2	0.02	0.24	0.47	0.7
	−0.66	−0.43	−0.21	−0.21	−0.18	0.54	0.78
NSDE	−0.67	−0.44	−0.21	0.01	0.23	0.46	0.69
	−0.67	−0.44	−0.28	−0.26	0.21	0.51	0.93
ZDE	−0.68	−0.45	−0.22	0	0.22	0.45	0.68
	−1.09	−0.46	−0.27	0	0.26	0.46	1.11
PSDE	−0.69	−0.46	−0.23	−0.01	0.21	0.44	0.67
	−0.93	−0.51	−0.21	0.26	0.27	0.44	0.67
PMDE	−0.7	−0.47	−0.24	−0.02	0.2	0.43	0.66
	−0.79	−0.54	−0.17	0.2	0.21	0.43	0.66
PLDE	−0.71	−0.48	−0.25	−0.03	0.19	0.42	0.65
	−0.73	−0.44	0.04	0.05	0.19	0.42	0.64

The next group of experiments had the goal to test the performance of a self-organizing fuzzy controller, when learning starts with a preset fuzzy rule-table. The singleton values obtained according to the model reference-based presetting algorithm (33) are displayed shaded in Table 5.4. Learning coefficients and the operating point were the same as in the previous experiment. The reference model and the measured angular speed responses obtained in the first run and after ten runs in both directions are shown in Figure 5.34. The system dynamics in the first run are much closer to the reference model dynamics than in the previous case, which means that the off-line presetting of fuzzy rule-table has significantly

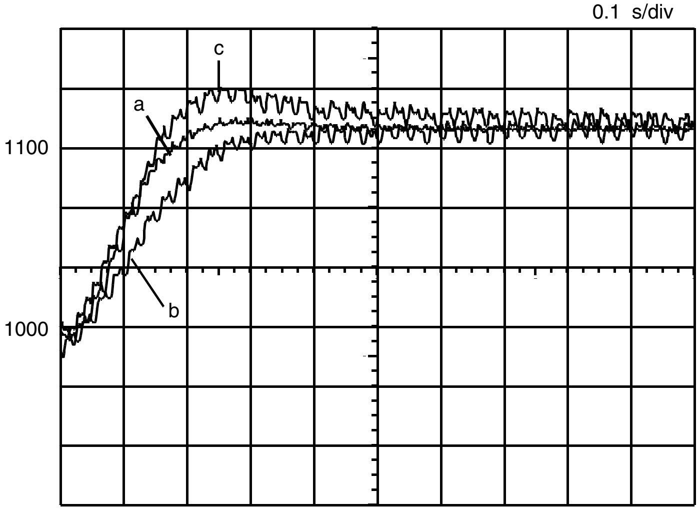


FIGURE 5.34 The measured angular speed responses (preset fuzzy rule-table): reference model (a), the 1st run (b), and the 10th run (c). (From Kovačić, Z., Bogdan, S., and Crnosija, P., *10th IEEE Intl. Symp. Intell. Contr.*, 389–394, 1995. With permission from Elsevier.)

improved system performance in the initial phase of learning. The fuzzy controller output responses shown in Figure 5.35 indicate very suitable forms for smooth control. The modified singleton values obtained after learning are displayed unshaded in Table 5.4. Since the influence of induced system trajectories has not reached the corners of fuzzy rule-table, those singleton values have remained almost unchanged, while the other values have been changed up to 50% of the preset values.

It was observed during experiments that the fuzzy controller starting self-organization with a blank fuzzy rule-table provided a more balanced (smoother) approach of the system response to the reference model response than the controller, that developed its control surface from the preset fuzzy rule-table. This can be explained by the characteristics of the preset control surface and the self-organizing mechanism, respectively. The singleton values of the preset control surface differ more or less from the final singleton values of the nonlinear control surface form. For those values that differ more, the learning mechanism needs more time to complete their modification, having in mind that learning coefficients usually (here, too) have rather small values.

5.3 SELF-ORGANIZING FUZZY CONTROL BASED ON SENSITIVITY FUNCTIONS

In Section 5.2, we have described an analytical self-organization method used for adjustment of fuzzy controller parameters, which is executed in every control

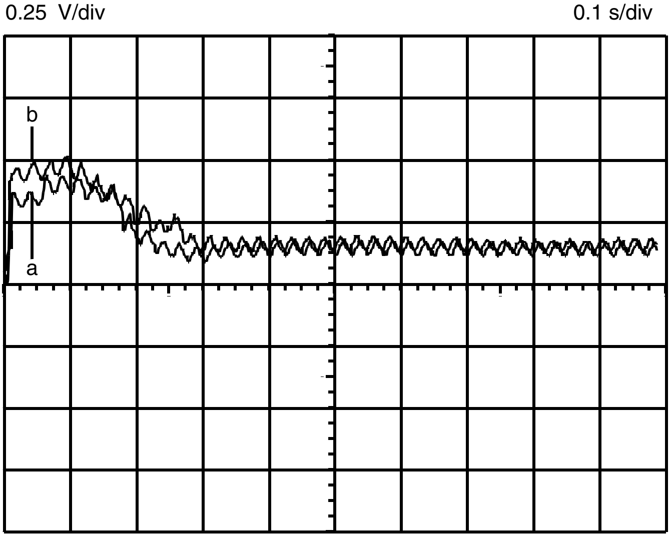


Figure 5.35 The fuzzy controller output responses (preset fuzzy rule-table): 1st run (a) and the 10th run (b).

interval T_{d}. In Section 4.2.2.1 we have described a sensitivity model-based adaptation method where changes of lead–lag compensator parameters are made in consecutive system runs (see tuning law [4.30]) in the moments when parameters have the highest influence on the system response. Applying the same strategy, here we describe a design of a model reference-based and a sensitivity model-based learning algorithm, which makes changes to the fuzzy controller parameter vector $\boldsymbol{\lambda}_{\mathrm{c}} = [\lambda_{\mathrm{c}1}, \ldots, \lambda_{\mathrm{cm}}]$ once in every run of the system, creating thus a discrete system with two sampling rates: a basic control rate and a learning rate.

5.3.1 Basic Concept of System Sensitivity

The fact that the influence of controller parameter λ_{ci} on the system response is not the same in the whole state space, and moreover, that it depends on the current system state, has lead to the strategy that a particular fuzzy controller parameter should be changed when its influence on the system response is the highest. This dependence of the fuzzy controlled system response changes on the changes of fuzzy controller parameters can be analytically expressed with sensitivity functions [23,34].

The sensitivity theory was developed from calculus as a result of investigation on the influence of differential equations parameters on their solutions. H.W. Bode was one of the pioneers in the field who noticed the possibilities that sensitivity theory offers for analysis and synthesis of control algorithms. We may say that closing the control loop with a feedback path was initially driven by a wish to decrease the system sensitivity to parameter variations. Wilkie and Perkins showed in Reference 35, a procedure for generation of sensitivity functions. Numerous contributions followed in 1960s and 1970s, equally from a theoretical [36–40] and practical aspect [41,42]. Sensitivity functions were used for synthesis of control algorithms in adaptive control systems [43,44]. Very often, control system synthesis is made based on the simplified models with constant parameter values. On the other hand, parameter variations cause inevitable changes of system dynamics. Sensitivity theory has had an important role in parameter optimization control strategies [45–47].

If they can be determined, sensitivity functions will show how and how much a particular parameter affects the process. Also, sensitivity functions will point out how and how much a particular parameter should be changed in order a desired change in process dynamics is achieved. In this way, one may design sensitivity functions-based parameter identification procedures that are executed in real-time [44]. This actually means that continuous adjustment of controller parameters becomes possible.

A sensitivity-based self-learning fuzzy logic control (SLFLC) scheme is thought for control of a class of unknown time-varying nonlinear high-order SISO control processes described by (5.40). The fuzzy controller to be organized by on-line learning is a PD-type Takagi–Sugeno zero-order fuzzy controller described with fuzzy control rules (2.14).

From the sensitivity theory point of view the system of Equations (5.40) can be viewed as a description of influence of the parameter vector $\boldsymbol{\lambda}$ on the state vector $\mathbf{x}$. If initial state variables values $\mathbf{x}(t_0) = \mathbf{x}_0$, the system structure and the input signal remain unchanged, then the changes of parameter vector $\Delta\boldsymbol{\lambda}$ will be reflected in the changes of state vector $\Delta\mathbf{x}$.

Figure 5.36 shows the trace of state vector $\mathbf{x}$ in the state space provoked by system parameter variations. Trajectory $\mathbf{x}^1$ obtained with parameter vector $\boldsymbol{\lambda}^1$ is different from trajectory $\mathbf{x}^2$ obtained with parameter vector $\boldsymbol{\lambda}^2$. The range of changes of the state vector $\Delta\mathbf{x}$ due to changes of the parameter vector $\Delta\boldsymbol{\lambda}$ is uniquely determined by the system of Equations (5.40).

The relation between $\Delta\mathbf{x}$ and $\Delta\boldsymbol{\lambda}$ for small parameter variations around nominal parameter values vector $\boldsymbol{\lambda}_0$ is as follows:

$$\Delta\mathbf{x} \approx \mathbf{S}(\boldsymbol{\lambda}_0)\Delta\boldsymbol{\lambda} = \left[S_{ij} = \left.\frac{\partial x_i}{\partial \lambda_j}\right|_{\boldsymbol{\lambda}_0} \right] \Delta\boldsymbol{\lambda} \tag{5.63}$$

where $\mathbf{S}$ is the sensitivity function matrix, and S_{ij} is an element of matrix $\mathbf{S}$.

There are several ways to define system output sensitivity functions. Since the self-organization process is a real-time process, it is convenient to determine a sensitivity function vector in the time domain. Let us suppose that the solution of system (5.40) for nominal parameters has a form:

$$y_{\mathrm{f}0} = y_{\mathrm{f}}(u, t, \boldsymbol{\lambda}_0) \tag{5.64}$$

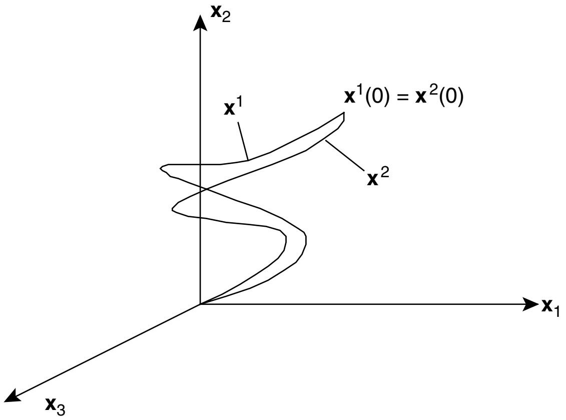


FIGURE 5.36 The influence of the parameter vector on the state variable vector's trajectories.

For changed parameter values, system (5.40) has a solution:

$$y_{\mathrm{f}} = y_{\mathrm{f}}(u, t, \boldsymbol{\lambda}) \tag{5.65}$$

Under assumption that y_{f} is a continuous function of parameter vector $\boldsymbol{\lambda}$, Equation (5.65) can be rewritten in the form of a Taylor series around nominal solution y_{f0}:

$$y_{\mathrm{f}} = y_{\mathrm{f0}} + \left.\frac{\partial y_{\mathrm{f}}}{\partial \boldsymbol{\lambda}}\right|_{\boldsymbol{\lambda}_0} \Delta\boldsymbol{\lambda} + \frac{1}{2}\left.\frac{\partial^2 y_{\mathrm{f}}}{\partial \boldsymbol{\lambda}^2}\right|_{\boldsymbol{\lambda}_0} \Delta\boldsymbol{\lambda}^2 + \cdots \tag{5.66}$$

For small parameter variations around nominal values, high-order Taylor series components can be neglected, and in that case (5.66) assumes a linear form:

$$y_{\mathrm{f}} = y_{\mathrm{f0}} + \left.\frac{\partial y_{\mathrm{f}}}{\partial \boldsymbol{\lambda}}\right|_{\boldsymbol{\lambda}_0} \Delta\boldsymbol{\lambda} \tag{5.67}$$

Relation (5.67) can be rewritten as:

$$\Delta y_{\mathrm{f}} = y_{\mathrm{f}} - y_{\mathrm{f0}} \approx \left.\frac{\partial y_{\mathrm{f}}}{\partial \boldsymbol{\lambda}}\right|_{\boldsymbol{\lambda}_0} \Delta\boldsymbol{\lambda} = \boldsymbol{\eta}_{\lambda}\Delta\boldsymbol{\lambda} \tag{5.68}$$

where $\boldsymbol{\eta}_{\lambda}$ is the system output sensitivity function vector of the following form:

$$\boldsymbol{\eta}_{\lambda} = \left[\frac{\partial y_{\mathrm{f}}}{\partial \lambda_1} \quad \frac{\partial y_{\mathrm{f}}}{\partial \lambda_2} \quad \cdots \quad \frac{\partial y_{\mathrm{f}}}{\partial \lambda_v}\right]_{\boldsymbol{\lambda}_0} \tag{5.69}$$

Vector $\boldsymbol{\eta}_{\lambda}$ represents a gradient of the system output with respect to system parameter variations. Although the self-organization method uses vector $\boldsymbol{\eta}_{\lambda}$, the method is essentially different from well-known gradient-based optimization methods [48–51]. Gradient-based self-organizing methods calculate and set up a new parameter vector in each control interval. The method described here calculates a new parameter vector in the moments when sensitivity functions in (5.69) reach their maximum and then sets a new vector up before a new change of the reference input u_{r} takes place.

Vector $\boldsymbol{\eta}_{\lambda}$ will retain the same form also for discrete control systems providing that control interval T_{d} is kept constant. In References 52 to 54 one may find more information about control systems sensitivity to variations of T_{d}. This fact is important for the design of a digital self-organizing fuzzy controller, as we may use vector $\boldsymbol{\eta}_{\lambda}$, which has been found for the corresponding continuous control system.

We have mentioned that sensitivity functions can be used not only for the analysis but also for the synthesis of control systems. In general, parameter vector $\boldsymbol{\lambda}$ contains controller parameters $\boldsymbol{\lambda}_{\mathrm{c}}$ and process parameters $\boldsymbol{\lambda}_{\mathrm{p}}$. If the response of a nominal system (5.64) is viewed as a predefined (desired) system response, and the response (5.65) as an actual system response, then by means of Equation (5.68) we can search for values of vector $\Delta\boldsymbol{\lambda}_{\mathrm{c}}$ components, which would compensate for variations of $\boldsymbol{\lambda}_{\mathrm{p}}$ and gradually push Δy_{f} toward zero. This would result in the system's follow-up of predefined system dynamics. By that, one must take care about the system stability, ensuring that values of parameter changes are much less than nominal parameter values [56].

5.3.2 Synthesis of a Self-Organizing Fuzzy Algorithm

By switching to the discrete time domain, the changes of system output $y_{\mathrm{f}}(k)$ due to small variations of system parameters are given by the following equation:

$$\Delta y_{\mathrm{f}}(k, \boldsymbol{\lambda}) = \sum_{i} \eta_{\lambda_i}(k)\big|_{\lambda_i} \Delta\lambda_i \tag{5.70}$$

Equation (5.70) can be used as a starting point for determination of vector $\Delta\boldsymbol{\lambda}$ whose aim is to produce the required change $\Delta y_{\mathrm{f}}(k, \boldsymbol{\lambda})$. One way to solve (5.70) is to write down n equations for n successive control intervals, where n denotes dimension of parameter vector $\boldsymbol{\lambda}$. The other way is to induce the given change of system output by changing only one parameter λ_i. Then a new parameter value is used for calculation of another vector component. The procedure is repeated until the tracking error assumes an acceptably small value. This method was effectively applied for determination of dead-beat controller parameters [57,58].

The structure of a self-learning fuzzy controller may vary depending on the type of process. The structure described by (5.42) contains a PD-type fuzzy controller and a feedforward control element. In order to establish steady-state accuracy and to cancel disturbance effects, an integral element can also be added. Let us consider the self-organization of a controller with the following structure:

$$u(k) = \Gamma[e(k), \Delta y_\mathrm{f}(k), \boldsymbol{\lambda}] + k_3(k)u_\mathrm{r}(k) = \sum A_i\varphi_i[e(k), \Delta y_\mathrm{f}(k)] + k_3(k)u_\mathrm{r}(k) \tag{5.71}$$

The difference with respect to structure (5.42) is in different fuzzy controller inputs. The change of error $\Delta e(k)$ applied in (5.42) is replaced in (5.71) by the change of system output $\Delta y_\mathrm{f}(k)$. Under assumption of constant or slow-varying reference input $u_\mathrm{r}(k)$, $\Delta e(k)$ and $\Delta y_\mathrm{f}(k)$ are related as

$$\Delta e(k) = -\Delta y_\mathrm{f}(k) \tag{5.72}$$

The change of the sign of one input variable affects fuzzy control rules but it does not change the structure of the fuzzy controller. The reason why $\Delta y_\mathrm{f}(k)$ is preferred to $\Delta e(k)$ is in its more convenient response to stepwise changes of the reference input. Namely, in the control interval when $\Delta u_\mathrm{r}(k)$ takes place, $\Delta e(1) = \Delta u_\mathrm{r}(1)$, while in the forthcoming intervals $\Delta e(1) \gg \Delta e(2), \Delta e(3), \ldots, \Delta e(k)$. This means that $\Delta e(1)$ is excessive in magnitude with respect to "regular" control error values $\Delta e(k)$, which does not occur with $\Delta y_\mathrm{f}(k)$.

A sensitivity model of a fuzzy controlled system can be built only if fuzzification and defuzzification operations have a form that allows a fuzzy input–output mapping function to assume an analytical and differentiable form. In this sense, the inference engine will utilize a product operator:

$$\begin{gathered}\mu_j[e(k), \Delta y_\mathrm{f}(k)] = \mu_k^e[e(k)] \cdot \mu_l^{\Delta y_\mathrm{f}}[\Delta y_\mathrm{f}(k)],\\ j = 1, 2, \ldots, r, \quad k = 1, 2, \ldots, p, \quad l = 1, 2, \ldots, q\end{gathered} \tag{5.73}$$

and input membership functions will also have a differentiable form:

$$\mu_k^e(x) = \mathrm{e}^{-(x-c_k^e)^2/(2(w_k^e)^2)}, \qquad \mu_l^{\Delta y_\mathrm{f}}(x) = \mathrm{e}^{-(x-c_l^{\Delta y_\mathrm{f}})^2/(2(w_l^{\Delta y_\mathrm{f}})^2)} \tag{5.74}$$

where c_i^j is the center of a membership function μ_i^j, and w_i^j is the width of a membership function μ_i^j.

Accordingly, controller parameter vector $\boldsymbol{\lambda}_c$ contains the following parameters: output singletons A_i, centers of input sets, c_i^e and $c_i^{\Delta yf}$, and widths of input sets, w_i^e and $w_i^{\Delta yf}$.

The crisp output of the fuzzy controller used in the SLFLC scheme is computed according to the center of gravity principle (2.23). It should be noted that this defuzzification algorithm has a differentiable form which is convenient for implementation of the SLFLC algorithm. How much the ith fuzzy control rule will contribute to a crisp controller output depends on the degree of contribution described with the fuzzy basis function:

$$\begin{aligned}\varphi_i[e(k), \Delta y_{\mathrm{f}}(k)] &= \frac{\mu_k^e[e(k)] \cdot \mu_l^{\Delta y_{\mathrm{f}}}[\Delta y_{\mathrm{f}}(k)]}{\sum_{m=1}^{p} \sum_{n=1}^{q} \mu_m^e[e(k)] \cdot \mu_n^{\Delta y_{\mathrm{f}}}[\Delta y_{\mathrm{f}}(k)]} \\ &= \frac{\mu_i[e(k), \Delta y_{\mathrm{f}}(k)]}{\sum_{j=1}^{r} \mu_j[e(k), \Delta y_{\mathrm{f}}(k)]} \end{aligned} \tag{5.75}$$

whose form is also differentiable.

We have shown that sensitivity functions can be very useful to express how much influence some variable or parameter has on the focused variable. In this sense, sensitivity functions represent information about interactions between causes and consequences, which may be very useful for planning interventions in the system.

Having this concept in mind, a total differential of the system output with respect to small variations of fuzzy controller parameters is determined by

$$\Delta y_{\mathrm{f}}(k, \boldsymbol{\lambda}) = \sum_{i}^{n} \eta_{\lambda_{ci}}(k)\bigg|_{\lambda_{ci}} \Delta\lambda_{ci} \tag{5.76}$$

where $\eta_{\lambda_{ci}}(k), i = 1, \ldots, n$, are system output sensitivity functions related to fuzzy controller parameters:

$$\eta_{\lambda_{ci}}(k) = \frac{\partial y_{\mathrm{f}}(k, \boldsymbol{\lambda})}{\partial \lambda_{ci}} = \frac{\partial y_{\mathrm{f}}(k, \boldsymbol{\lambda})}{\partial u(k)} \frac{\partial u(k)}{\partial \lambda_{ci}} = G_{\mathrm{p}} \frac{\partial u(k)}{\partial \lambda_{ci}} \tag{5.77}$$

where G_{p} is the process transfer function.

It may be seen that sensitivity functions related to controller parameters λ_{ci} depend on generally unknown dynamic characteristics of the control process G_{p}.

The sensitivity function of the controller output with respect to controller parameter variations has the form

$$\frac{\partial u(k)}{\partial \lambda_{ci}} = \frac{\partial}{\partial \lambda_{ci}} \{\Gamma[e(k), \Delta y_{\mathrm{f}}(k), \boldsymbol{\lambda}]\} \tag{5.78}$$

Insertion of (5.78) into (5.77) and further in (5.76) yields:

$$\Delta y_{\mathrm{f}}(k) \doteq \sum_{i=1}^{n} G_{\mathrm{p}} \left\{ \frac{\partial}{\partial \lambda_{ci}} \{\Gamma[e(k), \Delta y_{\mathrm{f}}(k), \boldsymbol{\lambda}]\} \right\} \Delta\lambda_{ci} \tag{5.79}$$

Equation (5.79) can be used for assessment of fuzzy controller parameter variations that would provide the given change of system output. The experience in commissioning of fuzzy controllers suggests that output singletons are parameters with the largest influence on the fuzzy controller output. Referring to controller function (5.71), system parameter variations will be compensated only by modifying the fuzzy output singletons A_i and the feedforward coefficient k_3 (input membership functions remain predefined), that is, $\boldsymbol{\lambda}_\mathrm{c} = [A_1 \quad A_2 \quad \cdots \quad A_\mathrm{r} \quad k_3]^\mathrm{T}$. Therefore, Equation (5.78) can be split into two parts:

$$\begin{aligned} \eta_{A_i}(k) &= \frac{\partial y_\mathrm{f}(k,\boldsymbol{\lambda})}{\partial A_i} = \frac{\partial y_\mathrm{f}(k,\boldsymbol{\lambda})}{\partial u(k)}\frac{\partial u(k)}{\partial A_i} = G_\mathrm{p}\frac{\partial u(k)}{\partial A_i} \\ \eta_{k_3}(k) &= \frac{\partial y_\mathrm{f}(k,\boldsymbol{\lambda})}{\partial k_3} = \frac{\partial y_\mathrm{f}(k,\boldsymbol{\lambda})}{\partial u(k)}\frac{\partial u(k)}{\partial k_3} = G_\mathrm{p}\frac{\partial u(k)}{\partial k_3} \end{aligned} \tag{5.80}$$

From (5.71) and (5.75) we can determine controller output sensitivity functions with respect to the components of vector $\boldsymbol{\lambda}_\mathrm{c}$:

$$\begin{aligned} \frac{\partial u(k)}{\partial A_i} &= \frac{\partial}{\partial A_i}\{\Gamma[e(k), \Delta y_\mathrm{f}(k), \boldsymbol{\lambda}]\} = \varphi_i[e(k), \Delta y_\mathrm{f}(k)] \\ \frac{\partial u(k)}{\partial k_3} &= \frac{\partial}{\partial k_3}[k_3 u_\mathrm{r}(k)] = u_\mathrm{r}(k) \end{aligned} \tag{5.81}$$

Accordingly, Equation (5.80) assumes the form:

$$\begin{aligned} \eta_{A_i}(k) &= G_\mathrm{p}\varphi_i[e(k), \Delta y_\mathrm{f}(k)] \\ \eta_{k_3}(k) &= G_\mathrm{p} u_\mathrm{r}(k) \end{aligned} \tag{5.82}$$

From (5.82) we can see that the self-organization algorithm is not so demanding from the computational point of view because the reference input $u_\mathrm{r}(k)$ is already known, while the fuzzy basis functions are normally calculated during the calculation of crisp fuzzy controller output (see [2.23]).

Sensitivity functions in (5.82) depend on the dynamic characteristics of the control process G_p. Since the exact model of the control process (5.40) is unknown (only a static process gain is directly identifiable), some approximation, denoted as G_pa, must be used. Among many possible linear process approximations, the one determined by the reference model dynamics (3.28) could be assumed to be a reasonable choice. Then $G_\mathrm{pa} = G_\mathrm{M}$. If such an approximation is adopted, then the sensitivity functions described by (5.82) have the form (for simplicity, $\phi_i[e(k), \Delta y_\mathrm{f}(k)]$ is replaced by $\phi_i(k)$):

$$\begin{aligned} \eta_{A_i}(k) &= a_\mathrm{M1}\cdot\eta_{A_i}(k-1) + a_\mathrm{M2}\cdot\eta_{A_i}(k-2) \\ &\qquad + b_\mathrm{M1}\cdot\varphi_{A_i}(k-1) + b_\mathrm{M1}\cdot\varphi_{A_i}(k-2) \\ \eta_{k_3}(k) &= a_\mathrm{M1}\cdot\eta_{k_3}(k-1) + a_\mathrm{M2}\cdot\eta_{k_3}(k-2) \\ &\qquad + b_\mathrm{M1}\cdot u_\mathrm{r}(k-1) + b_\mathrm{M1}\cdot u_\mathrm{r}(k-2) \end{aligned} \tag{5.83}$$

In general, selected approximations of the control process (5.40) may have different dynamic characteristics, which would finally result in different dynamic characteristics of the sensitivity functions (5.83). If such sensitivity functions are being used in the SLFLC algorithm, then the process approximation G_{pa} would act like a filter whose dynamic behavior determines the speed of learning and thus directly affects the final form of the fuzzy controller. In other words, after completion of the learning process, different variants of the fuzzy controller will be obtained for different process approximations.

Let us make the following assumptions: the static part of the control process is inherently stable, the boundary values (limits) of the system output $y_{\mathrm{f}}(k)$ and the tracking error $e_{\mathrm{M}}(k)$ are known, the reference input $u_{\mathrm{r}}(k)$ is imposed as a sequence of alternating step changes. As we have already discussed in Section 4.2.2.1, the given change of system output $\Delta y_{\mathrm{f}}(k)$ coincides in the model reference control concept with model tracking error $e_{\mathrm{M}}(k)$. Therefore, relation (5.76) assumes the form

$$e_{\mathrm{M}}(k) = \sum_{i=1}^{r} \eta_{A_i}(k)\Delta A_i + \eta_{k3}(k)\Delta k_3 \tag{5.84}$$

which can be used for determination of vector $\Delta\boldsymbol{\lambda}_{\mathrm{c}}$. Since tracking error $e_{\mathrm{M}}(k)$ is known in every control interval, Equation (5.84) converts into a learning algorithm of the form:

$$\Delta A_j^{\kappa}(k) = \frac{e_{\mathrm{M}}^{\kappa}(k) - \sum_{i=1,i\neq j}^{r} \eta_{A_i}^{\kappa}(k)\cdot\Delta A_i^{\kappa} - \eta_{k_3}^{\kappa}(k)\cdot\Delta k_3^{\kappa}}{\eta_{A_j}^{\kappa}(k)} \tag{5.85}$$

where κ denotes the current learning iteration (i.e., the current run of the system).

The meaning of this is that singletons A_j are changed only once during each run of the system. A new run of the system starts with a new change of $u_{\mathrm{r}}(k)$. It must be noted, that approximate equality in (5.79) is intentionally replaced by normal equality in (5.85) and (5.86).

The modification of the feedforward gain coefficient k_3 is always performed after the modification of all singleton values in the current (kth) learning interval is completed:

$$\Delta k_3^{\kappa} = \frac{e_{\mathrm{M}}^{\kappa}(k) - \sum_{i=1}^{r} \eta_{A_i}^{\kappa}(k)\cdot\Delta A_i^{\kappa}}{\eta_{k_3}^{\kappa}(k)} \tag{5.86}$$

This is done in the moment when the influence of a particular controller parameter is the highest, and that is in the maximum of the corresponding sensitivity function. From (5.75) one can see that fuzzy basis function φ_i is a convex exponential function obtained as a product of two Gaussian membership functions μ_k^e and $\mu_l^{\Delta y_{\mathrm{f}}}$. From the implementation point of view it is easy to determine the maximum of the Gaussian-type convex function. Sensitivity function $\eta_{k_3}^{\kappa}(k)$ is actually represented by the reference model response. Since we use a stable reference model, it is sufficient to keep the period of reference input changes long enough that the reference model reaches the steady state, as the steady-state value of $\eta_{k_3}^{\kappa}(k)$ is equal to the maximal one.

Accordingly,

$$\begin{aligned} \beta_j^{\kappa} &= \eta_{A_j}^{\kappa}(k_j^{\kappa}) = \max\left[\eta_{A_j}^{\kappa}(k)\right] \\ \beta_{k_3}^{\kappa} &= \eta_{k_3}^{\kappa}(k_{k_3}^{\kappa}) = \max\left[\eta_{k_3}^{\kappa}(k)\right] \end{aligned} \tag{5.87}$$

where k_j^{κ} and k_{k3}^{κ} are the moments when the maximum of corresponding sensitivity functions are detected.

On insertion of (5.87) into (5.85) and (5.86), the learning laws assume the form

$$\Delta A_j^{\kappa}(k) = \frac{e_{\mathrm{M}}^{\kappa}(k) - \sum_{i=1,i\neq j}^{r} \eta_{A_i}^{\kappa}(k)\cdot\Delta A_i^{\kappa} - \eta_{k_3}^{\kappa}(k)\cdot\Delta k_3^{\kappa}}{\beta_j^{\kappa}} \tag{5.88}$$

$$\Delta k_3^{\kappa} = \frac{e_{\mathrm{M}}^{\kappa}(k) - \sum_{i=1}^{r} \eta_{A_i}^{\kappa}(k)\cdot\Delta A_i^{\kappa}}{\beta_{k_3}^{\kappa}} \tag{5.89}$$

The learning speed depends on dynamic characteristics of the sensitivity functions generated during the system output transient response. By applying maximal values β_j^{κ} and $\beta_{k_3}^{\kappa}$, division in learning laws (5.88) and (5.89) is thus executed with larger numbers, which additionally contributes to the smooth convergence of learning and the stability of self-organization. If we would allow larger changes of controller parameters, then we would depart from the assumption that we deal with small parameter changes (see Equations [5.67] and [5.70]). During the learning interval, it may occur that some sensitivity function reaches a maximum value lower than the predetermined threshold value $\beta_{\min}$. This means that the corresponding singleton has a negligible influence on the process behavior, and so it will remain unchanged. By introducing a minimal threshold value $\beta_{\min}$, we also avoid a possible problem of division by zero.

The modification of controller parameter vector $\boldsymbol{\lambda}_{\mathrm{c}}^{\kappa}$ is executed by using the standard tuning law

$$\boldsymbol{\lambda}_{\mathrm{c}}^{\kappa+1} = \boldsymbol{\lambda}_{\mathrm{c}}^{\kappa} + \Delta\boldsymbol{\lambda}_{\mathrm{c}}^{\kappa} \tag{5.90}$$

Normally, learning starts from a blank fuzzy rule-table, and proceeds after each run of the system by adding changes to the singletons of the activated fuzzy control rules. Alternatively, learning may start from a preset fuzzy rule-table as discussed

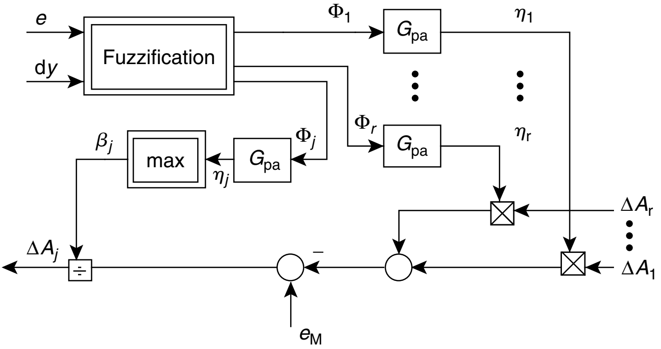


FIGURE 5.37 The block diagram of the learning mechanism. (From Kovačić, Z., Balenović, M., and Bogdan, S., *IEEE Contr. Syst. Mag.*, 18(3), 41–51, 1998. With permission.)

in Chapter 3. In order to prevent the overlearning problem that may occur, some learning stoppage criterion must be used. For example, one can use an integral criterion related to the absolute model tracking error given by

$$I_{e_M}^{\kappa} = \sum_{k=0}^{k_s} |e_M^{\kappa}(k)| \leq \delta_0 \tag{5.91}$$

where k_s is the last control interval of the current learning iteration and δ_0 is a predefined threshold value.

Another possible way among many others is to stop learning by observing the mean square ratio of parameter changes with respect to their full values:

$$I_A^{\kappa} = \frac{\sum_{i=1}^{r} (\Delta A_i^{\kappa})^2}{\sum_{i=1}^{r} (A_i^{\kappa})^2} \cdot 100\% \leq I_0 \tag{5.92}$$

Usually, the threshold value I_0 is set to 5 to 10%. In order to safely avoid an overlearning effect, criteria like (5.91) and (5.92) can be combined together.

The block diagram of the learning mechanism implementing the learning law (5.88) is shown in Figure 5.37.

Example 5.3 Sensitivity model-based self-organizing fuzzy control — linear case.

In order to illustrate the effectiveness of a sensitivity model-based self-organizing fuzzy controller, let us describe the design and implementation for the case of controlling a third-order linear system (3.56), already used in Section 3.4 for testing several fuzzy controller initial setting methods. For this purpose, a laboratory process simulator device Feedback PCS 327 will be used.

Each fuzzy controller input has five fuzzy sets, and accordingly, the fuzzy rule-table has 25 control rules. Input membership functions have a Gaussian form

(5.74) and α-cut ($\alpha = 0.05$) is applied to them (see relation [2.26]). The centers and widths of input membership functions are the following:

$$
\begin{aligned}
&c_{\mathrm{NL}}^{e} = c_{\mathrm{NL}}^{\Delta y_{\mathrm{f}}} = -1\\
&c_{\mathrm{NM}}^{e} = c_{\mathrm{NM}}^{\Delta y_{\mathrm{f}}} = -0.4\\
&c_{\mathrm{Z}}^{e} = c_{\mathrm{Z}}^{\Delta y_{\mathrm{f}}} = 0\\
&c_{\mathrm{PM}}^{e} = c_{\mathrm{PM}}^{\Delta y_{\mathrm{f}}} = 0.4\\
&c_{\mathrm{PL}}^{e} = c_{\mathrm{PL}}^{\Delta y_{\mathrm{f}}} = 1\\
&w_{\mathrm{NL}}^{e} = w_{\mathrm{NL}}^{\Delta y_{\mathrm{f}}} = w_{\mathrm{PL}}^{e} = w_{\mathrm{PL}}^{\Delta y_{\mathrm{f}}} = 0.227\\
&w_{\mathrm{NM}}^{e} = w_{\mathrm{NM}}^{\Delta y_{\mathrm{f}}} = w_{\mathrm{PM}}^{e} = w_{\mathrm{PM}}^{\Delta y_{\mathrm{f}}} = 0.048\\
&w_{\mathrm{Z}}^{e} = w_{\mathrm{Z}}^{\Delta y_{\mathrm{f}}} = 0.01
\end{aligned}
$$

One can see from the above listed center values that universes of discourse are scaled to the interval $[-1, 1]$. Scaling coefficient values are $K_{\mathrm{e}} = 1$ and $K_{\Delta y_{\mathrm{f}}} = 25$. The minimal threshold value for the sensitivity functions is set to be $\beta_{\mathrm{min}} = 0.1$.

Desired dynamic characteristics of a high-order closed-loop control system are described by a second-order reference model, whose parameters are determined according to selected performance indices: the response overshoot, $\sigma_{\mathrm{m}} = 5\%$, and the peak time $t_{\mathrm{m}} = 5$ sec, which yields (sampling interval $T_{\mathrm{d}} = 0.1$ sec):

$$
\begin{aligned}
y_{\mathrm{M}}(k) = 1.8799 y_{\mathrm{M}}(k-1) - 0.8871 y_{\mathrm{M}}(k-2)\\
+ 0.0036 u_{\mathrm{r}}(k-1) + 0.0035 u_{\mathrm{r}}(k-2)
\end{aligned}
\tag{5.93}
$$

The reference model (5.93) also represents process approximation G_{pa} that is used for calculation of sensitivity functions (see [5.82] and [5.83]).

The self-organization process is tested first for the nominal feedforward gain coefficient value $k_3 = 0.58$ (i.e., k_3 is reciprocal to the nominal control process gain). Stepwise changes of the reference input $\Delta u_{\mathrm{r}} = \pm 1$ V are fed into the system. Figure 5.38 shows the reference model and system output responses obtained after first three runs of the system. Figure 5.39 and Figure 5.40 shows the tracking error and controller output responses, respectively. In the first run, the dynamic behavior is determined only by the feedforward control element and the tracking error exceeds 50% of the imposed change of reference input. In the third run, the tracking error is already reduced to 10%.

Figures 5.41 to 5.43 show the same group of responses obtained after twelve positive and negative runs of the system. The system follows the reference model very closely and the steady-state tracking error is kept below 5%. The learning process converges smoothly, while the controller output (Figure 5.40 and Figure 5.43) has a desirable nonoscillatory form.

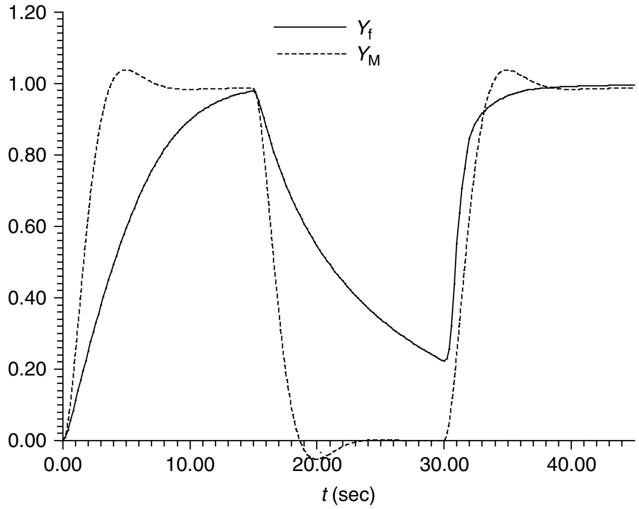


FIGURE 5.38 The system output and reference model responses at the beginning of learning. (From Bogdan, S. and Kovačić, Z., *4th IEEE Mediterr. Symp. Contr. Autom.*, 799–804, 1996. With permission.)

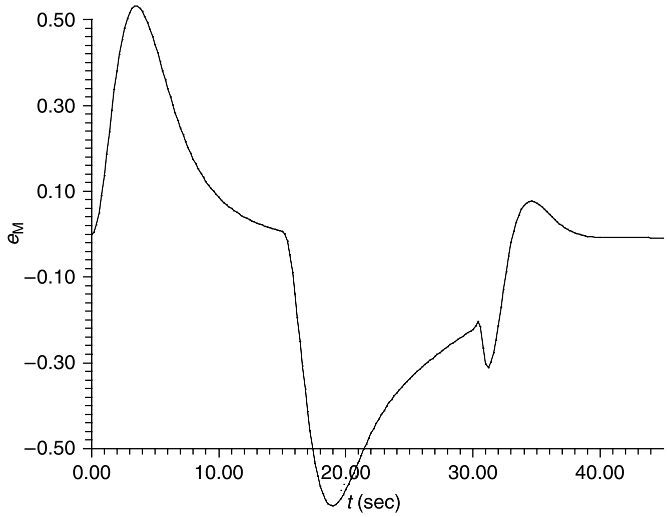


FIGURE 5.39 The tracking error responses at the beginning of learning. (From Bogdan, S. and Kovačić, Z., *4th IEEE Mediterr. Symp. Contr. Autom.*, 799–804, 1996. With permission.)

Figure 5.44 shows that for the nominal conditions the feedforward gain coefficient k_3 remains almost unchanged, that is, unaffected by the learning algorithm (5.89) operation.

Figure 5.45 shows the transition of the learning stoppage criterion I_A^κ. One can notice a very fast and smooth decrease of the criterion (the criterion dropped

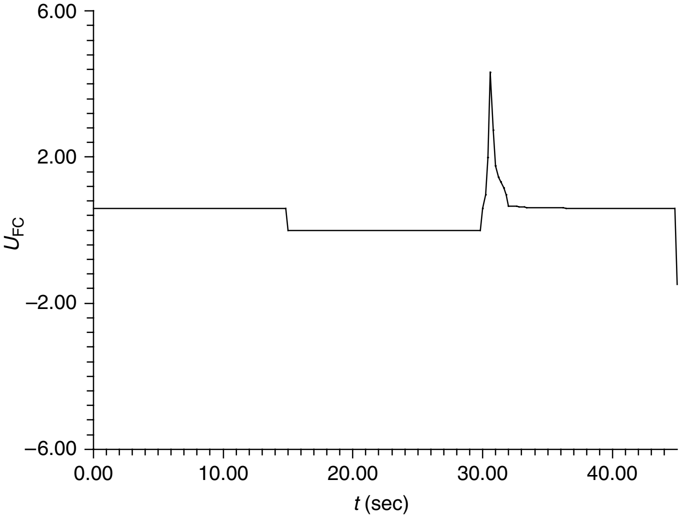


FIGURE 5.40 The self-learning fuzzy controller output responses at the beginning of learning. (From Bogdan, S. and Kovačić, Z., *4th IEEE Mediterr. Symp. Contr. Autom.*, 799–804, 1996. With permission.)

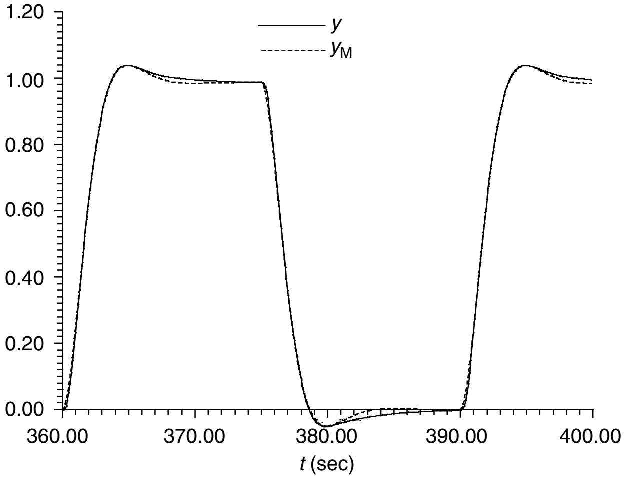


FIGURE 5.41 The system output and reference model responses at the end of learning. (From Bogdan, S. and Kovačić, Z., *4th IEEE Mediterr. Symp. Contr. Autom.*, 799–804, 1996. With permission.)

below 10% after five learning iterations). In the steady-state, the criterion varies around minimal value. The reasons why I_A^κ does not asymptotically approach zero lie in the usage of an approximate learning algorithm (due to the fact that only two terms of the Taylor series were taken into account in the derivation of sensitivity

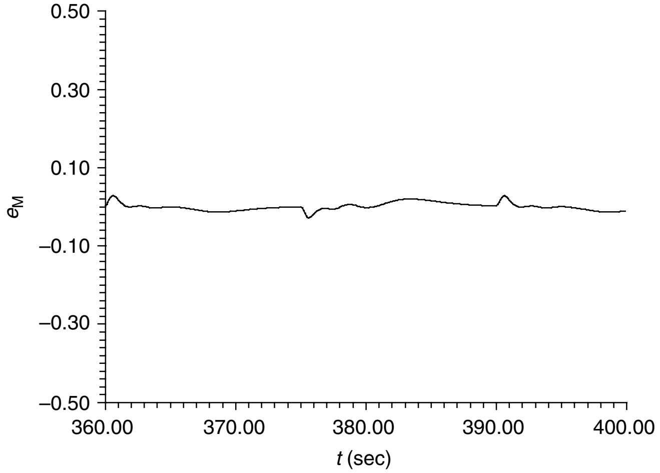


Figure 5.42 The tracking error responses at the end of learning. (From Bogdan, S. and Kovačić, Z., *4th IEEE Mediterr. Symp. Contr. Autom.*, 799–804, 1996. With permission.)

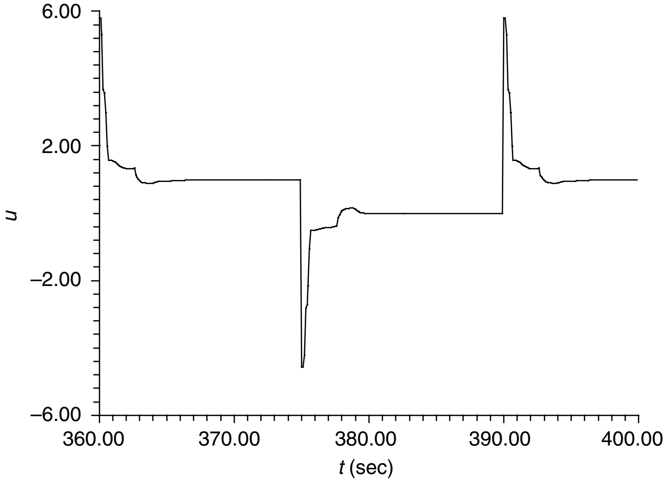


Figure 5.43 The self-learning fuzzy controller output responses at the end of learning. (From Bogdan, S. and Kovačić, Z., *4th IEEE Mediterr. Symp. Contr. Autom.*, 799–804, 1996. With permission.)

model-based learning algorithm) and the mismatch of the system and the reference model (the controlled process is a third-order system, while the reference model is a second-order system). In spite of the fact that the tracking error cannot be fully eliminated, the self-learning algorithm continuously attempts to generate singleton changes that would eliminate the error. This normally leads to the overlearning

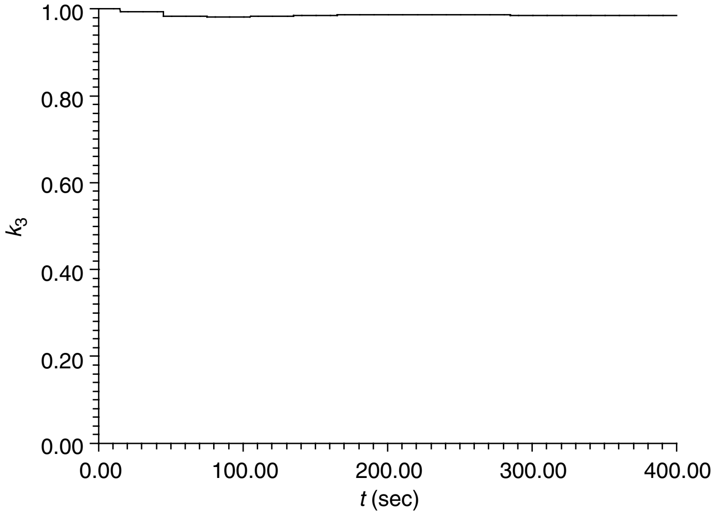


FIGURE 5.44 The transition of the feedforward gain coefficient.

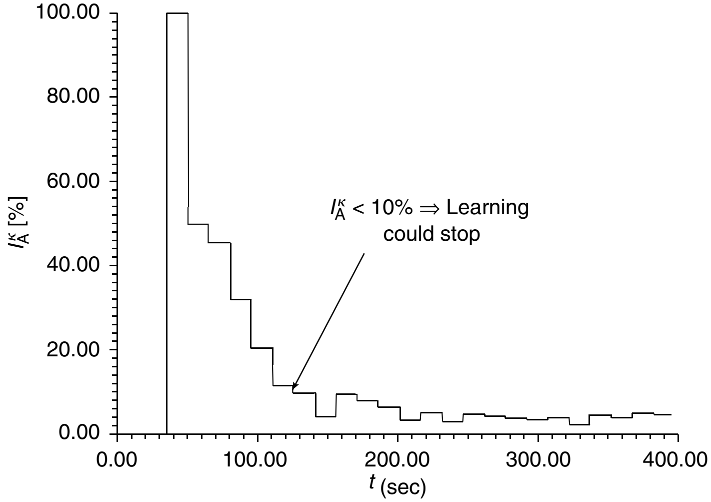


FIGURE 5.45 The transition of the learning stoppage criterion.

problem. As denoted in Figure 5.45, the 10% value of I_A^κ could be used as a threshold value to stop self-organization and thus prevent overlearning.

In the next experiment, learning has started with a supposed feedforward gain coefficient value $k_3 = 0.7$. Figure 5.46 shows the reference model and system output responses obtained after first three runs of the system. Figure 5.47 and

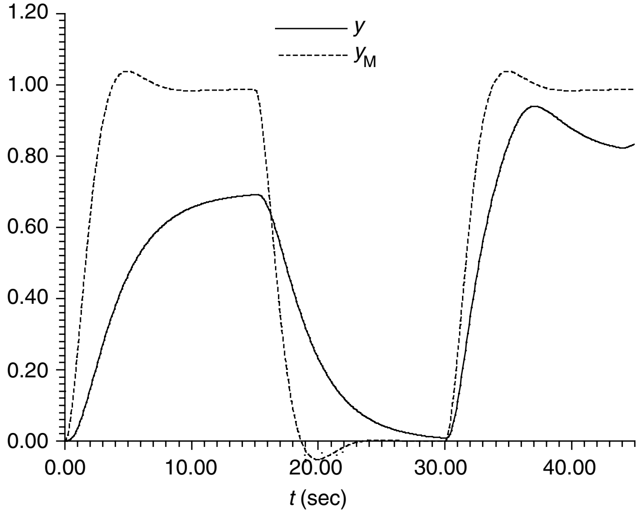


FIGURE 5.46 The system output and reference model responses at the beginning of learning.

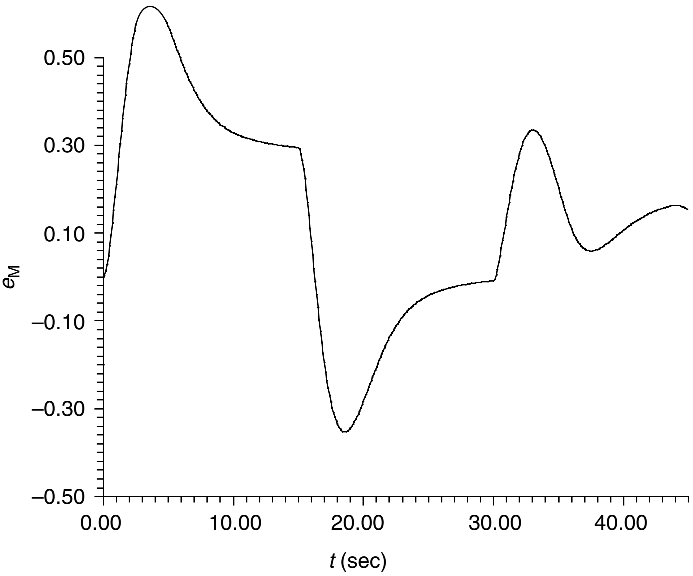


FIGURE 5.47 The tracking error responses at the beginning of learning ($k_3 = 0.7$).

Figure 5.48 show the tracking error and controller output responses, respectively. One can see that the maximum tracking error exceeds 60% in the first negative run, without reaching the steady-state. In the second learning iteration, the maximum tracking error is already reduced two times. Like with the nominal feedforward gain

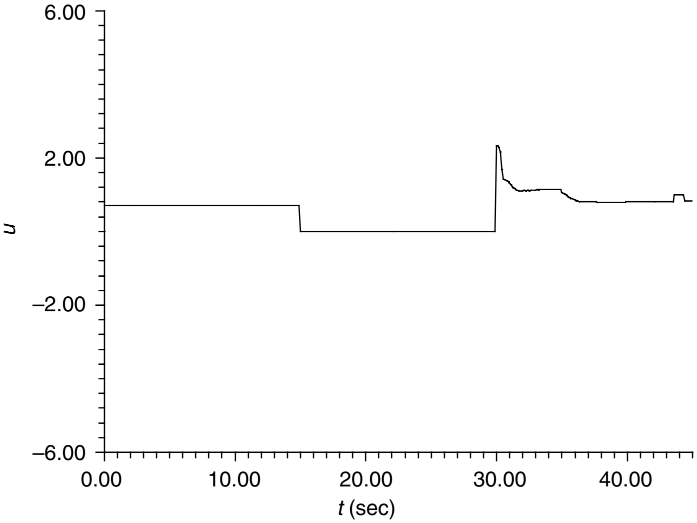


FIGURE 5.48 The self-learning fuzzy controller output responses at the beginning of learning.

coefficient value, output singletons were successfully tuned, keeping the maximal tracking error value below 5%. The zero steady-state tracking error value indicates also that the feedforward gain coefficient value is correct.

Figures 5.49 to 5.51 show the same group of responses obtained after twelve learning iterations. Likewise in the first experiment, the system follows the reference model very closely and the steady-state tracking error is kept below 5%. Different controller output waveforms in opposite transition directions demonstrate the nonlinear character of control. Figure 5.52 shows fast convergence of the feedforward gain coefficient estimation. The comparison of controller output responses (Figure 5.43 and Figure 5.51) indicates their similarity.

Figure 5.53 shows the transition of the learning stoppage criterion I_A^κ. One can see that regardless of a fairly incorrect initial value of the feedforward gain coefficient, criterion assumes the value which is below 10% in ten iterations (from the implementation point of view, we can assume that 10% is a reasonable threshold value where learning should stop).

There are only minor differences between the output singleton values obtained for the cases of nominal and estimated values of feedforward gain coefficient k_3. The final form of a fuzzy rule-table for the estimated value $k_3 = 0.7$ is shown in Table 5.5. Table areas filled with zeros are areas that stood out of reach of the system phase-trajectory. In those parts of the table the system is unstable. The fact that the trajectory of the fuzzy controlled system did not go through those unstable parts indicates the stability of the self-organization process itself. One can also notice a symmetrical form of the table, which was expected due to the linear character of the controlled process.

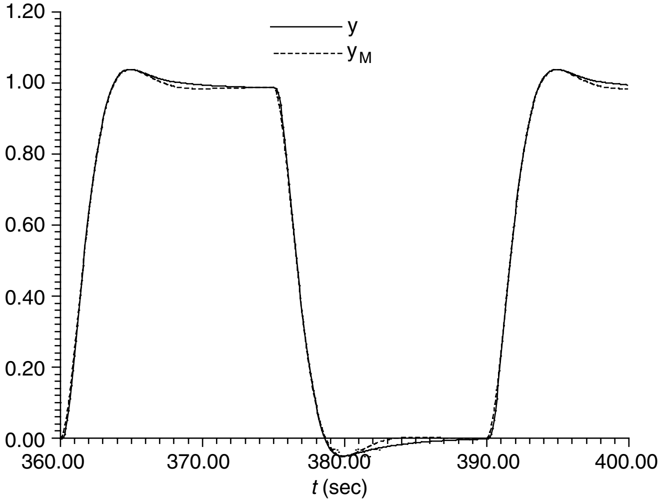


FIGURE 5.49 The system output and reference model responses at the end of learning ($k_3 = 0.7$).

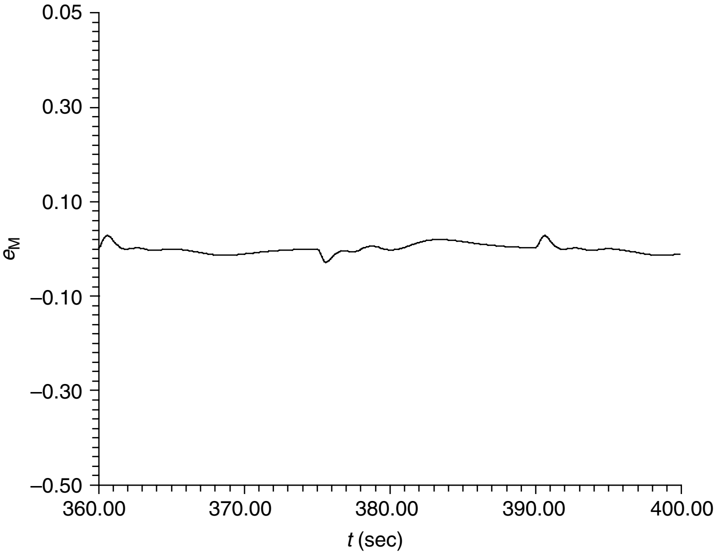


FIGURE 5.50 The tracking error responses at the end of learning ($k_3 = 0.7$).

The simulation results describing the performance of a self-organizing fuzzy controller in the case of controlling a nonlinear second-order process are given in Reference 59. Likewise in the linear case, the fuzzy controller reduced the tracking error below 5% of the imposed reference change. Due to the nonlinear

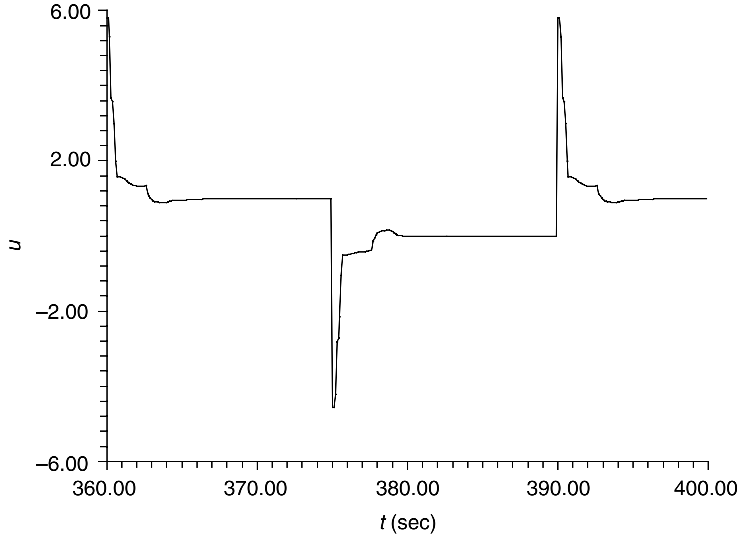


FIGURE 5.51 The self-learning fuzzy controller output responses at the end of learning.

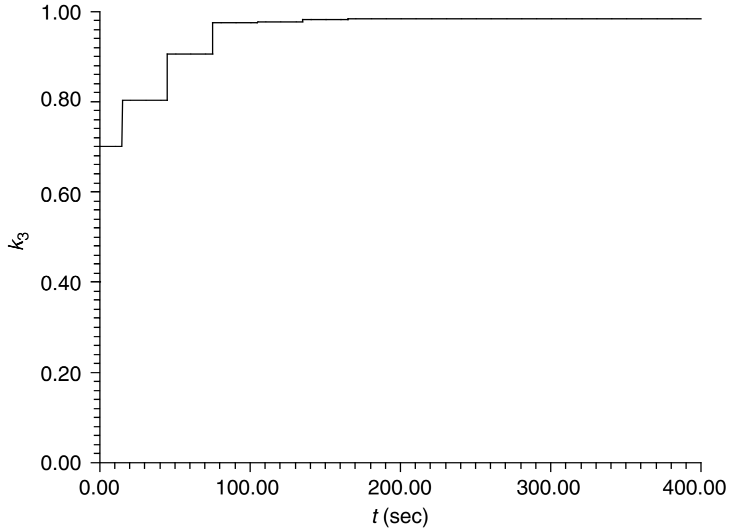


FIGURE 5.52 The transition of the feedforward gain coefficient ($k_3 = 0.7$).

character of the system, the process gain varied with the variation of operating point. Nevertheless, the self-learning procedure determined the value of feedforward gain coefficient k_3 very accurately, so that the steady-state tracking error was kept below 1%.

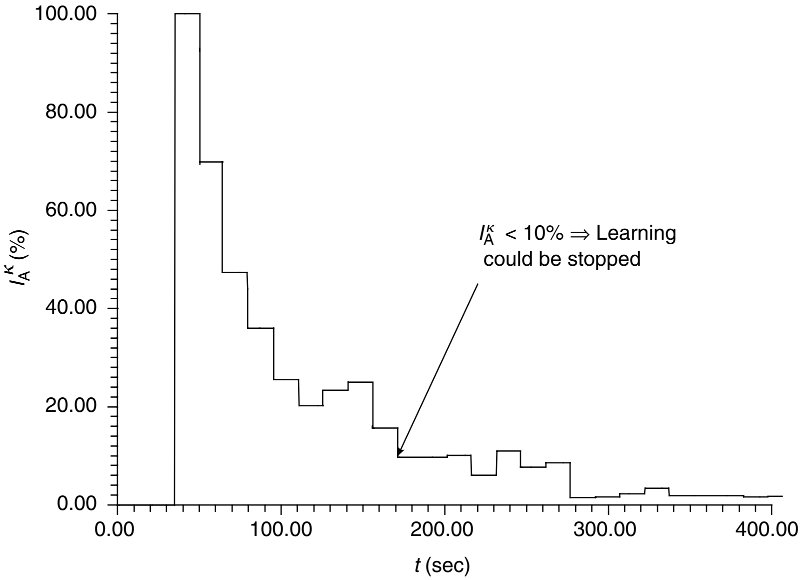


FIGURE 5.53 The transition of the learning stoppage criterion.

TABLE 5.5
The Fuzzy Rule-Table of the Self-Learning Fuzzy Controller (k_3 = 0.7)

	NLE	NSE	ZE	PSE	PLE
NLDYF	−0.321	−0.028	0	0	0
NSDYF	−4.302	0.096	0.075	0	0
ZDYF	−4.823	−0.178	0	0.780	4.966
PSDYF	0	0	−0.082	−0.320	4.122
PLDYF	0	0	0	0.146	0.301

Since the reference model is also used as a process approximation, the choice of model parameters has a large influence on the self-organization process. That influence in the case of controlling a third-order linear system (3.36) is shown in Figure 5.54.

A discrete form of the ITAE criterion was used as a model tracking quality criterion:

$$I_{te\mathrm{M}}^{\kappa} = \sum_{k=0}^{k_s} kT_{\mathrm{d}} |e_{\mathrm{M}}^{\kappa}(k)| \tag{5.94}$$

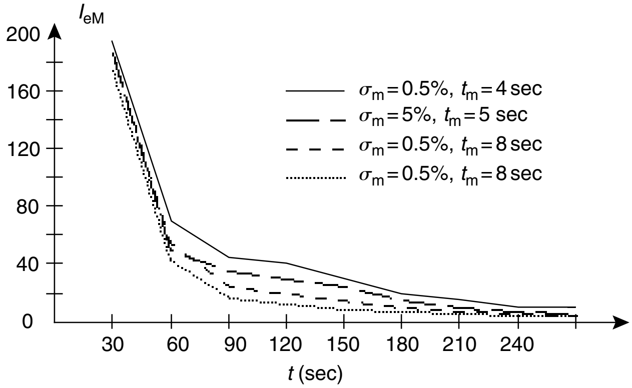


FIGURE 5.54 The influence of reference model parameters on self-organization.

where k_s is a number of sampling intervals during one learning iteration in the positive and negative direction.

The following combinations of reference model parameters were used: (1) $\sigma_m = 0.5\%$, $t_m = 4$ sec; (2) $\sigma_m = 5\%$, $t_m = 5$ sec; (3) $\sigma_m = 0.5\%$, $t_m = 6$ sec; (4) $\sigma_m = 0.5\%$, $t_m = 8$ sec. One can see from Figure 5.54 that the speed of self-organization, indicated by the criterion value, does not depend much on the parameters of the reference model. In all four cases, the criterion is reduced approximately to one-third of its initial value in the first learning iteration, and then it keeps decreasing with almost the same dynamics. However, from Figure 5.54 one can also see that self-organization is faster when models are slower.

The self-learning algorithm was derived under the assumption that the controlled process could be described well by its second-order reference model approximation. For the processes that we know better, other process approximations rather than a second-order reference model could be used, but experiments in some study cases showed that changes of self-organization dynamics due to other process approximations were practically negligible.

A fuzzy controller, whose parameters are synthesized by on-line self-organization, has a PD character, which means that it cannot eliminate a steady-state error in static control systems. Instead, a feedforward element takes care of that thanks to the on-line adjustment of the gain coefficient k_3. However, when disturbance is present, regardless of the presence of the feedforward element, a static error will occur. The solution is to add an integral element parallel to the fuzzy controller [60]. In order to avoid the undesired need for external adjustment of the integral gain coefficient, the self-learning procedure for this purpose is described in Section 5.3.4.

After simulation experiments, the self-learning fuzzy controller was tested experimentally on the laboratory process simulator Feedback PCS 327 (see Figure 3.12). The reference model, sampling interval and all other parameters in the experiment are equal to those from simulations. Experiments were conducted

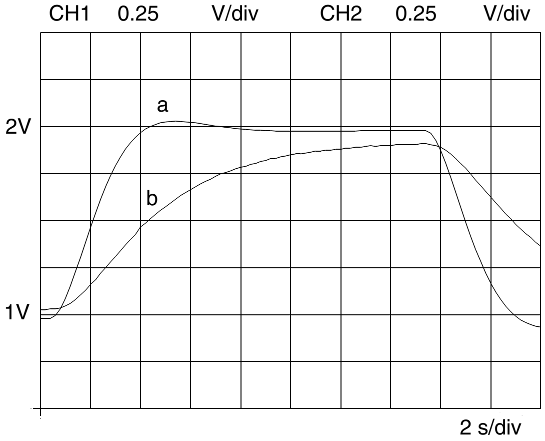


FIGURE 5.55 The system output and reference model responses at the beginning of learning.

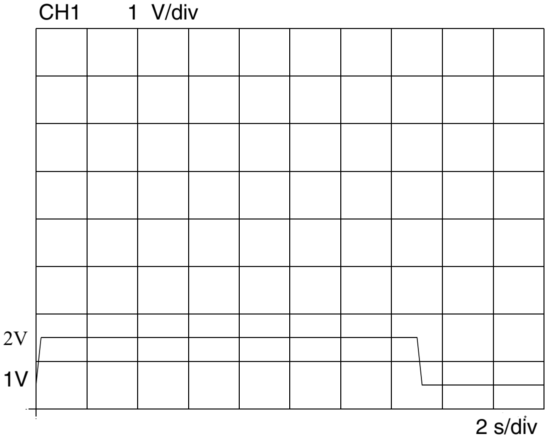


FIGURE 5.56 The self-learning fuzzy controller output responses at the beginning of learning.

in the operating point $y_{f0} = 1$ V for stepwise changes of the reference input $\Delta u_{\mathrm{r}} = \pm 1$ V.

Figure 5.55 and Figure 5.56 show the results obtained in the beginning of learning with a nominal feedforward gain coefficient value (reciprocal to the known value of process gain).

The fuzzy rule-table is initially filled with zeros so only the feedforward element is acting on the process. This causes a very large tracking error at the beginning of learning. The results obtained at the end of learning are shown in

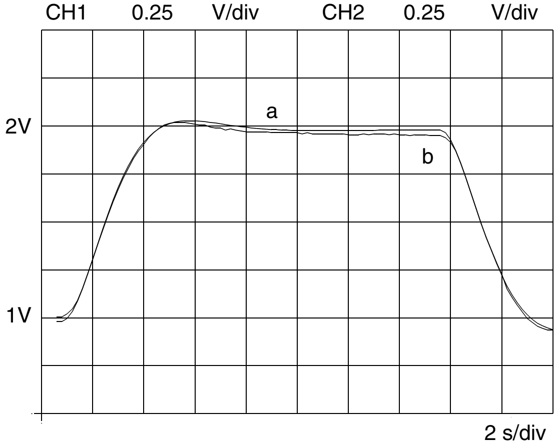


FIGURE 5.57 The system output and reference model responses at the end of learning.

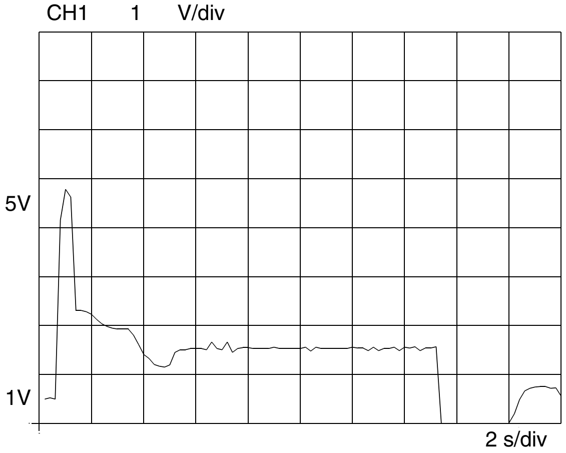


FIGURE 5.58 The self-learning fuzzy controller output responses at the end of learning.

Figure 5.57 and Figure 5.58. From Figure 5.57 we can see that the model tracking error is practically negligible. The form of the control input signal shown in Figure 5.58 is very similar to the one obtained in simulations (see Figure 5.43).

The changes of the nominal (known in advance) feedforward gain coefficient $k_3 = 1$ are shown in Figure 5.59. It can be noticed that the value of k_3 is slightly varying (1 to 2%) due to the approximate character of the learning algorithm and the presence of measurement noise. The transition of the learning stoppage criterion (5.92) is shown in Figure 5.60. By comparison of the criterion shown in Figure 5.60 to the criterion shown in Figure 5.45, one can see that the speed of

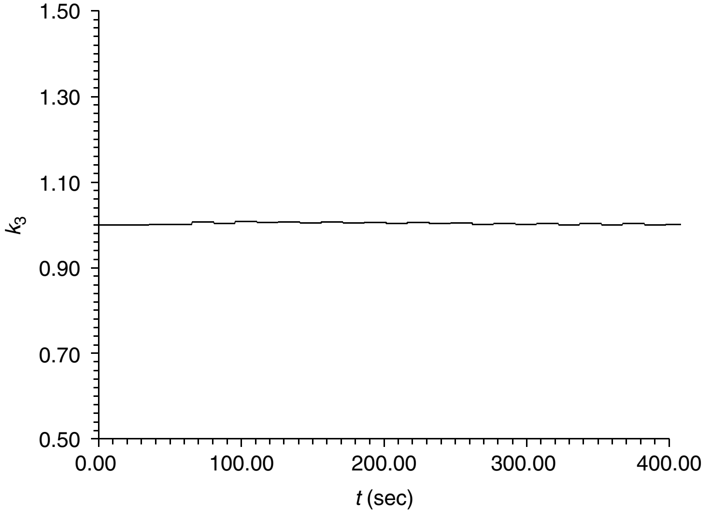


FIGURE 5.59 The transition of the feedforward gain coefficient.

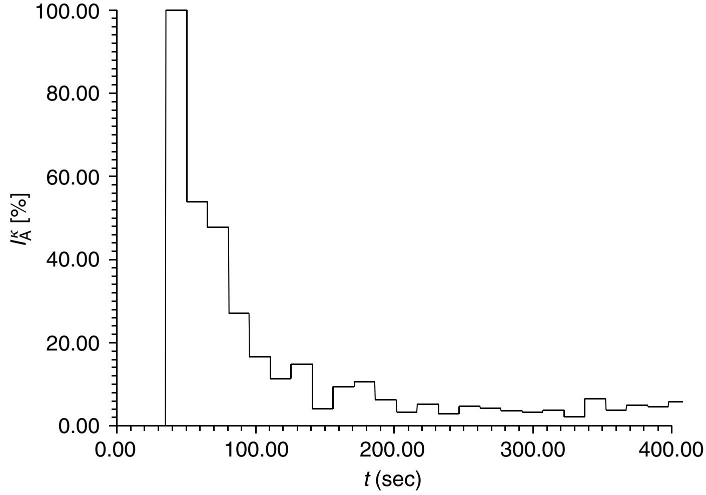


FIGURE 5.60 The transition of the learning stoppage criterion.

fuzzy controller tuning in the simulation and in the experiment is almost identical. After 6 to 7 iterations the criterion satisfies the condition to stop learning.

The system output and reference model responses obtained at the beginning of learning with a supposed feedforward gain coefficient value $k_3 = 0.7$ (reciprocal to the supposed value of the process gain) are shown in Figure 5.61. A pretty large

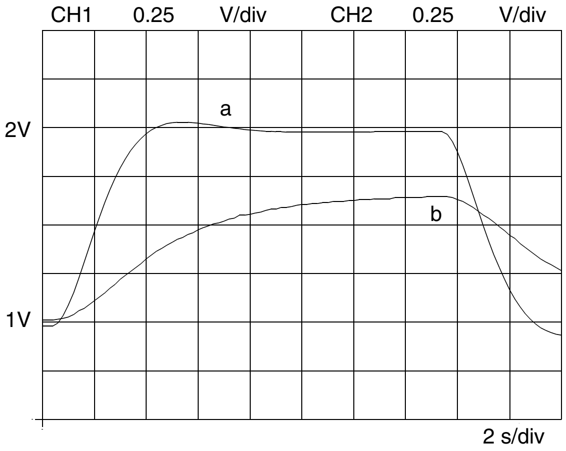


FIGURE 5.61 The system output and reference model responses at the beginning of learning ($k_3 = 0.7$).

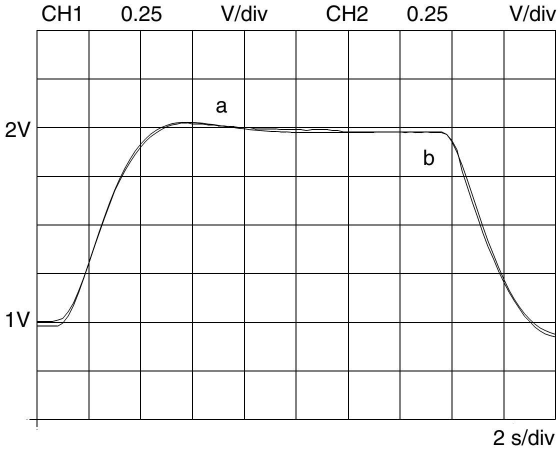


FIGURE 5.62 The system output and reference model responses at the end of learning ($k_3 = 0.7$).

steady-state error can be seen in the first learning iteration due to an inaccurate estimation of k_3.

Figure 5.62 shows the system output and reference model responses obtained at the end of learning, while Figure 5.63 shows the responses of the self-learning fuzzy controller output.

The feedforward gain coefficient assumes its final value in a little bit slower manner than in simulations, as shown in Figure 5.64.

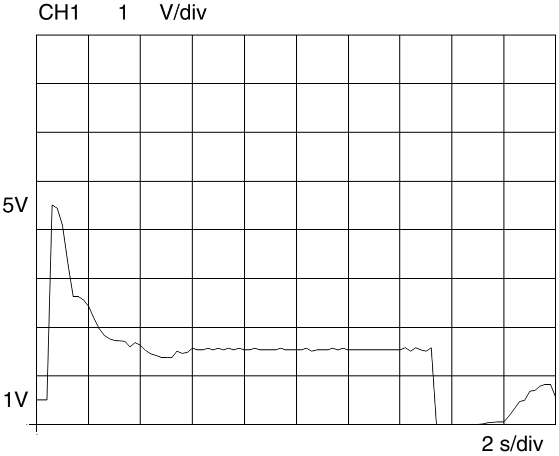


FIGURE 5.63 The self-learning fuzzy controller responses at the end of learning ($k_3 = 0.7$).

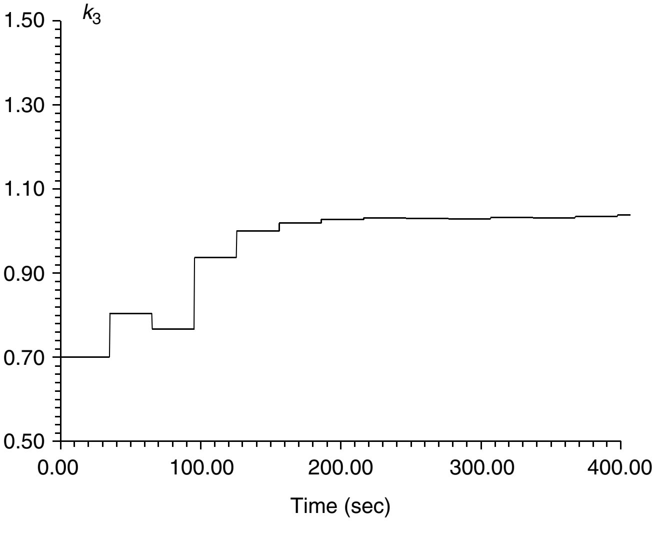


FIGURE 5.64 The transition of the feedforward gain coefficient.

The comparison of the learning stoppage criterion transition shown in Figure 5.65 with previously obtained criterion transitions (Figures 5.45, 5.53, and 5.60) indicates their large similarity and match-up of simulation and experimental results.

Example 5.4 Sensitivity model-based self-learning fuzzy control of a positioning servo system — nonlinear case I.

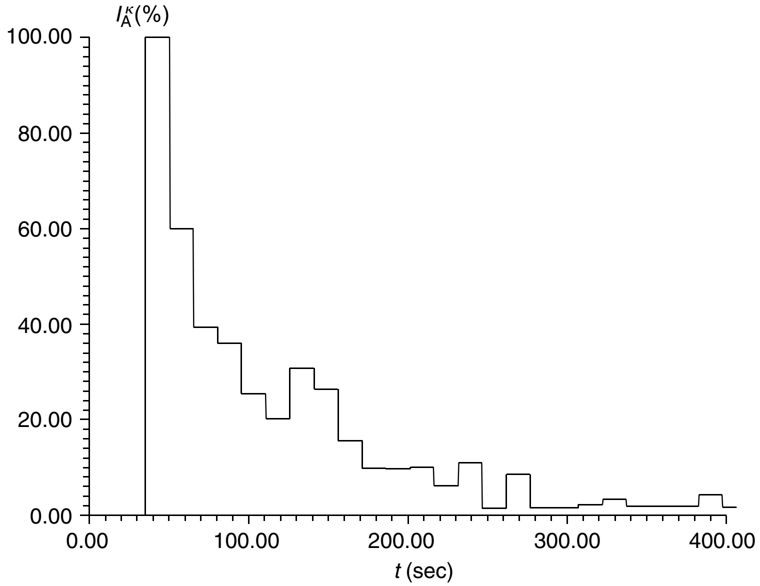


FIGURE 5.65 The transition of the learning stoppage criterion.

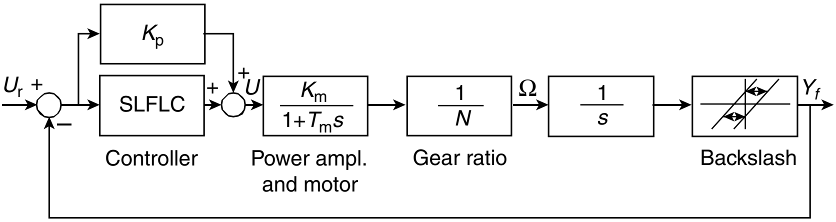


FIGURE 5.66 Fuzzy controlled DC servo motor drive.

Let us describe the design, simulation, and experimental verification of a sensitivity model-based self-learning fuzzy logic controller that was tested in the position control loop of a laboratory DC servo motor drive ES 130 (Feedback Inc.) shown in Figure 5.66. The rated parameter values are: $k_p = 0.5$ V/V (proportional controller gain), $K_d = 2.67$ V/rad (position feedback gain), $K_m =$ 175 rad/Vsec (amplifier and motor gain), $T_m = 160$ msec (electromechanical time constant), $N = 16$ (gear ratio). The drive is affected by a strong impact of backlash nonlinearity (Figure 5.67). The backlash width is set to $\Delta\theta = \pm 3°$ (0.0523 rad).

As in the position control scheme here we deal with control of an astatic process, feedforward and integral control elements responsible for steady-state accuracy in static control systems (see the controller structure shown in Figure 5.18) are not

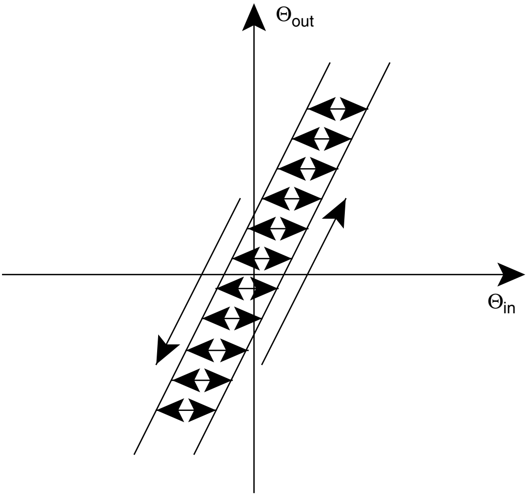


FIGURE 5.67 A backlash nonlinearity.

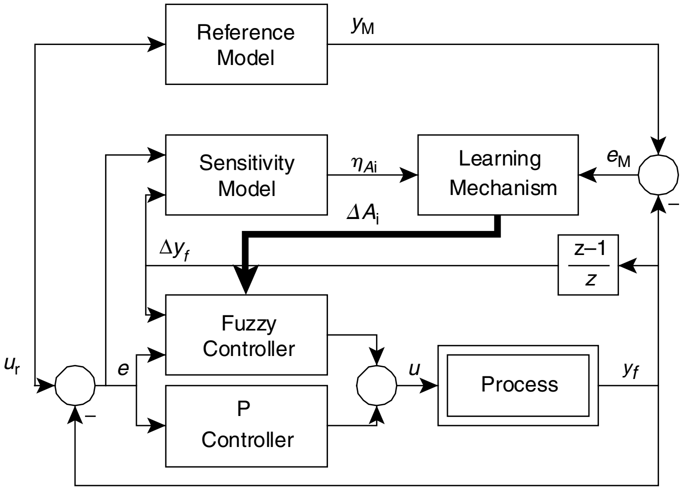


FIGURE 5.68 The block scheme of a self-learning hybrid fuzzy system.

needed. Since originally the positioning servo system is controlled by a P controller, in order to improve the control characteristics, let us just add a self-learning fuzzy controller in parallel to the P controller. In this way, the structure becomes a hybrid one (see Figure 5.68).

Desired dynamic characteristics of the closed-loop position control system are described by a second-order reference model, whose parameters are determined according to selected performance indices: the response overshoot, $\sigma_m = 1.5\%$

and the peak time $t_{\mathrm{m}} = 0.5$ sec, which yields (sampling interval $T_{\mathrm{d}} = 0.01$ sec):

$$\begin{aligned} y_{\mathrm{M}}(k) = 1.8429 y_{\mathrm{M}}(k-1) - 0.8521 y_{\mathrm{M}}(k-2) \\ + 0.0047 u_{\mathrm{r}}(k-1) + 0.0045 u_{\mathrm{r}}(k-2) \end{aligned} \tag{5.95}$$

In accordance with the described design procedure, reference model (5.95) also represents process approximation G_{pa} used for calculation of sensitivity functions (see relations [5.82] and [5.83]). The minimal threshold value for the sensitivity functions is set to be $\beta_{\min} = 0.1$.

Five linguistic subsets have been defined for both fuzzy controller inputs (universes of discourse E and DYF): NL, NM, Z, PM, and PL. Accordingly, the fuzzy rule-table has 25 control rules. Input membership functions have a Gaussian form (5.74) and an α-cut ($\alpha = 0.05$) is applied to them (see relation [2.26]). The input fuzzy sets are scaled into interval $[-1, 1]$. Scaling coefficient values are $K_{\mathrm{e}} = 1$ and $K_{\Delta y_{\mathrm{f}}} = 34$. The centers and widths of input membership functions are distributed in the same way as in the previous example:

$$\begin{aligned}
&c_{\mathrm{NL}}^{e} = c_{\mathrm{NL}}^{\Delta y_{\mathrm{f}}} = -1 \\
&c_{\mathrm{NM}}^{e} = c_{\mathrm{NM}}^{\Delta y_{\mathrm{f}}} = -0.4 \\
&c_{\mathrm{Z}}^{e} = c_{\mathrm{Z}}^{\Delta y_{\mathrm{f}}} = 0 \\
&c_{\mathrm{PM}}^{e} = c_{\mathrm{PM}}^{\Delta y_{\mathrm{f}}} = 0.4 \\
&c_{\mathrm{PL}}^{e} = c_{\mathrm{PL}}^{\Delta y_{\mathrm{f}}} = 1 \\
&w_{\mathrm{NL}}^{e} = w_{\mathrm{NL}}^{\Delta y_{\mathrm{f}}} = w_{\mathrm{PL}}^{e} = w_{\mathrm{PL}}^{\Delta y_{\mathrm{f}}} = 0.227 \\
&w_{\mathrm{NM}}^{e} = w_{\mathrm{NM}}^{\Delta y_{\mathrm{f}}} = w_{\mathrm{PM}}^{e} = w_{\mathrm{PM}}^{\Delta y_{\mathrm{f}}} = 0.048 \\
&w_{\mathrm{Z}}^{e} = w_{\mathrm{Z}}^{\Delta y_{\mathrm{f}}} = 0.01
\end{aligned}$$

The performance of the self-learning fuzzy controller has been tested in the case of a series of reference input step changes equal to $\Delta u_{\mathrm{r}} = \pm 0.5$ V (corresponding to $\Delta\theta_{\mathrm{r}} = \pm 10°$). Figure 5.69 shows the reference model and system output responses, while Figure 5.70 and Figure 5.71 show the respective tracking error and controller output responses obtained after first three runs of the system. The backlash effect is clearly seen in the delayed start and trimmed peak values of the system response. In the first run, due to the empty fuzzy rule-table the dynamic behavior is determined only by the proportional controller. The tracking error exceeds 30% of the reference input change. Because of backlash the controller output is different in different control directions. Improvements in system dynamics and steady-state behavior can be noticed already in the second learning iteration.

Figures 5.72 to 5.74 show the same group of responses obtained after self-organization was completed (after twelve learning iterations). The system closely

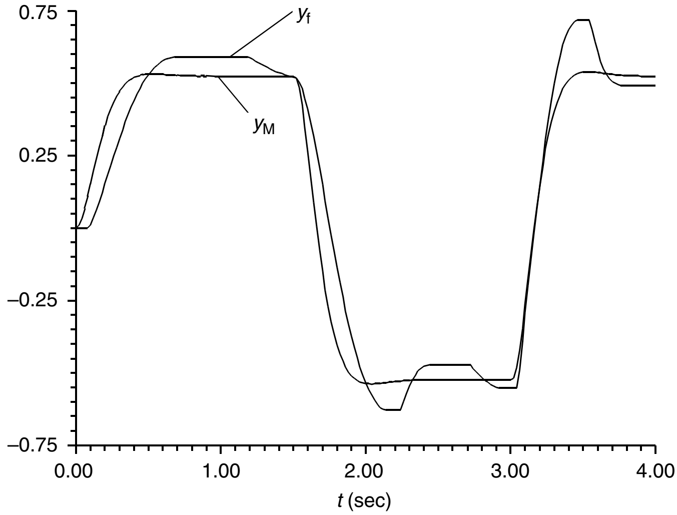


FIGURE 5.69 The system output and reference model responses at the beginning of learning. (From Kovačić, Z., Bogdan, S., and Balenović, M., *The IEEE Transac. Energy Conver.*, 13(4), 1479–1484, 1999. With permission.)

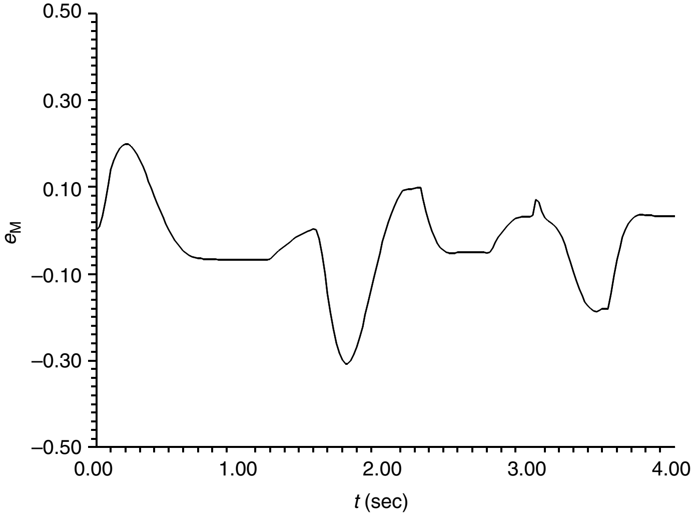


FIGURE 5.70 The model tracking error responses at the beginning of learning. (From Kovačić, Z., Bogdan, S., and Balenović, M., *The IEEE Transac. Energy Conver.*, 13(4), 1479–1484, 1999. With permission.)

follows the reference model and the tracking error never exceeds 5% of the reference input change. The controller output (Figure 5.74) has a very acceptable form with a stronger effort in the initial phase of response. The steady-state control error is kept within ±2%, which is a significant achievement, since the ±3° backlash

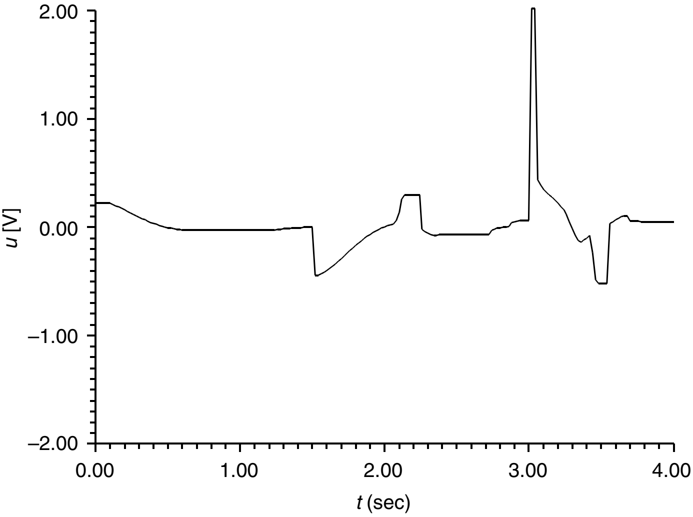


FIGURE 5.71 The self-learning fuzzy controller output responses at the beginning of learning. (From Kovačić, Z., Bogdan, S., and Balenović, M., *The IEEE Transac. Energy Conver.*, 13(4), 1479–1484, 1999. With permission.)

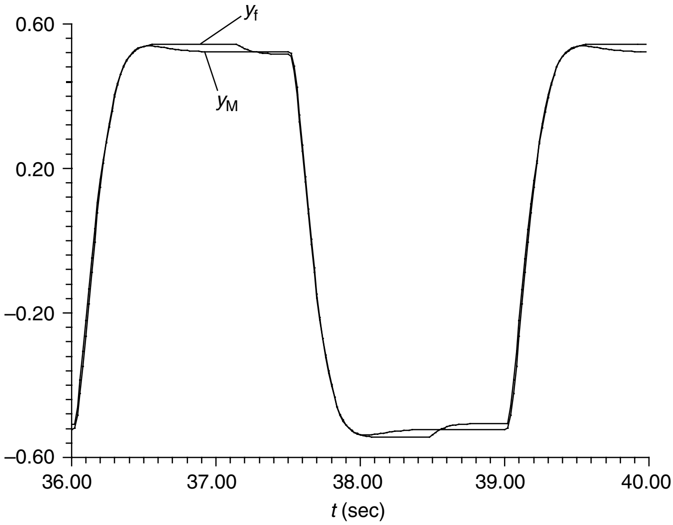


FIGURE 5.72 The system output and reference model responses at the end of learning. (From Kovačić, Z., Bogdan, S., and Balenović, M., *The IEEE Transac. Energy Conver.*, 13(4), 1479–1484, 1999. With permission.)

dead-zone would normally cause a static error within ±15%. The final singleton values of the self-organized fuzzy rule-table are shown in Table 5.6.

Let us present the performance of the self-learning fuzzy controller tested by a series of experiments on the laboratory servo system Feedback ES 130.

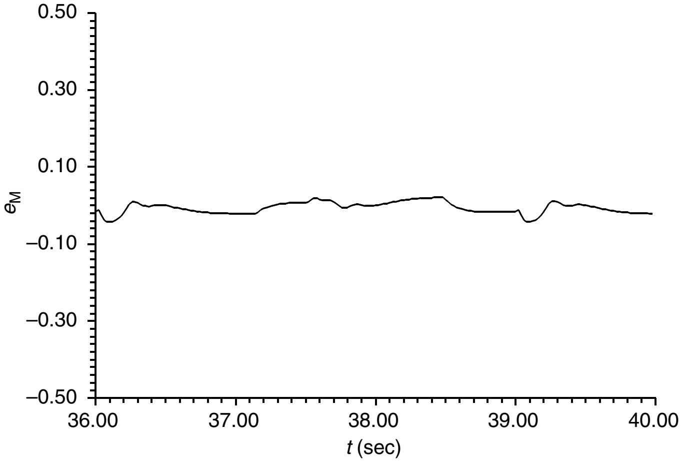


FIGURE 5.73 The model tracking error responses at the end of learning. (From Kovačić, Z., Bogdan, S., and Balenović, M., *The IEEE Transac. Energy Conver.*, 13(4), 1479–1484, 1999. With permission.)

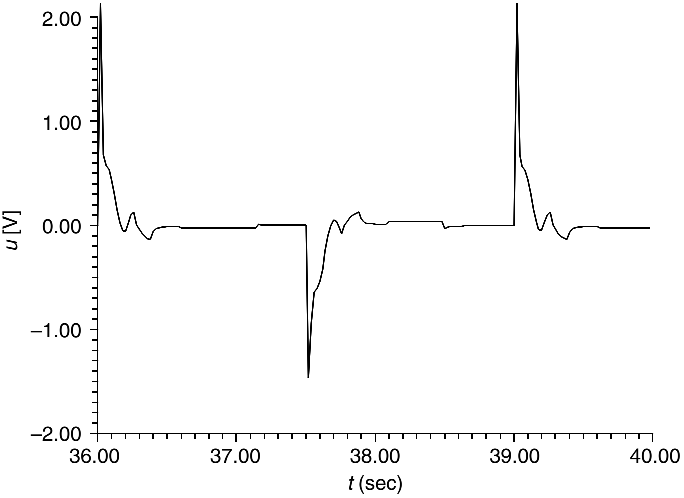


FIGURE 5.74 The self-learning fuzzy controller output responses at the end of learning. (From Kovačić, Z., Bogdan, S., and Balenović, M., *The IEEE Transac. Energy Conver.*, 13(4), 1479–1484, 1999. With permission.)

We use the same program (the same structure of the fuzzy controller and the same reference model) as in simulations. A position feedback signal is obtained by sampling the servo potentiometer voltage signal fed to the input of a 12-bit bipolar A/D–D/A card (input range ±9 V, $T_d = 10$ msec). Due to the measurement noise, the feedback signal is filtered by computing the average of last two acquired values.

TABLE 5.6
The Self-Organized Fuzzy Rule-Table

	NLE	NME	ZE	PME	PLE
NLDYF	−0.033	0.125	0	0	0
NMDYF	−0.485	0.117	0.107	0	0
ZDYF	−1.491	0	0	0.660	1.766
PMDYF	0	0	−0.183	0.022	1.955
PLDYF	0	0	0	−0.286	0.172

The series of reference input changes is generated by a computer program. Since the ±9 V analog input range is converted into the 12-bit digital input range (±2048), with feedback gain coefficient $K_\mathrm{d} = 2.67$ V/rad, the change of ±300 LSB corresponds to the angular displacement of about ±28°. As we use the same fuzzy controller program as in simulations (where inputs are scaled to interval $[-1, 1]$), fuzzy controller input variables $e(k)$ and $\Delta y_\mathrm{f}(k)$ are scaled with respective scaling factor values $K_\mathrm{e} = 0.00167$ and $K_{\Delta y_\mathrm{f}} = 0.05667$. The backlash value identified in the servo system ES 130 was approximately 5%.

The reference model and the system output responses obtained after first three runs of the system are shown in Figure 5.75. The time scale starts after three seconds as in that moment the first change of the reference input occurred. We can see that the system is faster than the reference model. Also, we can notice a rather high peak value (overshoot) and a large steady-state error. In the second learning iteration, the fuzzy controller already adjusted the rise time dynamics, reduced the overshoot, and significantly decreased the steady-state error. The tracking error exceeds 20° which is more than 30% of the reference input change (Figure 5.6). In the first learning iteration the controller output value assumes nonzero values, but the static friction helps to keep the working mechanism still. In the next iteration the fuzzy controller already starts contributing to overall stronger control effort (Figure 5.76).

Figure 5.78 and Figure 5.79 show the system output, reference model output and tracking error responses obtained after five learning iterations. The system closely follows the reference model and the maximal tracking error is kept within ±3° (±10% of the reference input change). The steady-state tracking error value is very small and comparable to the value of measurement noise. The controller output, shown in Figure 5.80, is very similar to the controller output obtained in simulations (Figure 5.74).

From the results we can see that the sensitivity model-based self-organization effectively dealt with backlash nonlinearity and after only 6 to 8 learning iterations the created fuzzy control surface was good enough to satisfy initial design goals.

The controller synthesis does not require knowledge about the process model and the type of system nonlinearity. It is sufficient to know an approximate value of process gain and boundaries of system variables to prevent the self-organization

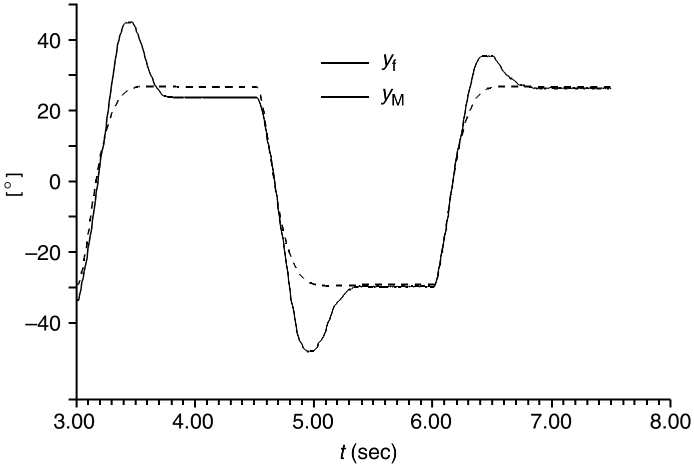


FIGURE 5.75 The system output and reference model responses at the beginning of learning.

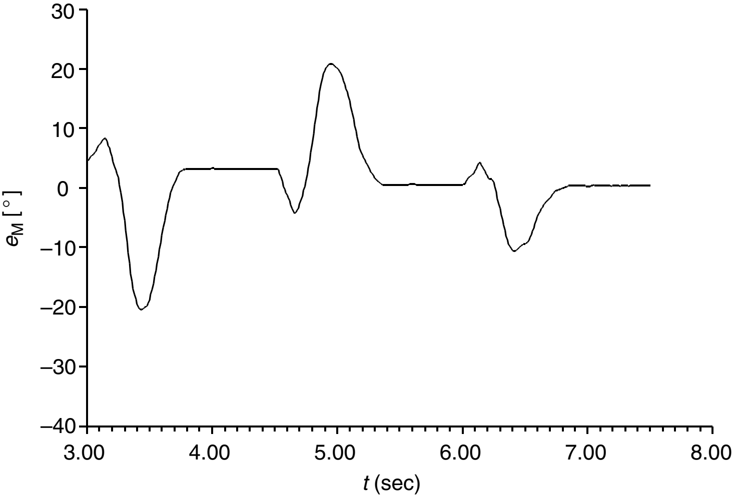


FIGURE 5.76 The model tracking error responses at the beginning of learning.

algorithm to lead to system instability. The final singleton values of the self-organized fuzzy rule-table are shown in Table 5.7.

Example 5.5 Sensitivity model-based self-learning fuzzy control of a positioning servo system — nonlinear case II.

Let us present the performance of a sensitivity model-based self-learning hybrid fuzzy controller controlling the positioning servo system described in detail

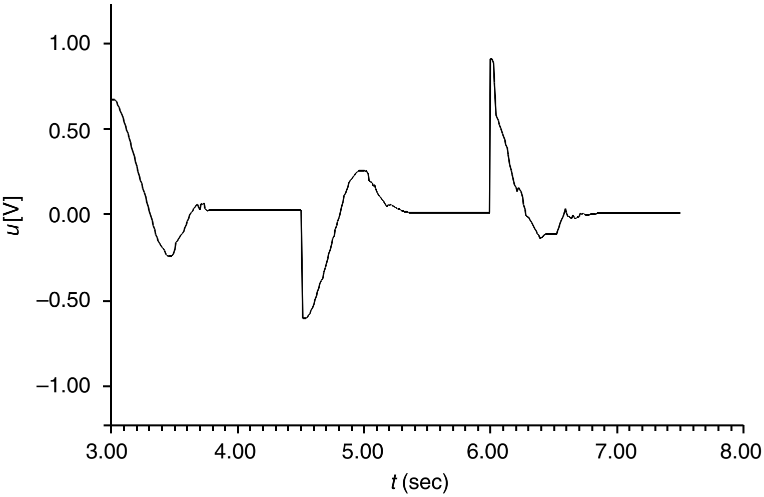


FIGURE 5.77 The self-learning fuzzy controller output responses at the beginning of learning.

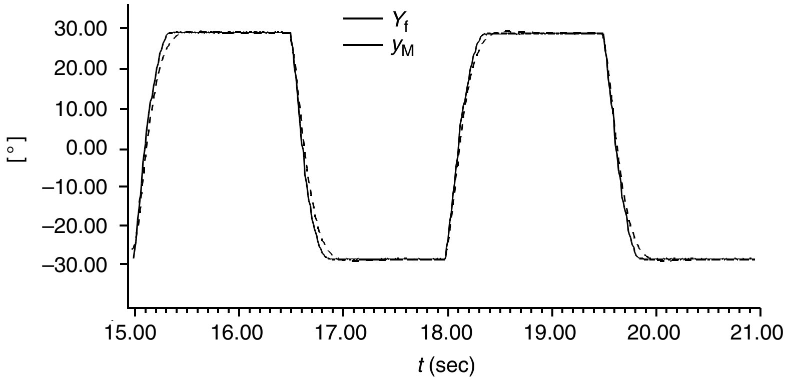


FIGURE 5.78 The system output and reference model responses at the end of learning.

in Example 5.1. The block structure of the nonlinear control process shown in Figure 5.81 represents a position control loop affected by nonlinear friction and a gravitation-dependent shaft load (see Figure 5.3 and Figure 5.4). We already said that from the control point of view these nonlinearities (and backlash as well) are generally assumed as difficult to deal with. A weight attached to the bar represents a full load weight while the bar itself represents a reduced load weight.

Referring to Figure 5.3, the nonlinear gravitation-dependent load (5.24) can be described as a function of angular displacement measured from the natural

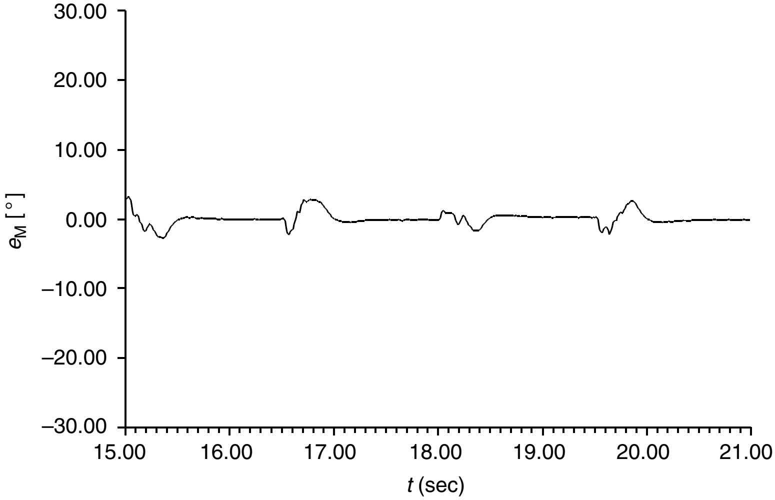


FIGURE 5.79 The model tracking error responses at the end of learning.

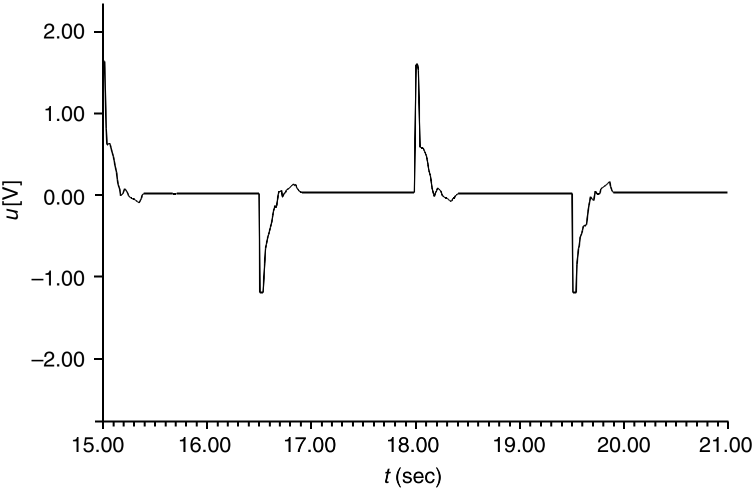


FIGURE 5.80 The self-learning fuzzy controller output responses at the end of learning.

equilibrium state ($\theta = y_f + \pi/2 = 0°$):

$$T_l = f_l(\theta) = T_{l0} \sin\left(\theta \frac{\pi}{180}\right) \tag{5.96}$$

where θ is the angle expressed in degrees.

TABLE 5.7
The Self-Organized Fuzzy Rule-Table

	NLE	NME	ZE	PME	PLE
NLDYF	0.183	0.19	0	0	0
NMDYF	0.145	0.073	0.07	0	0
ZDYF	−0.37	−0.13	0	0.16	0.96
PMDYF	0	−0.06	−0.008	−0.008	−0.132
PLDYF	0	0	−0.201	−0.286	−0.272

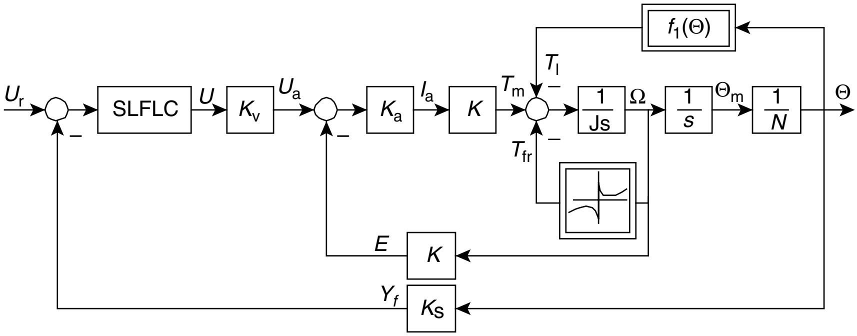


FIGURE 5.81 The structure of a nonlinear position control loop. (From Kovačić, Z., Balenović, M., and Bogdan, S., *IEEE Contr. Syst. Mag.*, 18(3), 41–51, 1998. With permission.)

In the synthesis of a self-learning fuzzy controller friction parameters (static, viscous, and dynamic) are treated as unknown. Although we could easily identify friction parameters in the controlled system, this is not necessary.

The rated parameter values are: $K_p = 2$ V/V (P gain), $K_s = 0.318$ V/rad (feedback gain), $K = 0.191$ Nm/A, $K_a = 0.106$ V/A (armature gain), $J_T = 2.7E^{-4}$ kg m^2 (reduced load weight), $J_T = 1.27E^{-3}$ kg m^2 (full load weight), $N = 4$ (gear ratio). The distance of the load mass center (full load weight) is $l = 26.5$ cm, and mass of the weight is $m = 0.22$ kg.

From two extreme moment of inertia values related to reduced and full load weights, we can see that parameter J_T varies approximately in the range 1:5. Referring to the block diagram of the control process shown in Figure 5.5, it should be noted that because of implemented hardware technology it was not possible to determine exact values of some process parameters (e.g., chopper gain K_c, gain coefficient K_v dependent on the internal angular speed controller gain, current feedback gain K_i). This does not represent a problem for the synthesis of a self-learning fuzzy controller, since the exact model of the process is not needed.

A personal computer with a 12-bit A/D and D/A converter board has been used for implementation of the self-learning fuzzy controller. The structure of a fuzzy controller is the same as in the design example 5.2, that is, five linguistic subsets are defined for both fuzzy controller inputs (universes of discourse E and DYF): NL, NM, Z, PM, PL. A linear distribution of corresponding membership functions (5.74) has been selected. The sampling interval is $T_\mathrm{d} = 10$ msec. Parameters of the second-order reference model are determined according to the selected performance indices: overshoot in response, $\sigma_\mathrm{m} = 0.5\%$ and peak times: $t_\mathrm{m} = 0.6$ sec and $t_\mathrm{m} = 0.4$ sec. The reference model is simultaneously used as a process approximation G_pa in the sensitivity model (5.83).

The self-learning fuzzy control method has been experimentally tested for a series of step changes of the reference input equal to $\Delta\theta_\mathrm{r} = \pm 20°$ in two intentionally selected operating points, $\theta_0 = 0°$ (the bar is in the bottom vertical position) and $\theta_0 = -90°$ (the bar is in the left-hand horizontal position), which correspond to extreme magnitudes of the position dependent load torque $T_\mathrm{l} = f_1(\theta) = T_{10} \sin\theta$. For the given 12-bit A/D and D/A converters changes of the reference input equal to $\Delta\theta_\mathrm{r} = \pm 20°$ correspond with digital values of ±100 LSB. Scaling coefficients are $K_\mathrm{e} = 0.005$ and $K_{\Delta y_\mathrm{f}} = 0.0833$.

Figure 5.82 shows the reference model ($t_\mathrm{m} = 0.6$ sec) and the measured position responses obtained after two runs of the system (one in each direction) under full load weight ($T_{10} = \max$) conditions in the operating point $\theta_0 = 0°$. Effects of friction and nonlinear time-varying load reflect in different dynamics for each direction and in presence of a steady-state error. Figure 5.83 and Figure 5.84 show the tracking error and controller output responses. In the first run, the dynamic behavior is determined only by the proportional controller (see the hybrid fuzzy structure shown in Figure 5.68) and the tracking error exceeds 30% of the imposed change of reference input.

Figures 5.85 to 5.87 show the same group of responses obtained after completion of learning (after twelve positive runs of the system). The closed-loop control system follows the reference model very closely and the maximum tracking error value is kept below 5%. From the practical point of view, the controller output has a very acceptable nonoscillatory form and also, the steady-state system error is kept at zero as required.

The singleton values obtained after completion of the self-learning process are shown in Table 5.8. Singleton values are expressed in volts.

It should be noticed that the sign of gravity-dependent load T_l (see [5.96]) depends on the direction of bar movement. Moreover, the static friction value depends on the current bar position (operating point). Since the fuzzy controller does not have an integral character, the role of singleton A_{33}, which contributes the most to the controller output in moments when input variables belong to their zero fuzzy sets (see Table 5.8), is to compensate the steady-state error. The result is that singleton A_{33} attains the sign and value, which depend on the current operating point and the sign of reference input.

So in the case of a negative change of the reference input, $A_{33} = -0.28$, while for a positive change, $A_{33} = 0.48$. The difference between two values that appear in Table 5.8 is caused by static friction whose influence is dominant in the operating point $\theta_0 = 0°$. Different signs of the above two values indicate that A_{33} compensates the gravitational load, too.

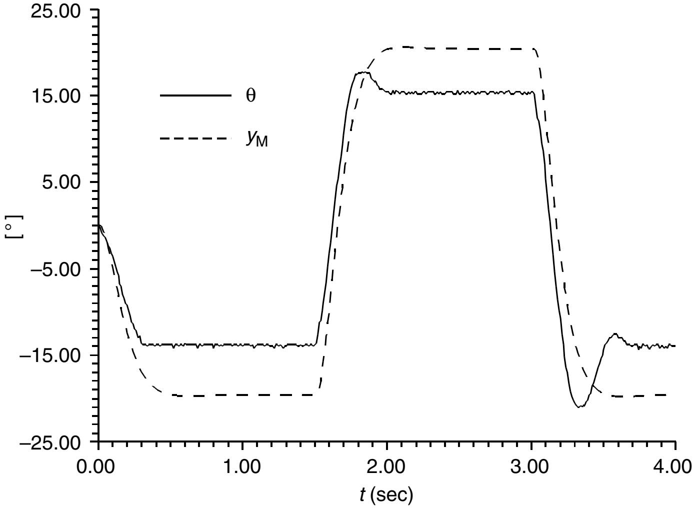


FIGURE 5.82 Reference model and system output; start of learning; full load weight; $\theta_0 = 0°$. (From Kovačić, Z., Balenović, M., and Bogdan, S., *IEEE Contr. Syst. Mag.*, 18(3), 41–51, 1998. With permission.)

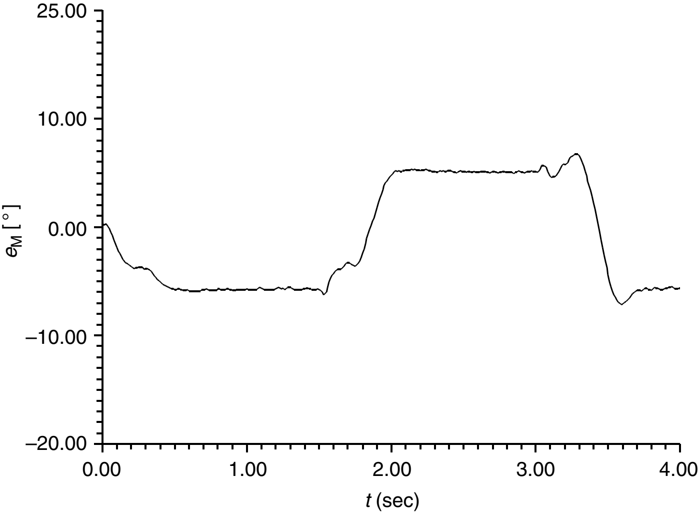


FIGURE 5.83 Tracking error; start of learning; full load weight; $\theta_0 = 0°$. (From Kovačić, Z., Balenović, M., and Bogdan, S., *IEEE Contr. Syst. Mag.*, 18(3), 41–51, 1998. With permission.)

In order to check the robustness of the self-learning process, a new experiment is made in the same operating point $\theta_0 = 0°$, but under reduced load weight ($T_{L0} = \text{min}$) conditions. Figure 5.88 shows the reference model and system output responses, while Figure 5.89 shows the tracking error response obtained after first

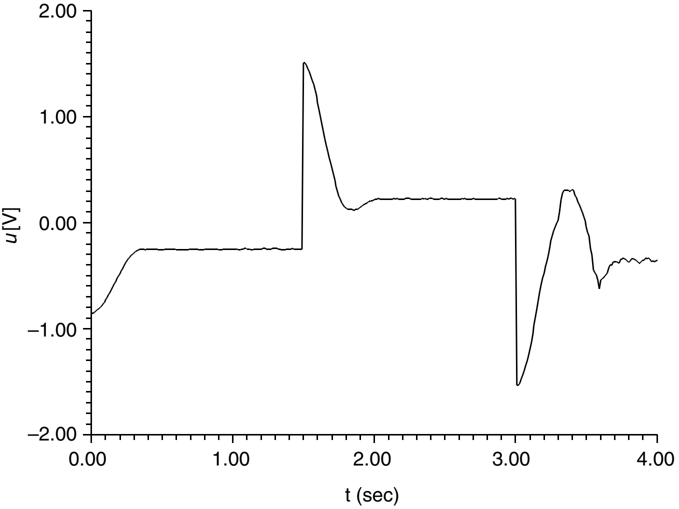


FIGURE 5.84 Controller output; start of learning; full load weight; $\theta_0 = 0°$. (From Kovačić, Z., Balenović, M., and Bogdan, S., *IEEE Contr. Syst. Mag.*, 18(3), 41–51, 1998. With permission.)

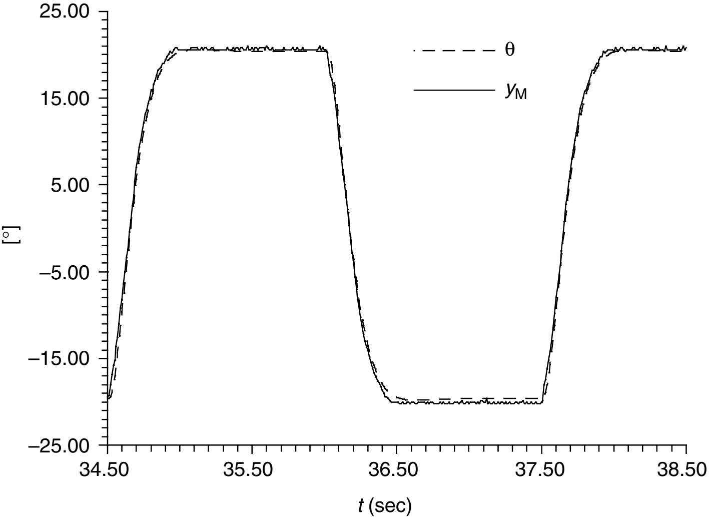


FIGURE 5.85 Reference model and system output; end of learning; full load weight; $\theta_0 = 0°$. (From Kovačić, Z., Balenović, M., and Bogdan, S., *IEEE Contr. Syst. Mag.*, 18(3), 41–51, 1998. With permission.)

two runs of the system. The system dynamics are now faster that the reference model dynamics. The steady state error (which is now lower because of reduced bar weight) and varying dynamics exhibited in different movement directions, caused by the friction and the gravitational load, are clearly seen, and in the initial

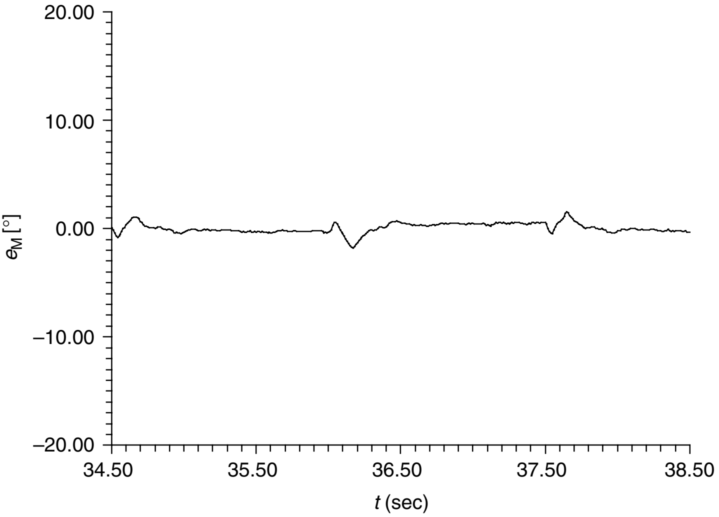


FIGURE 5.86 Tracking error; end of learning; full load weight; $\theta_0 = 0°$. (From Kovačić, Z., Balenović, M., and Bogdan, S., *IEEE Contr. Syst. Mag.*, 18(3), 41–51, 1998. With permission.)

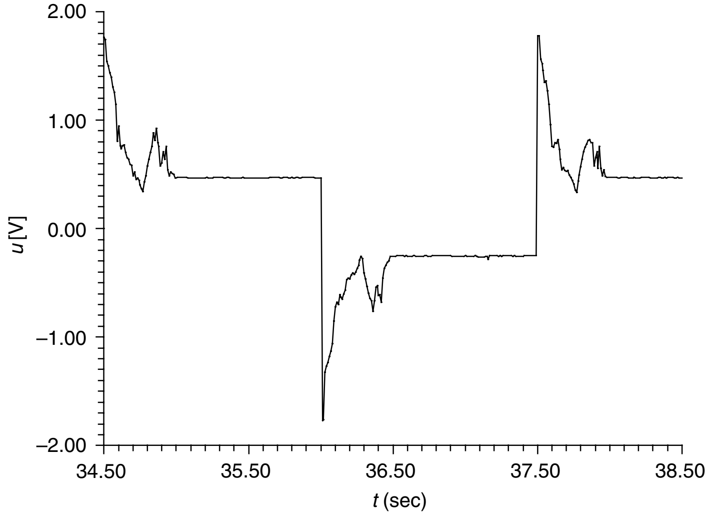


FIGURE 5.87 Controller output; end of learning; full load weight; $\theta_0 = 0°$. (From Kovačić, Z., Balenović, M., and Bogdan, S., *IEEE Contr. Syst. Mag.*, 18(3), 41–51, 1998. With permission.)

phase of learning, the tracking error has exceeded 25% of the imposed change of reference input.

Figure 5.90 and Figure 5.91 show the same group of responses after twelve learning iterations. As can be seen, the system follows the reference model very

TABLE 5.8
The Self-Organized Fuzzy Rule-Table

	NLE	NME	ZE	PME	PLE
NLDYF	0.980	0.295	0	0	0
NMDYF	0.325	−0.013	−0.743	0	0
ZYF	0	−0.334	A_{33}	0.018	0
PMDYF	0	−0.202	0.909	0.066	−0.105
PLDYF	0	0	0	−0.207	−0.783

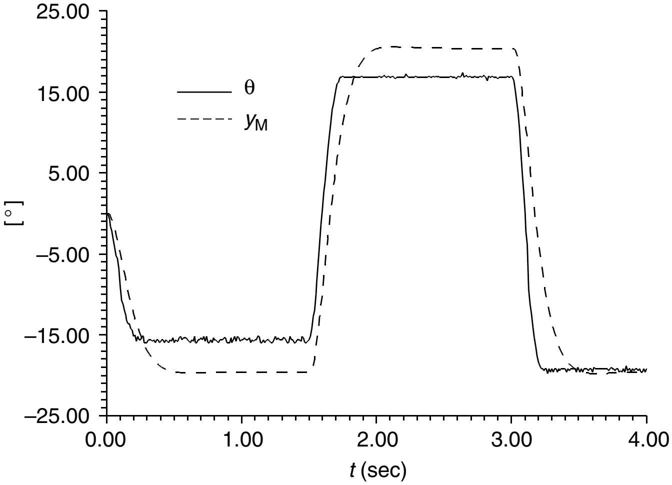


FIGURE 5.88 Reference model and system output; start of learning; reduced load weight. (From Kovačić, Z., Balenović, M., and Bogdan, S., *IEEE Contr. Syst. Mag.*, 18(3), 41–51, 1998. With permission.)

closely and the maximum tracking error value is kept below 10%, while the steady-state error is kept at zero.

From the control point of view, the worst case is control of the system under full load weight conditions in the operating point $\theta_0 = -90°$. Figure 5.92 shows the reference model and system output responses, while Figure 5.93 shows the tracking error response after first two runs of the system. The system output response clearly indicates that this is the most difficult starting operating point for the system what results in the highest tracking error (over 80% due to the presence of active load, which cannot be compensated only by means of a P controller).

Figure 5.94 and Figure 5.95 show the same group of responses after fourteen learning iterations. The system follows the reference model very closely and the maximum tracking error value is reduced almost 15 times and kept below 10%. The steady-state error is kept at zero indicating that the controller learned to compensate

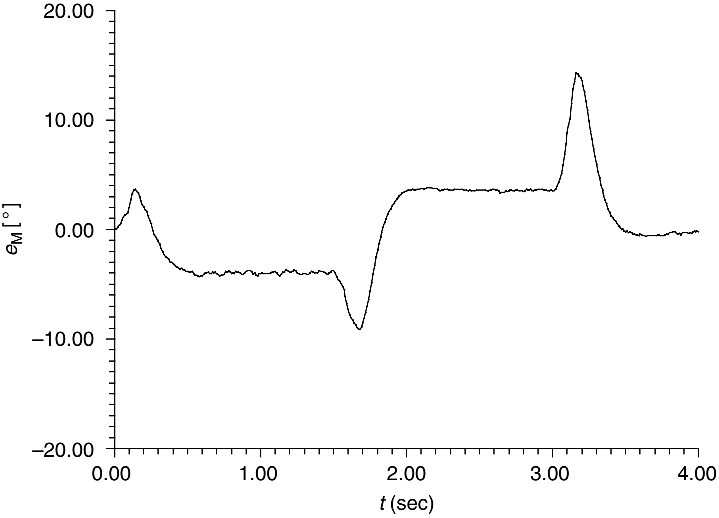


FIGURE 5.89 Tracking error; start of learning; reduced load weight; $\theta_0 = 0°$. (From Kovačić, Z., Balenović, M., and Bogdan, S., *IEEE Contr. Syst. Mag.*, 18(3), 41–51, 1998. With permission.)

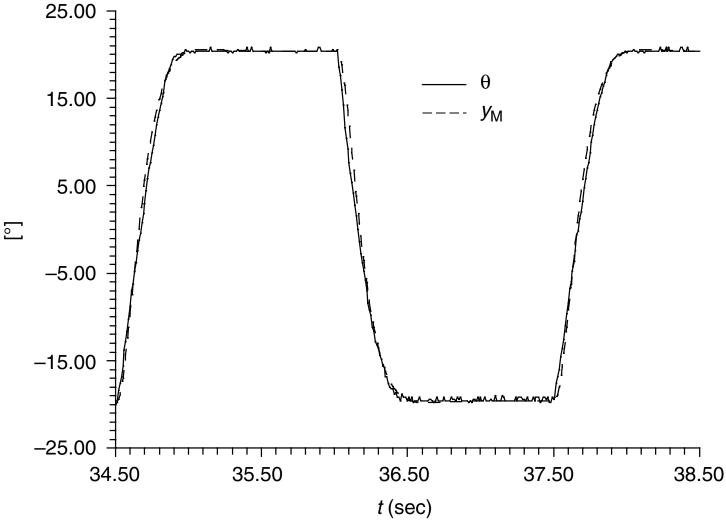


FIGURE 5.90 Reference model and system output; end of learning; reduced load weight. (From Kovačić, Z., Balenović, M., and Bogdan, S., *IEEE Contr. Syst. Mag.*, 18(3), 41–51, 1998. With permission.)

the influence of all system disturbances. The form of the self-learning fuzzy controller output is different in different movement directions due to the presence of all nonlinearities, as shown in Figure 5.96. In the case of the negative change of reference input, the controller, after short enforcing, starts braking since the direction of the load torque action coincides with the bar movement direction. The singleton

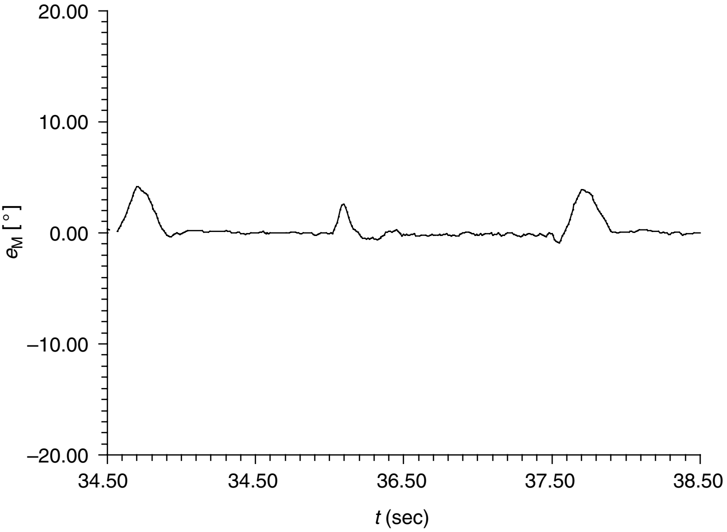


FIGURE 5.91 Tracking error; end of learning; reduced load weight; $\theta_0 = 0°$. (From Kovačić, Z., Balenović, M., and Bogdan, S., *IEEE Contr. Syst. Mag.*, 18(3), 41–51, 1998. With permission.)

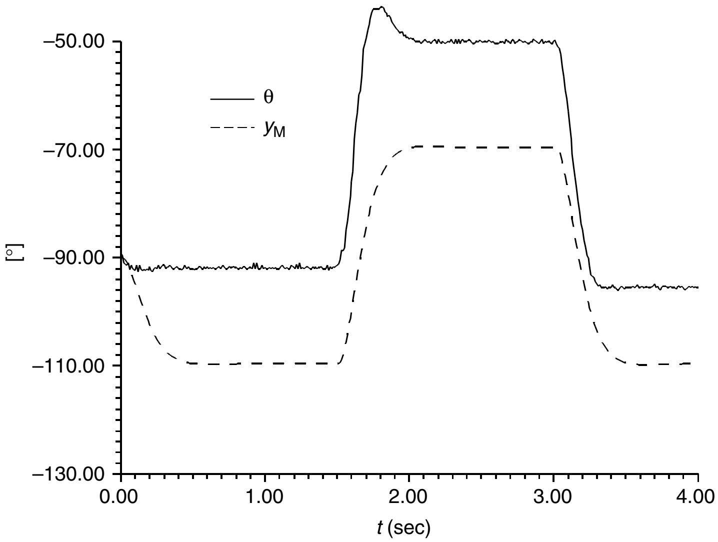


FIGURE 5.92 Reference model and system output; start of learning; full load weight; $\theta_0 = -90°$. (From Kovačić, Z., Balenović, M., and Bogdan, S., *IEEE Contr. Syst. Mag.*, 18(3), 41–51, 1998. With permission.)

values obtained by self-organization are shown in Table 5.9. Their values are expressed in volts.

In the case of a negative change of reference input, $A_{33} = -0.945$ V, while for a positive change, $A_{33} = -0.985$ V. One can notice that the difference between

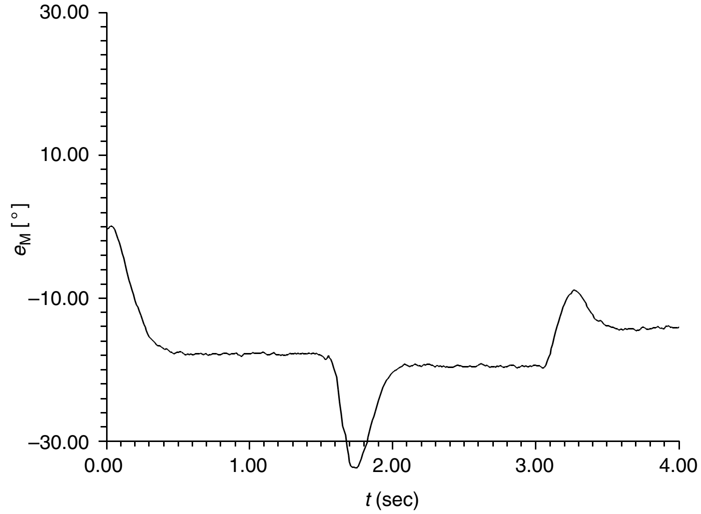


Figure 5.93 Tracking error; start of learning; full load weight; $\theta_0 = -90°$. (From Kovačić, Z., Balenović, M., and Bogdan, S., *IEEE Contr. Syst. Mag.*, 18(3), 41–51, 1998. With permission.)

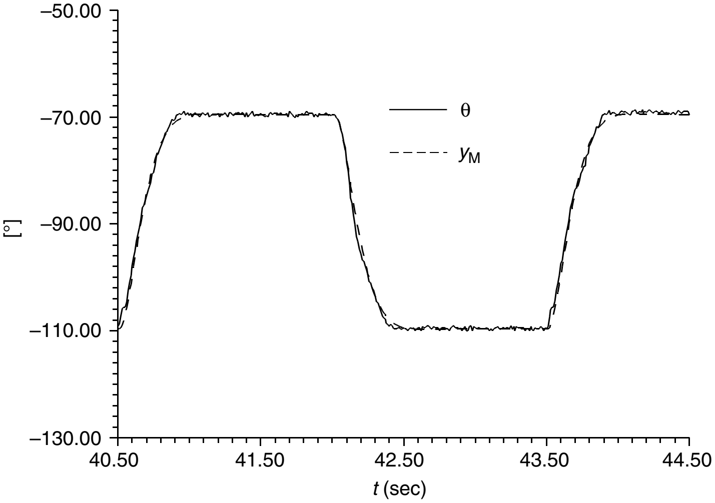


Figure 5.94 Reference model and system output; end of learning; full load weight; $\theta_0 = -90°$. (From Kovačić, Z., Balenović, M., and Bogdan, S., *IEEE Contr. Syst. Mag.*, 18(3), 41–51, 1998. With permission.)

these two values is smaller than in the case when operating point $\theta_0 = 0°$. It is due to variability of static friction with the operating point, caused by a radial influence of the bar on the bearings and the gear.

By comparison of Tables 5.8 and 5.9 one can find out that corresponding fuzzy rules have different singleton values in two different operating points. In other

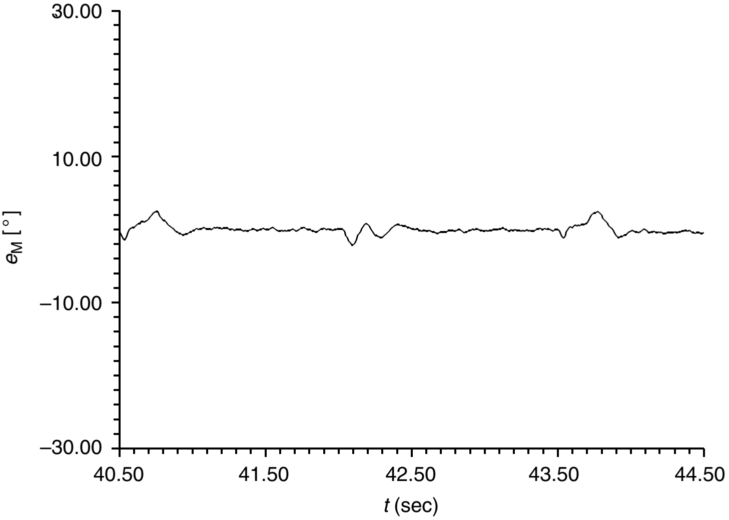


Figure 5.95 Tracking error; end of learning; full load weight; $\theta_0 = -90°$. (From Kovačić, Z., Balenović, M., and Bogdan, S., *IEEE Contr. Syst. Mag.*, 18(3), 41–51, 1998. With permission.)

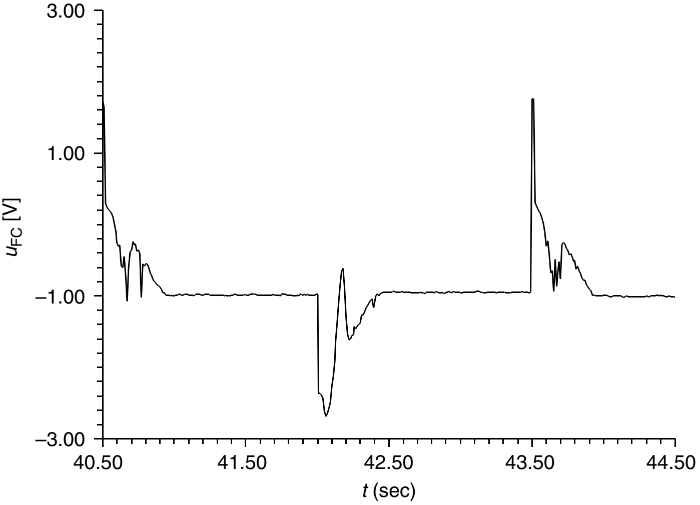


Figure 5.96 Self-learning fuzzy controller output; end of learning; full load weight; $\theta_0 = -90°$. (From Kovačić, Z., Balenović, M., and Bogdan, S., *IEEE Contr. Syst. Mag.*, 18(3), 41–51, 1998. With permission.)

words, such controllers are designed for constrained operating ranges around the operating points, and their operation out of these ranges would most likely start a new process of self-organization. The influence of operating point changes that are out of the expected controller's operating range is shown in Figure 5.97 and

TABLE 5.9
The Self-Organized Fuzzy Rule-Table

	NLE	NME	ZE	PME	PLE
NLDYF	−0.725	0.594	0	0	0
NMDYF	−1.038	−1.288	−1.095	0	0
ZDYF	−0.602	−1.499	A_{33}	0	0
PMDYF	0	−1.143	−0.939	−0.545	−1.398
PLDYF	0	−0.941	−0.893	−2.475	−1.578

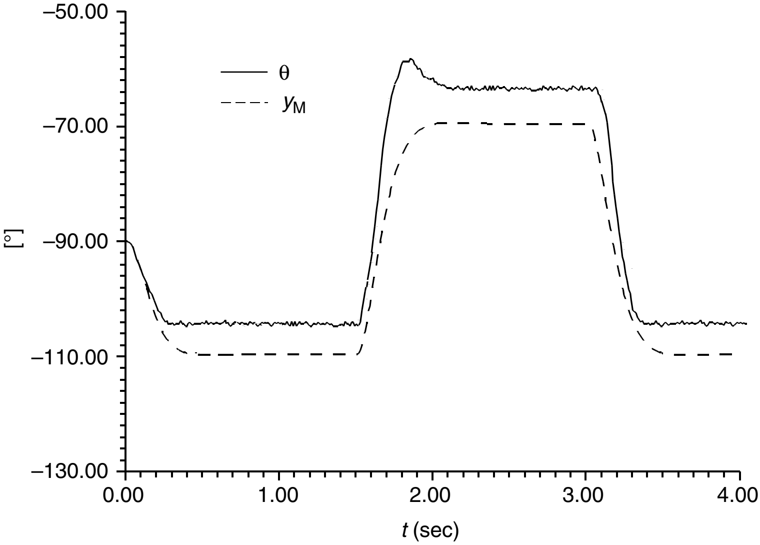


FIGURE 5.97 Reference model and system output; full load weight; $\theta_0 = -90°$, fuzzy controller designed for $\theta_0 = 0°$ (Table 5.8).

Figure 5.98. From the responses one can see that the nonlinear gravity-dependent load torque has a very large impact. In the operating point $\theta_0 = -90°$ if the system is controlled by the fuzzy rule-table obtained for the operating point $\theta_0 = 0°$, the steady-state error becomes evident (Figure 5.97). On the other hand, the system operating in $\theta_0 = 0°$ and controlled by the fuzzy controller obtained for $\theta_0 = -90°$ has become unstable (Figure 5.98).

In order to avoid a start of a new self-organization process every time that it is triggered by significant changes of the operating point, a corresponding fuzzy rule-table can be stored in the controller memory and called when it is needed. Such multiple fuzzy rule-table approach is described in Sections 4.2.3 and 5.3.3. In this way, the fuzzy controller is synthesized only once for a certain operating

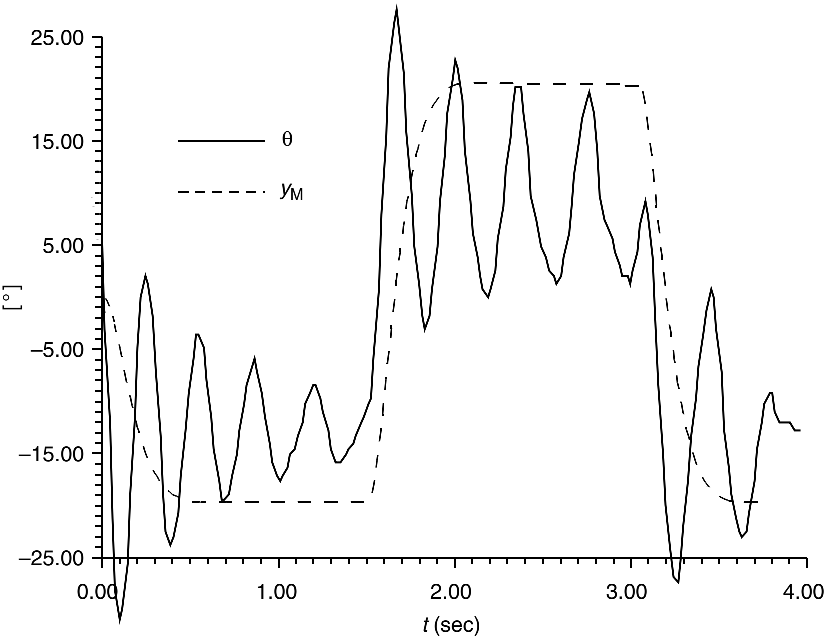


FIGURE 5.98 Reference model and system output; full load weight; $\theta_0 = 0°$, fuzzy controller designed for $\theta_0 = -90°$ (Table 5.9).

range and then repeatedly used for this range during the full range operation. The other possibility is to add load torque observers and thus compensate the nonlinear load torque impact by applying an adequate compensation signal [61]. Then one fuzzy rule-table could be sufficient for the whole operating range.

More experiments performed with changed reference model parameters ($t_m = 0.4$; 0.6; 0.8 sec) have indicated that the choice of reference model parameters is not a critical issue, since it has not influenced much the convergence of the learning process, as shown in Figures 5.99 to 5.104 for the case $t_m = 0.4$ sec. An attempt to accomplish further increase in robustness by adding an integral term in parallel to the P controller and the FLC has shown that the responses obtained in the presence of an integral term did not differ much from those obtained without it. Namely, the integral term was activated conditionally (i.e., only if it was necessary) after the major part of learning process was completed.

Since the selection of process approximation G_{pa} affects the dynamics of sensitivity functions, we have investigated the influence of zero-, first-, second-, and third-order linear process approximations on the performance of learning mechanism, while keeping the second-order reference model unchanged. It must be emphasized that all experiments finished successfully indicating a stable convergence of the learning process and providing satisfactory results of model tracking control.

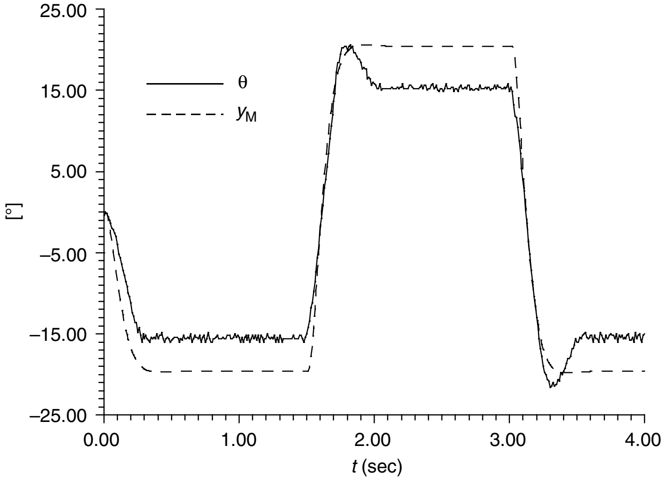


FIGURE 5.99 The reference model and system output responses; start of learning (full load). (From Kovačić, Z., Balenović, M., and Bogdan, S., *IEEE Contr. Syst. Mag.*, 18(3), 41–51, 1998. With permission.)

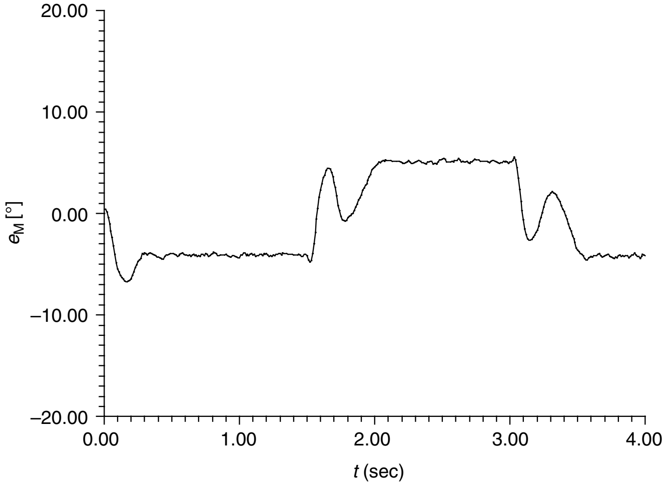


FIGURE 5.100 The tracking error responses; start of learning (full load weight, faster reference model). (From Kovačić, Z., Balenović, M., and Bogdan, S., *IEEE Contr. Syst. Mag.*, 18(3), 41–51, 1998. With permission.)

Although experimental results have confirmed the closed-loop stability, the described self-learning fuzzy control method lacks a rigorous proof of stability that would establish criteria for the synthesis of the fuzzy controller.

The obtained experimental results demonstrated the ability of the self-learning fuzzy controller to track the reference model and simultaneously compensate for operating point-dependent variations of shaft load. On the other hand, the described

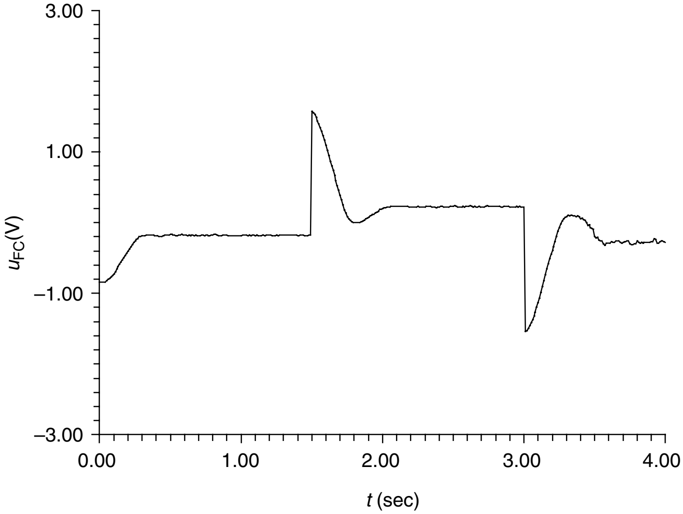


FIGURE 5.101 The self-learning fuzzy controller output responses; start of learning (full load weight, faster reference model). (From Kovačić, Z., Balenović, M., and Bogdan, S., *IEEE Contr. Syst. Mag.*, 18(3), 41–51, 1998. With permission.)

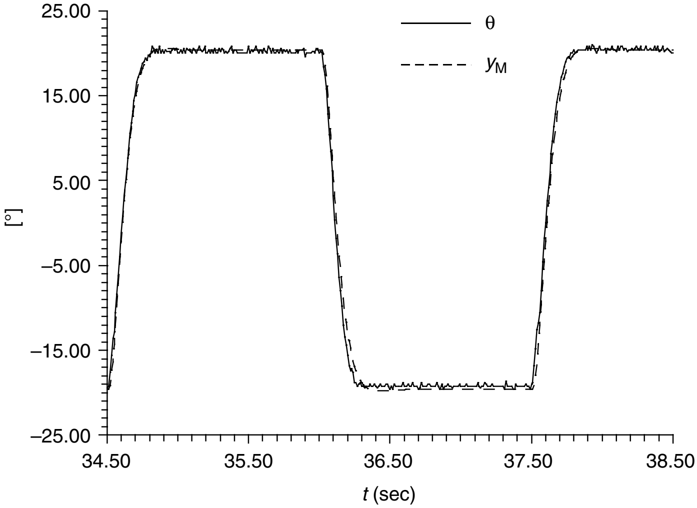


FIGURE 5.102 The reference model and system output responses; end of learning (full load weight, faster reference model). (From Kovačić, Z., Balenović, M., and Bogdan, S., *IEEE Contr. Syst. Mag.*, 18(3), 41–51, 1998. With permission.)

strategy of learning is not effective for systems exposed to external disturbances, which do not allow any change of the reference input (usually called stabilization systems). If such self-learning strategy could be successfully found, it could be combined with described learning concept to cover a wider spectrum of potential applications.

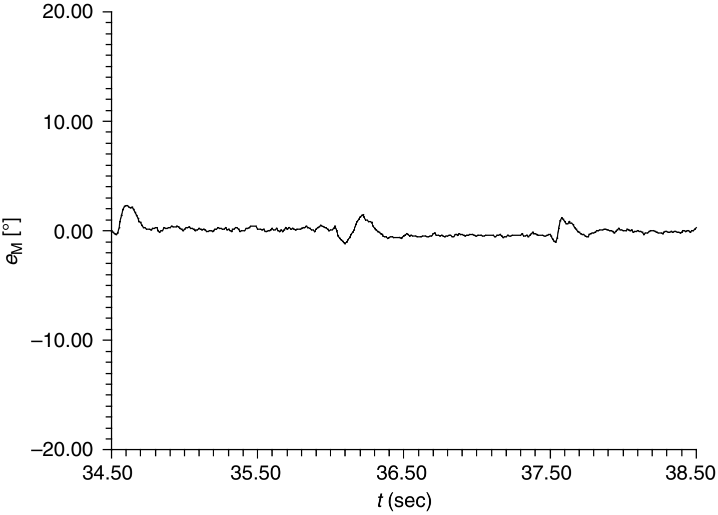


FIGURE 5.103 The tracking error responses; end of learning (full load weight, faster reference model). (From Kovačić, Z., Balenović, M., and Bogdan, S., *IEEE Contr. Syst. Mag.*, 18(3), 41–51, 1998. With permission.)

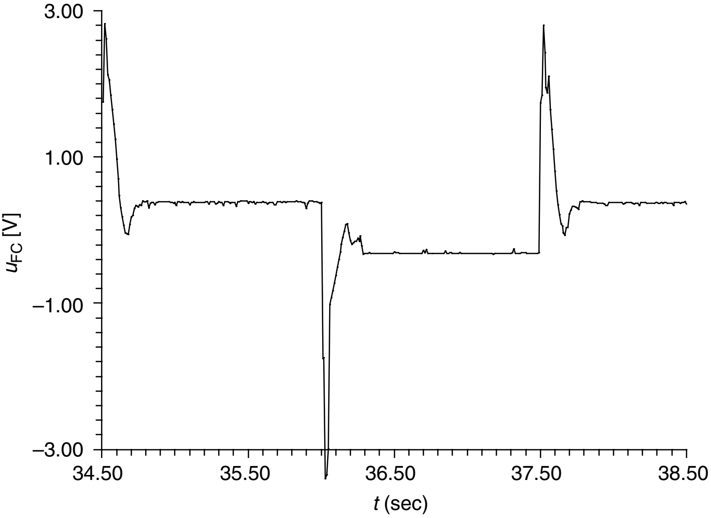


FIGURE 5.104 The self-learning fuzzy controller output responses; end of learning (full load weight, faster reference model). (From Kovačić, Z., Balenović, M., and Bogdan, S., *IEEE Contr. Syst. Mag.*, 18(3), 41–51, 1998. With permission.)

5.3.3 Example: Multiple Fuzzy Rule Table-Based Control

In Example 5.4, we have pointed out that any change of operating point that lies out of the self-organizing fuzzy logic controller input universes of discourse could potentially deteriorate the quality of control and even initiate a new self-organization process. As described in Section 4.2.3, adaptation to changes of operating points can be achieved by using multiple fuzzy rule tables designed for respective operating regions. A heuristic design of a larger number of fuzzy controllers could be a repulsive job, but self-organization methods could make it very attractive.

In this example, we demonstrate control of a nonlinear positioning servo system described in detail in Example 5.1 by using multiple position-dependent fuzzy rule tables [62]. The block structure of the nonlinear control process affected by static friction and nonlinear gravitational load is shown in Figure 5.81. Since changes of a gravitational load are described with the sine (cosine) law, that variable changes not only in a nonlinear way but also periodically. Therefore, instead of using one fuzzy rule-table for the full range (360°) positioning, the use of multiple fuzzy rule tables for selected operating regions is applied. The whole operating range in the studied example has been split into 12 operating regions. Six regions from 0 to 150° are shown in Figure 5.105. Similarly, as shown in Figure 5.106, corresponding fuzzy rule tables cover other six regions from 180 to 330°. It must be noted that except output singletons, other fuzzy controller parameters are the same in all operating regions. Since both positive and negative changes of the reference input initiate learning iterations, each fuzzy rule-table depicted in Figure 5.105 and Figure 5.106 actually has two values of the central singleton A_{33} (please refer for better understanding to Tables 5.8 and 5.9). This may be perceived as a controller with a total number of 24 fuzzy rule tables.

The transition from one fuzzy rule-table to another has been managed in a smooth and bumpless way due to the hysteresis of 2° embedded in the algorithm for switching of adjacent fuzzy rule tables.

The question arises whether the stability of such servo control loop could be guaranteed or not. In Section 5.1 we showed that the stability of particular model reference-based adaptive fuzzy control systems can be observed by using a Lyapunov approach [63]. Here, only practical proofs of stability can be reached based on the fact that a model reference-based learning process converges, or that a desired system behavior is provided.

In order to get a better assessment of the self-learning multiple fuzzy rule table-based control, the servo system has been first controlled with a single fuzzy rule-table controller obtained after learning around the operating position $\theta_0 = 0°$ for the customized sequence of reference input changes shown in Figure 5.107. The responses of a model tracking error are shown in Figure 5.108. The error is negligible in the region around $\theta_0 = 0°$, while in other regions it increases up to 8°, while the maximum steady-state error reaches 2°. Very good following of the reference model dynamics in only one region and deterioration of performance in other regions clearly shows the motivation for introduction of multiple fuzzy rule-table control.

Figure 5.109 shows the reference model and system output responses obtained in the case of using a multiple fuzzy rule-table controller for the same customized sequence of reference input changes as in the previous experiment. It must be

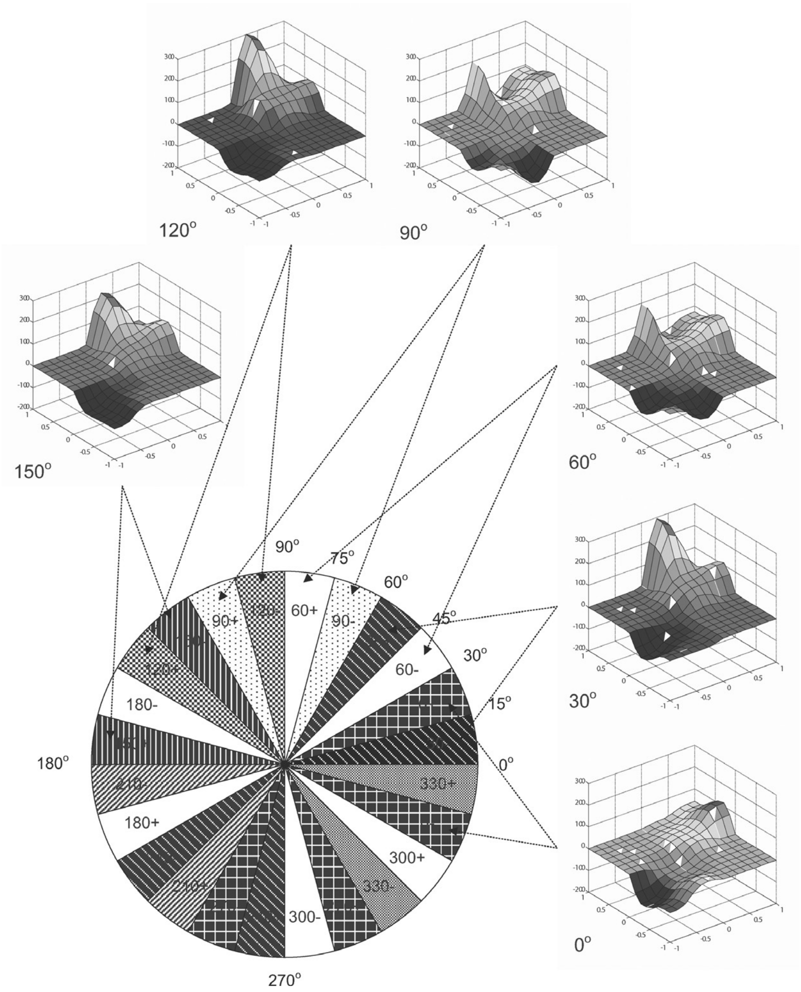


FIGURE 5.105 Distribution of fuzzy rule tables (controllers) for the operating range 0° to 150°. (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *IFAC Robot Contr. 2000*, 229–233, 2000. With permission from Elsevier.)

noted that all fuzzy rule tables used in the experiment have had a stationary form obtained after completion of an earlier learning phase for each operating region. Also, care has been taken during implementation to ensure bumpless switching between fuzzy rule tables to prevent a possible chattering problem. The responses of the reference model tracking error are shown in Figure 5.110. As can be seen, very good following of the reference model dynamics has been achieved over the full operating range and deterioration of performance is significantly reduced. Thus the maximum tracking error and the steady-state error are now four times smaller (2.2° and 0.44°, respectively). The experimental results confirm that introduction

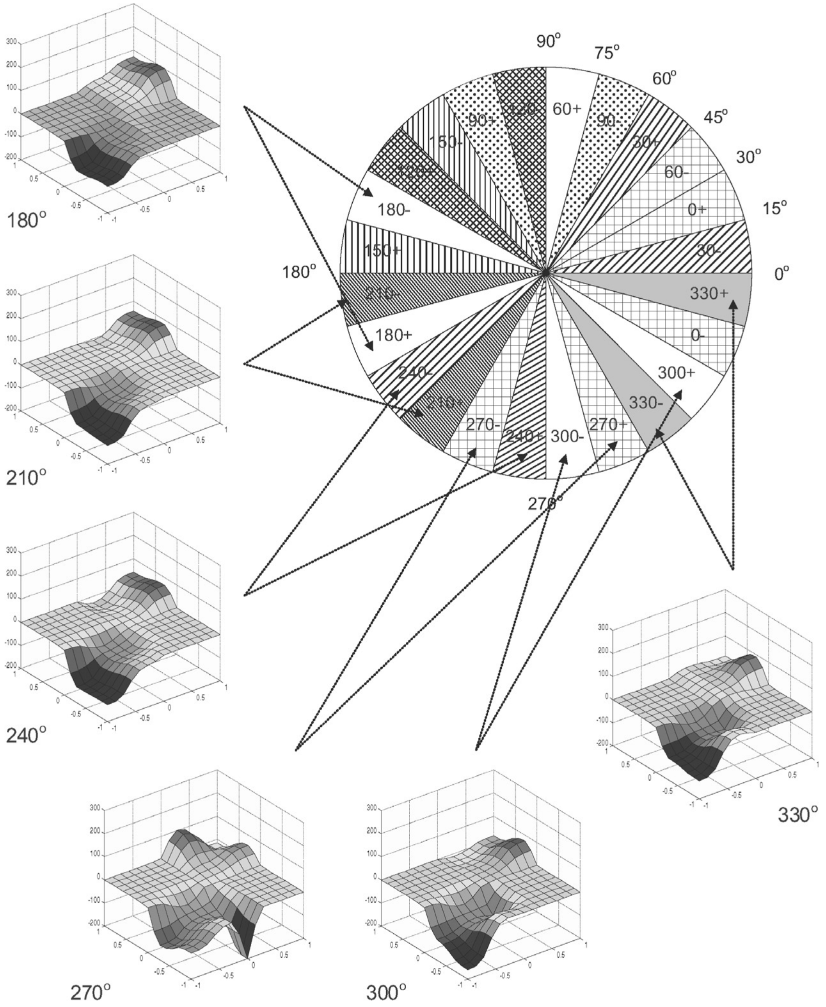


FIGURE 5.106 Distribution of fuzzy rule tables (controllers) for the operating range 180° to 330°.

of the multiple fuzzy rule-table controller is justifiable, since it has led to a much better servo system performance.

From the practical point of view, it is important that the waveform of the self-learning fuzzy controller output is such that it does not wear the robot joint actuators and mechanical parts. The multiple fuzzy rule-table controller output is shown in Figure 5.111. The adopted numeration of fuzzy rule tables is shown in Figure 5.112. The sequence of active fuzzy rule tables is depicted in Figure 5.113. It can be seen that switching between tables is bumpless, and the controller output waveform is acceptable for practical servo applications.

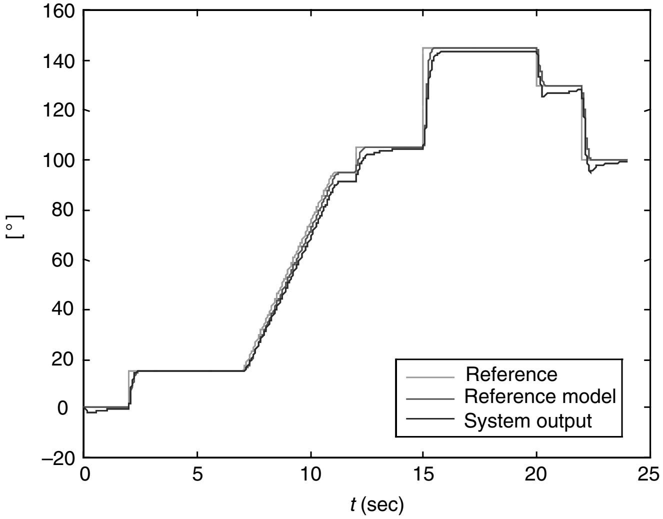


FIGURE 5.107 Tracking of reference trajectory with the single fuzzy rule-table controller. (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *IFAC Robot Contr. 2000*, 229–233, 2000. With permission from Elsevier.)

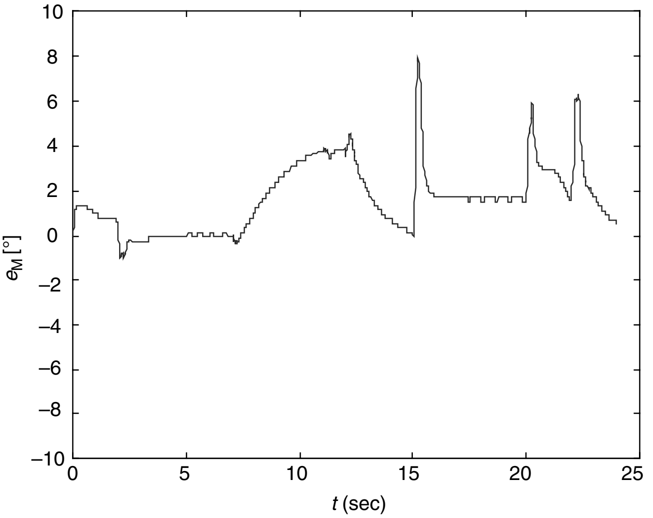


FIGURE 5.108 Model tracking error responses with the single fuzzy rule-table controller. (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *IFAC Robot Contr. 2000*, 229–233, 2000. With permission from Elsevier.)

After comparison of experimental results obtained in the cases of using a single fuzzy rule-table and a multiple fuzzy rule-table controller, it has been clearly demonstrated that the latter controller provides much better performance with closer tracking of reference model dynamics and very small steady-state errors.

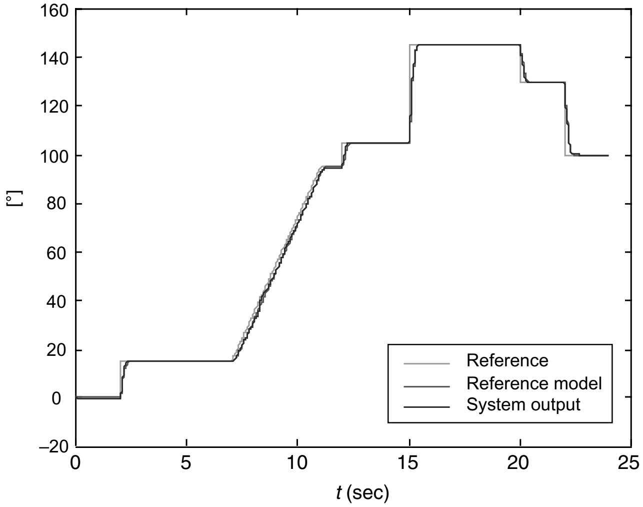


FIGURE 5.109 Tracking of the reference trajectory for the multiple fuzzy rule-table controller. (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *IFAC Robot Contr. 2000*, 229–233, 2000. With permission from Elsevier.)

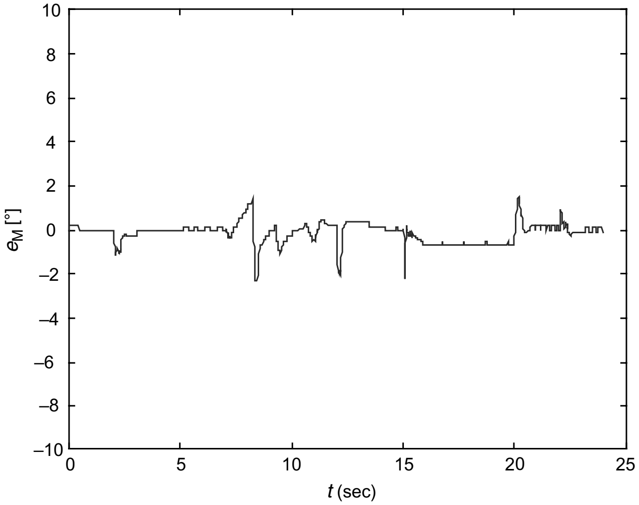


FIGURE 5.110 Model tracking error responses for the multiple fuzzy rule-table controller. (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *IFAC Robot Contr. 2000*, 229–233, 2000. With permission from Elsevier.)

5.3.4 Self-Organizing Fuzzy Control with a Self-Learning Integral Term

All self-learning fuzzy logic controllers described in Chapter 5 are PD-type controllers. The basic controller structure described in Section 5.2 (see Figure 5.18)

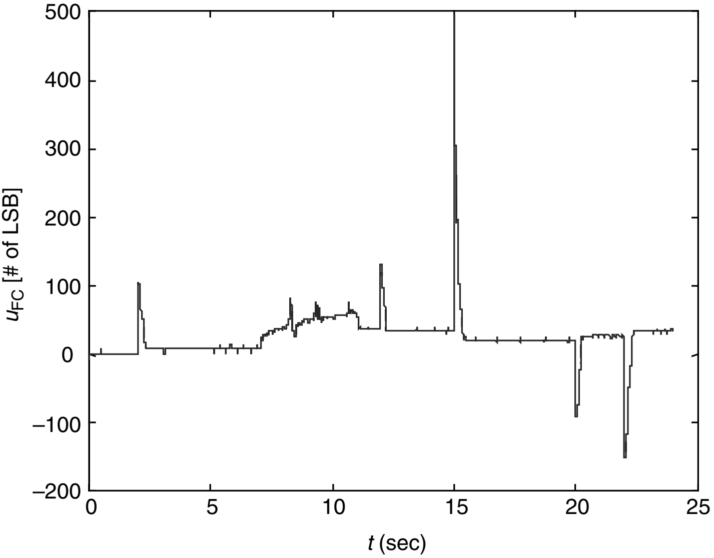


FIGURE 5.111 The self-learning fuzzy controller output responses. (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *IFAC Robot Contr. 2000*, 229–233, 2000. With permission from Elsevier.)

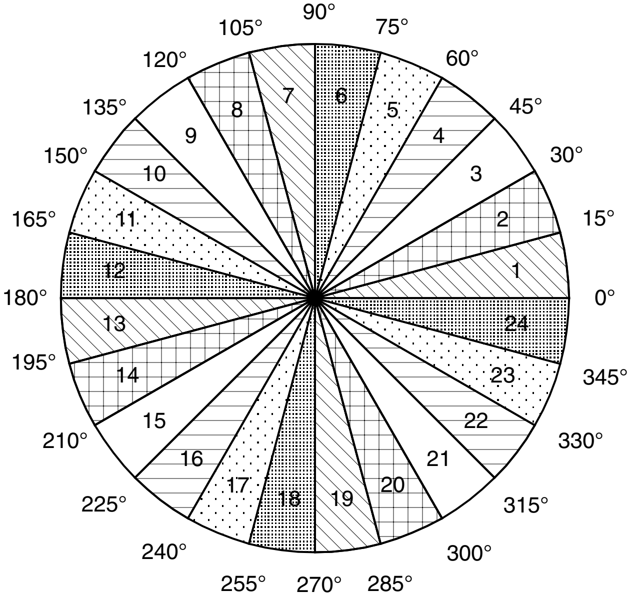


FIGURE 5.112 Numerical notation of fuzzy rule tables. (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *IFAC Robot Contr. 2000*, 229–233, 2000. With permission from Elsevier.)

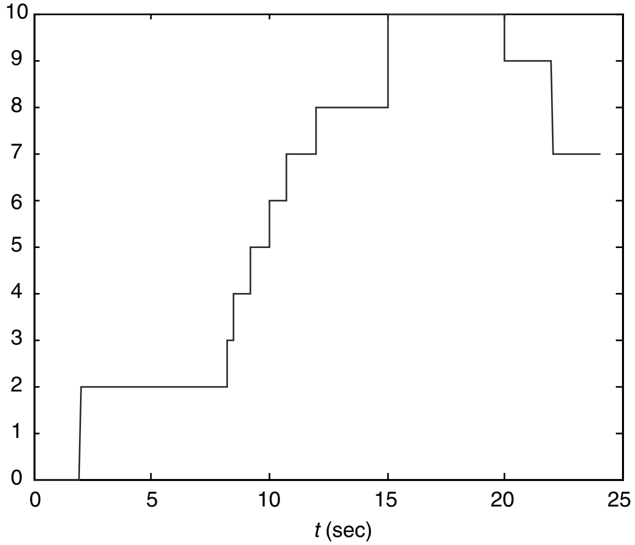


FIGURE 5.113 The fuzzy rule-table switching sequence. (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *IFAC Robot Contr. 2000*, 229–233, 2000. With permission from Elsevier.)

contains a PD-type self-learning fuzzy controller and a self-learning feedforward term. Such controller ensures very good reference model tracking but it cannot compensate a steady-state error caused by external disturbances, which were not present during the self-organization process.

To cope better with a possible steady-state accuracy problem, as shown in Figure 5.114, an integral term is added to the self-learning fuzzy controller (5.71)

$$u(k) = \Gamma[e(k), \Delta y_{\mathrm{f}}(k), \boldsymbol{\lambda}] + k_3(k)u_{\mathrm{r}}(k) + (1-\xi)u_{\mathrm{i}}(k) \qquad (5.97)$$

where $u_{\mathrm{i}}(k) = u_{\mathrm{i}}(k-1) + k_{\mathrm{i}}e(k)$, k_{i} is an integral gain coefficient, and ξ is a Boolean parameter, which becomes

$$\begin{aligned} \xi &= 1 \quad \text{if } e(k) \notin \mathrm{ZE} \ \wedge \Delta y_{\mathrm{f}}(k) \notin \mathrm{ZDYF} \\ \xi &= 0 \quad \text{if } e(k) \in \mathrm{ZE} \ \vee \Delta y_{\mathrm{f}}(k) \in \mathrm{ZDYF} \end{aligned} \qquad (5.98)$$

Any new parameter introduction into a self-learning fuzzy control scheme can be justified only if the value of this parameter is determined in an automatic way. The integral gain coefficient k_{i} will be determined by a sensitivity model-based learning algorithm activated after the completion of main learning algorithms (5.88) and (5.89). As shown in Figure 5.114, the fuzzy controller structure contains a mechanism for generation of signal u_{d}, which simulates a persistent source of disturbance of known magnitude and duration that acts on the controlled process from the controller direction. Instead of learning in the presence

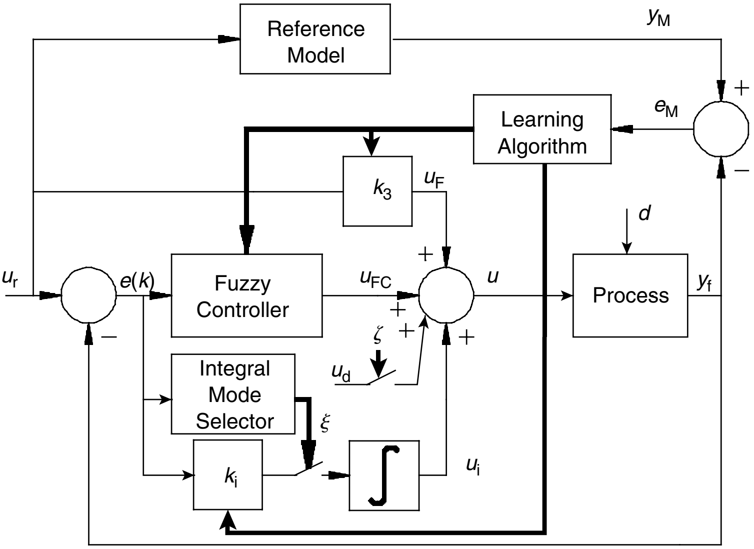


FIGURE 5.114 The structure of a self-learning fuzzy controller with a self-learning integral term. (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *IFAC Robot Contr. 2000*, 229–233, 2000. With permission from Elsevier.)

of stochastic and nonpersistent external disturbances, learning of k_i is performed during a sequence of internally generated alternating step changes of $u_d(k)$ added to the controller output. To ensure a good result of learning, it is essential to make a fair estimation of a maximally expected magnitude and duration of simulated disturbance variations. By that process, the reference input $u_r(k)$ should be kept constant. Activation of u_d depends on the state of Boolean parameter ζ, which is set to one during the period of learning k_i. Accordingly, during execution of main learning algorithms (5.88) and (5.89), as well as during routine operation of the fuzzy controller, ζ is set to zero.

Following the principle applied in the synthesis of main learning algorithms, the sensitivity function of integral gain coefficient k_i can be determined as

$$\eta_{k_i}(k) = \frac{\partial y_f(k)}{\partial k_i} = \frac{\partial y_f(k)}{\partial u(k)}\frac{\partial u(k)}{\partial k_i} = G_p\frac{\partial u(k)}{\partial k_i} = G_p\eta^*_{k_i}(k) \tag{5.99}$$

where $\eta^*_{k_i}(k) = \eta^*_{k_i}(k-1) + e(k)$.

Consequently, the total differential of the system output attains the form

$$\Delta y_f(k) \approx \zeta\sum_{i=1}^{n} G_p\left\{\frac{\partial}{\partial\lambda_{c_i}}\{\Gamma[e(k), \Delta y_f(k), \boldsymbol{\lambda}]\}\right\}\Delta\lambda_{c_i} + (1-\zeta)G_p\eta^*_{k_i}(k)\Delta k_i \tag{5.100}$$

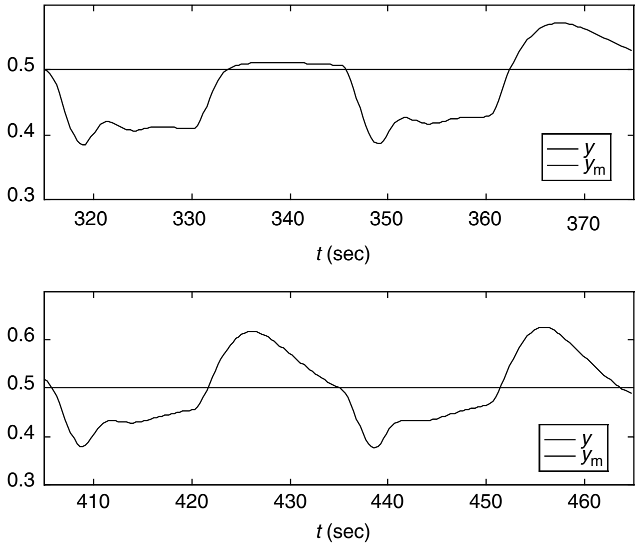


FIGURE 5.115 The reference model and system output responses during the start-up (above) and the end of learning (below) of integral gain coefficient k_i. (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *IFAC Robot Contr. 2000*, 229–233, 2000. With permission from Elsevier.)

where one can see that the change of parameter ζ switches off the tuning of k_i during the tuning of fuzzy controller parameters (including feedforward coefficient k_3) and vice versa.

From (5.100), the total differential of system output during the tuning of k_i ($\zeta = 0$) is

$$\Delta y_\mathrm{f}(k) \approx G_\mathrm{p}\eta^*_{k_\mathrm{i}}(k)\Delta k_\mathrm{i} \tag{5.101}$$

The sensitivity function $\eta_{k_\mathrm{i}}(k)$ is calculated in each control interval, while the increment of gain coefficient k_i is calculated at the very end of the learning interval. Taking into account that in the model reference-based control systems $\Delta y_\mathrm{f}(k)$ can be replaced with $e_\mathrm{M}(k)$, an algorithm for tuning of the integral gain coefficient has the form:

$$\Delta k_\mathrm{i} = \frac{e^\kappa_\mathrm{M}(v^\kappa_{k_\mathrm{i}})}{\eta^\kappa_{k_\mathrm{i}}(v^\kappa_{k_\mathrm{i}})} \tag{5.102}$$

where $v^\kappa_{k_\mathrm{i}}$ is the last control interval in the current learning iteration.

During learning intervals the simulated disturbance signal alters between the constant maximal value and zero. In order to improve learning stability, calculation of Δk_i is performed only in the nonzero disturbance intervals. Learning is stopped if the obtained value of k_i satisfactorily cancels the steady-state error in the given time.

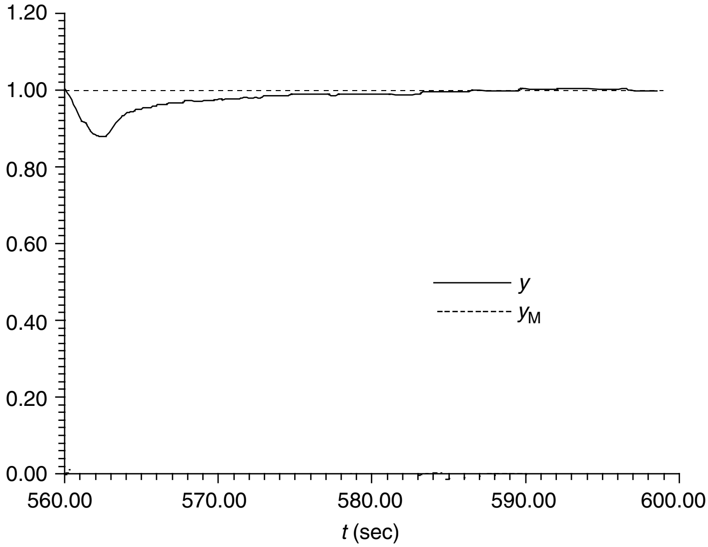


Figure 5.116 The system output response in the presence of disturbance obtained with a self-learning integral term after completion of learning.

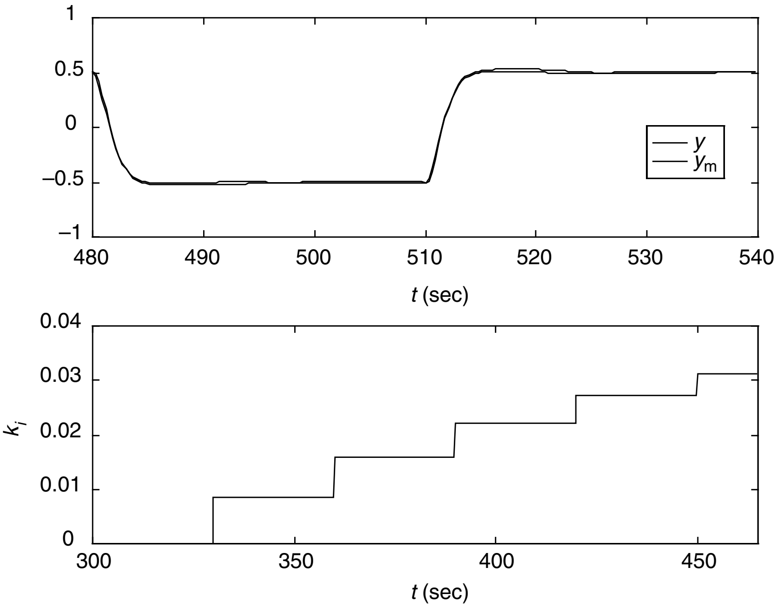


Figure 5.117 The reference model and system output responses after the end of learning (above); convergence of integral gain coefficient k_i (below). (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *IFAC Robot Contr. 2000*, 229–233, 2000. With permission from Elsevier.)

The proposed SLFLC has been tested by simulation in the case of controlling a third-order linear control process (3.56) described in Section 3.4. Figure 5.115 shows the reference model and system output responses at the start-up and the end of learning of k_i, respectively (please notice that learning of k_i started at $t = 330$ sec, that is, after completion of the main learning algorithm). The period of simulated disturbance $u_\mathrm{d}(k)$ is 30 sec, and its amplitude is $U_\mathrm{d} = 0.3$. The learning algorithm is executed during the first half of the period, while the second half (15 sec) is needed to drain the integrator before the next learning iteration. The system output obtained after all learning algorithms were completed is shown in Figure 5.116. After inclusion of the self-learning integral term, the system's response to the changes of reference input insignificantly changed, having just a little bit higher peak value (see Figure 5.117). Convergence of k_i during the learning process is also illustrated in Figure 5.117. In the studied case, the whole process of learning ended in six iterations.

The results obtained in the simulated experiments have confirmed a stable convergence of learning and effective reference model tracking control without steady state errors in the presence of external disturbance. In Reference 60 it has been shown that such approach also satisfactorily improves controller robustness in the case of nonlinear process control.

References

1. Fuzzy TECH User's Guide, Microchip Technology Inc., 1994.
2. Roger Jang, J.S. and Gulley, N., "Fuzzy logic toolbox — for use with MATLAB," The Mathworks Inc., 1995.
3. Procyk, T.J. and Mamdani, E.H., "A linguistic self-organizing process controller," *Automatica*, 15, 15–30, 1979.
4. Shao, S., "Fuzzy self-organizing controller and its application for dynamic processes," *Fuzzy Sets and Systems*, 26, 151–164, 1988.
5. Harris, C. and Moore, C., "Intelligent identification and control for AGVs using adaptive fuzzy based algorithms," *Engineering Applications of Artificial Intelligence*, 2, 267–285, 1989.
6. Guely, F. and Siarry, P., "A centred formulation of Takagi–Sugeno rules for improved learning efficiency," *Fuzzy Sets and Systems*, 62, 277–285, 1994.
7. Ishibuchi, H., Noyaki, K., Tanaka, H., Hosaka, Y., and Matsuda, M., "Empirical study on learning in fuzzy systems by rice taste analysis," *Fuzzy Sets and Systems*, 64, 129–144, 1994.
8. Gürocak, H.B. and de San Lazaro, A., "A fine tuning method for fuzzy rule bases," *Fuzzy Sets and Systems*, 67, 147–161, 1994.
9. Vuorimaa, P., "Fuzzy self-organizing map," *Fuzzy Sets and Systems*, 66, 223–231, 1994.
10. Maeda, M. and Murakami, S., "A self-tuning fuzzy controller," *Fuzzy Sets and Systems*, 51, 29–40, 1992.
11. Wu, Z.Q., Wang, P.Z., Heng, T.H., and Song, S.S., "A rule self-regulating fuzzy controller," *Fuzzy Sets and Systems*, 47, 13–21, 1992.

12. He, S.Z., Tan, S., Hang, C.C., and Wang, P.Z., "Control of dynamical processes using an on-line rule-adaptive fuzzy control system," *Fuzzy Sets and Systems*, 54, 11–22, 1993.
13. Chang, H.-C. and Wang, M.-H., "Neural network-based self-organizing fuzzy controller for transient stability of multimachine power systems," *IEEE Transactions on Energy Conversion*, 10, 339–347, 1995.
14. Kasabov, N.K., "Learning fuzzy rules and approximate reasoning in fuzzy neural networks and hybrid systems," *Fuzzy Sets and Systems*, 82, 135–139, 1996.
15. Berenji, H.R. and Khedkar, P., "Learning and tuning fuzzy logic controllers through reinforcements," *IEEE Transactions on Neural Networks*, 3, 724–740, September 1992.
16. Jang, R., "Self-learning fuzzy controllers based on temporal back propagation," *IEEE Transactions on Neural Networks*, 3, 714–723, 1992.
17. Kovačić, Z., Bogdan, S., and Crnosija, P., "Design and stability of self-organizing fuzzy control of high-order systems," *Proceedings of the 10th IEEE ISIC*, Monterey, pp. 389–394, 1995.
18. King, P.J., Burnham, K.J., and James, D.J.G., "A model reference self-organizing fuzzy logic controller," *Control and Computers*, 23, 56–65, 1995.
19. Kovačić, Z. and Bogdan, S., "Model reference adaptive control of high-order systems," *An International Journal for Engineering Applications of Artificial Intelligence* (Pergamon Press), 7, 501–511, 1994.
20. Layne, J.R. and Passino, K.M., "Fuzzy model reference learning control for cargo ship steering," *IEEE Control Systems Magazine*, 13, 23–34, December 1993.
21. Layne, J.R. and Passino, K.M., "Fuzzy model reference learning control," *Journal of Intelligent and Fuzzy Systems*, 4, 33–47, 1996.
22. Kwong, W.A. and Passino, K.M., "Dynamically focused fuzzy learning control," *IEEE Transactions on Systems, Man and Cybernetics*, 26, 53–74, 1996.
23. Kovačić, Z., Balenović, M., and Bogdan, S., "Sensitivity-based self-learning fuzzy logic control for a servo system," *The IEEE Control Systems Magazine*, 18, 41–51, 1998.
24. Kang, H. and Vachtsevanos, G., "Adaptive fuzzy logic control: explicit adaptive control with Lyapunov stability and learning capability," in *Proceedings of the American Control Conference*, Chicago, Vol. 3, pp. 2279–2283, 1992.
25. Bai, E.-W., "Parameterization and adaptive compensation of friction forces," *International Journal of Adaptive Control Signal Process*, 11, 21–32, 1997.
26. Canudas de Wit, C., Astrom, K.J., and Braun, K., "Adaptive friction compensation in DC-motor drives," *IEEE Journal of Robotics and Automation*, RA-3, 681–685, 1987.
27. Canudas de Wit, C.A., Olsson, H., Astrom, K.J., and Lischinsky, P., "A new model for control of systems with friction," *IEEE Transactions on Automatic Control*, AC-40, 419–425, 1995.
28. Canudas de Wit, C.A. and Lischinsky, P., "Adaptive friction compensation with partially known dynamic friction model," *International Journal of Adaptive Control Signal Processing*, 11, 65–80, 1997.
29. Selmic, R.R. and Lewis, F.L., "Neural network approximation of piecewise continuous functions: application to friction compensation," *Proceedings of IEEE International Symposium on Intelligent Control*, Istanbul, 1997.
30. Armstrong-Helouvry, B., Dupont, P., and Canudas de Wit, C., "A survey of models, analysis tools and compensation methods for the control of machines with friction," *Automatica*, 30, 1083–1138, 1994.

31. Jee, S. and Koren, Y., "A self-organizing fuzzy logic control for friction compensation in feed drives," *Proceedings of the American Control Conference*, Seattle, pp. 205–209, 1995.
32. Wang, L.X., *Adaptive Fuzzy Systems and Control: Design and Stability Analysis*, Prentice Hall, New York, 1994.
33. Wang, L.X. and Mendel, J.M., "Fuzzy basis functions, universal approximation, and orthogonal least-squares learning," *IEEE Transactions on Neural Networks*, 3, 807–814, 1992.
34. Kovačić, Z., Bogdan, S., and Balenović, M., "A model reference & sensitivity model-based self-learning fuzzy logic controller as a solution for control of nonlinear servo systems," *IEEE Transactions on Energy Conversion*, 13, 1479–1484, 1999.
35. Wilkie, D.F. and Perkins, W.R., "Generation of sensitivity functions for linear systems using low-order models," *IEEE Transactions on Automatic Control*, AC-14, 123–130, 1969.
36. Porter, B., "Eigenvalue sensitivity of modal control systems to loop-gain variations," *International Journal of Control*, 10, 159–162, 1969.
37. Porter, W.A., "Sensitivity problems in linear systems," *IEEE Transactions on Automatic Control*, AC-10, 301–307, July 1965.
38. Reddy, D.C., "Eigenfunction sensitivity and the parameter variation problem," *International Journal of Control*, 9, 227–233, 1969.
39. Morgan, B.S. Jr., "Sensitivity analysis and synthesis of multivariable systems," *IEEE Transactions on Automatic Control*, AC-11, 506–512, July 1966.
40. Bongiorno, J.J., "Minimum sensitivity design of linear multivariable feedback control systems by matrix spectral factorization," *IEEE Transactions on Automatic Control*, AC-14, 665–673, 1969.
41. Rillings, J.H. and Roy, R.J., "Analog sensitivity design of Saturn V launch vehicle," *IEEE Transactions on Automatic Control*, AC-15, 437–442, 1970.
42. Ringlee, R.J., "Sensitivity methods for economic dispatch of hydroelectric plants," *IEEE Transactions on Automatic Control*, AC-10, 315–322, July 1965.
43. Henderson, D.E., Kokotovic, P.V., Schiano, J.L., and Rhode, D.S., "Adaptive control of an arc welding process," *IEEE Control Systems Magazine*, 13, 49–53, February 1993.
44. Crnošija, P., Bogdan, S., and Kovačić, Z., "Sensitivity model and synthesis of dead-beat algorithms in digital servosystems," *Automatica*, 30, 1345–1350, 1994.
45. Rohrer, R.A. and Sobral, M., "Sensitivity considerations in optimal system design," *IEEE Transactions on Automatic Control*, AC-10, 43–48, 1965.
46. Belanger, P.R., "Some aspects of control tolerances and first-order sensitivity in optimal control systems," *IEEE Transaction on Automatic Control*, AC-11, 77–83, 1966.
47. Dorato, P. and Kestenbaum, A., "Application of game theory to the sensitivity design of optimal systems," *IEEE Transaction on Automatic Control*, AC-12, 85–87, February 1967.
48. Frank, P.M., *Introduction to System Sensitivity Theory*, Academic Press, 1978.
49. Guely, F. and Siarry, P., "A centred formulation of Takagi–Sugeno rules for improved learning efficiency," *Fuzzy Sets and Systems*, 62, 277–285, 1994.
50. Ishibuchi, H., Noyaki, K., Tanaka, H., Hosaka, Y., and Matsuda, M., "Empirical study on learning in fuzzy systems by rice taste analysis," *Fuzzy Sets and Systems*, 64, 129–144, 1994.

51. Gürocak, H.B. and de San Lazaro, A., "A fine tuning method for fuzzy rule bases," *Fuzzy Sets and Systems*, 67, 147–161, 1994.
52. Jang, J.-S.R. and Gulley, N., "Fuzzy logic toolbox — for use with MATLAB," The MathWorks Inc., 1995.
53. Sage, A.P., *Optimum Systems Control*, Prentice-Hall, New York, 1968.
54. Tomovic, R. and Bekey, G.A., "Adaptive sampling based on amplitude sensitivity," *IEEE Transaction on Automatic Control*, AC-11, 282–284, April 1966.
55. Bekey, G.A. and Tomovic, R., "Sensitivity of discrete systems to variation of sampling interval," *IEEE Transactions on Automatic Control*, AC-11, 284–287, April 1966.
56. Shane, B.A. and Barnett, S., "Sensitivity of stable linear systems," *IEEE Transactions on Automatic Control*, AC-17, 148–150, February 1972.
57. Bogdan, S., Crnošija, P., Kovačić, Z., and Stajić, D., "Adaptive time optimal model reference and sensitivity based control of servosystems," *Proceedings of the 13th IFAC Triennial World Congress*, Vol. K, San Francisco, pp. 187–192, 1996.
58. Bogdan, S., Crnošija, P., Kovačić, Z., and Stajić, D., "Self-tuning time optimal control of servosystems by using a reference model and a sensitivity model," *Proceedings of the IEEE Conference on Control Applications CCA 95*, Albany, pp. 730–735, 1995.
59. Bogdan, S. and Kovačić, Z., "On the design of a self-learning fuzzy controllers for nonlinear control systems by using a reference model and a sensitivity model," *Proceedings of the 4th IEEE Mediterranean Symposium on New Directions in Control and Automation*, Chania, pp. 799–804, 1996.
60. Kovačić, Z., Bogdan, S., and Balenović, M., "Robustness improvement of a model reference & sensitivity model-based self-learning fuzzy logic controller," *Proceedings of the IEEE Conference on Control Applications CCA 98*, Trieste, pp. 643–647, 1998.
61. Kovačić, Z., Petik, V., Reichenbach, T., and Bogdan, S., "Robust self-learning fuzzy logic servo control with neural network-based load compensator," *CD-ROM Proceedings of the 3rd IMACS/IEEE International Conference on Circuits, Systems Communications and Computers*, Athens, 1999.
62. Kovačić, Z., Bogdan, S., and Reichenbach, T., "Nonlinear position control by using multiple position dependent self-organizing fuzzy logic controllers," *6th IFAC Symposium on Robot Control SYROCO'00*, Vienna, Austria, pp. 229–233, 2000.
63. Kovačić, Z., Cupec, R., and Bogdan, S., "A servo positioning by using model reference adaptive fuzzy controller," *CD-ROM Preprints of the 1st IFAC/IEEE Symposium on System Structure and Control 2001*, Prague, 2001.

6 Fuzzy Controllers as MATLAB® Superblocks

The usage of simulation software packages for modeling, simulation, and optimization of control systems has become a part of regular engineering practice. Recently added features of such software packages like a possibility to generate real-time executable code directly from simulation models enabled shorter development times and faster validation of new control solutions. The solutions developed with the world standard software packages like MATLAB, Matrix$_x$, or Mathematica, become transparent to a large number of users. Respecting the fact that MATLAB+Simulink is one of the most popular and worldwide used simulation software packages, in this chapter we give several worked out examples of fuzzy control systems whose simulation models are built by using MATLAB. Since MATLAB contains the Fuzzy Logic Toolbox (FLT) that allows the designer to create and test new fuzzy control designs, we give a short description of basic features of the MATLAB Fuzzy Logic Toolbox needed for the successful usage of the tool [1].

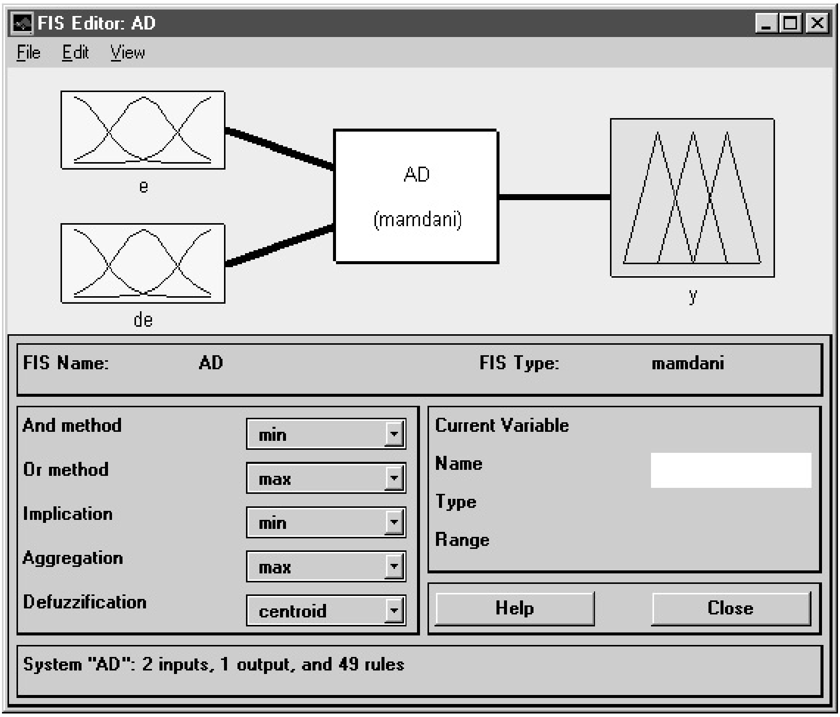
In order to make it possible for the readers of this book to test several fuzzy controllers presented in this book by themselves, the hybrid fuzzy controller and two types of self-organizing fuzzy controllers (i.e., model tracking error polynomial-based and sensitivity model-based) are implemented as ready-to-use MATLAB+Simulink function blocks that are compatible with the FLT standards. Since new versions of MATLAB are expected to appear, updated versions of function blocks will be available to interested readers via the World Wide Web [2].

6.1 FEATURES OF MATLAB FUZZY LOGIC TOOLBOX

MATLAB FLT is a program tool for working with fuzzy logic systems. FLT contains four main tools: FIS (Fuzzy Inference System) Editor, Membership Functions Editor, Rule Editor, and Rule Viewer.

6.1.1 FIS Editor

The FIS Editor is a tool in FLT, where the number of inputs, names of input, and output variables, as well as the type of fuzzy controller (Mamdani or Sugeno) is determined (Figure 6.1). Selection of a Mamdani-type fuzzy controller assumes that linguistic values of the output variable are regular fuzzy sets. Selection of a Sugeno-type fuzzy controller assumes that linguistic values of the output variable are singletons.



FIS Name: AD

FIS Type: mamdani

And method: min

Or method: max

Implication: min

Aggregation: max

Defuzzification: centroid

System "AD": 2 inputs, 1 output, and 49 rules

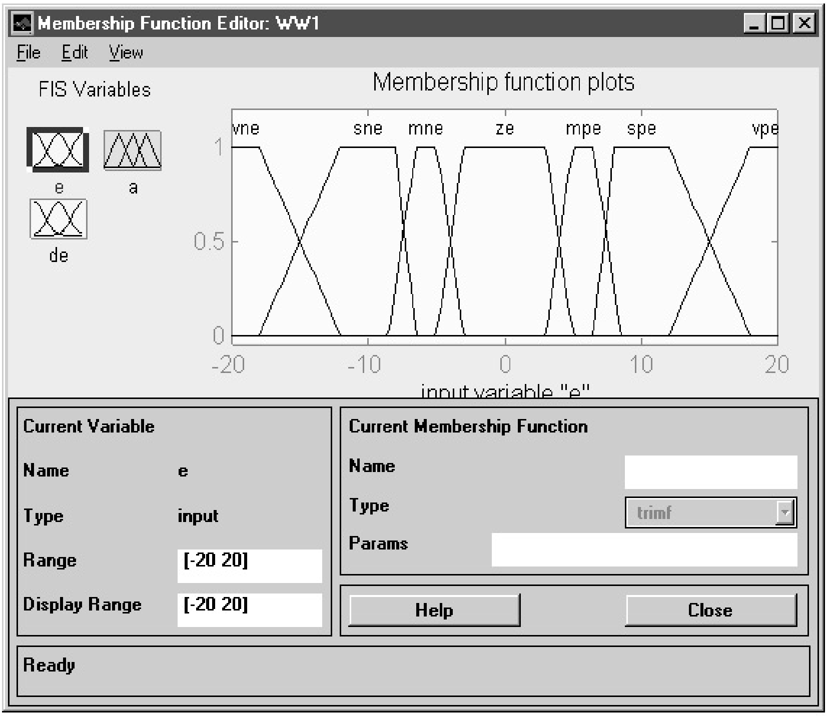
FIGURE 6.1 FIS editor.

The FIS editor also serves for selection of a fuzzy inference method and it enables the choice of aggregation operator that gives a different type of a compositional output fuzzy set. The *max*-type aggregation leads toward the center of area (COA) defuzzification, while the *sum*-type aggregation leads toward the center of gravity (COG) defuzzification (the reader can refer to the discussion about different defuzzification methods in Section 2.3.2).

The fuzzy controller structure is stored in a so-called FIS matrix. The connection between FLT and Simulink may be accomplished by reading the FIS matrix from the MATLAB command window: >*w* = *readfis*(‘*name_of_file.fis*’), followed by a list of all FIS matrix elements, or by saving the fuzzy controller structure created with FIS editor into the MATLAB workspace.

6.1.2 Membership Function Editor

Membership function editor shown in Figure 6.2 for an example of a DISO fuzzy controller enables definition of membership function forms for the inputs and the output, and allows settings of boundary parameters for each membership function. The designer can choose from eleven standard functions (triangle, trapeze, bell,



Current Variable

Name e

Type input

Range [-20 20]

Display Range [-20 20]

Current Membership Function

Name

Type trimf

Params

Ready

FIGURE 6.2 Membership function editor.

Z-shape, Π-shape, S-shape, Gauss, etc., see Figure 6.3). The tool allows definition of the functions range, name of functions, and the range of display. Also, new membership functions can be added.

6.1.3 Rule Editor

Rule Editor shown in Figure 6.4 serves for insertion of new fuzzy rules. Rules can be inserted in forms of text, symbols, or indices. Before insertion of rules, fuzzy membership functions must be defined.

6.1.4 Rule Viewer

Rule viewer shown in Figure 6.5 is a tool that provides a more detailed insight into the fuzzy inference process of a DISO fuzzy controller. Each row represents one fuzzy rule containing two input membership functions and one output membership function. In this way all fuzzy rules create a table with three corresponding columns. Actual values of fuzzy controller inputs, depending on selected features

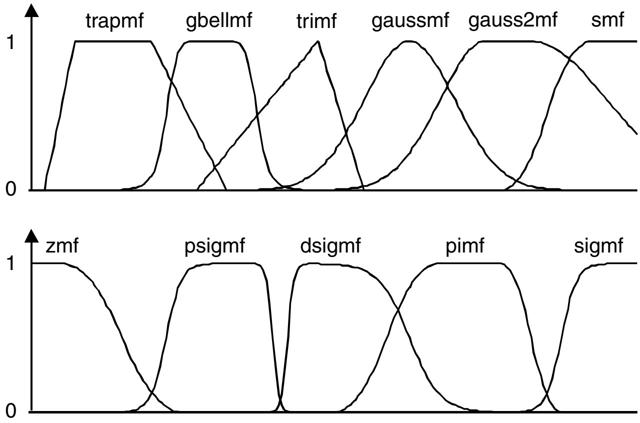
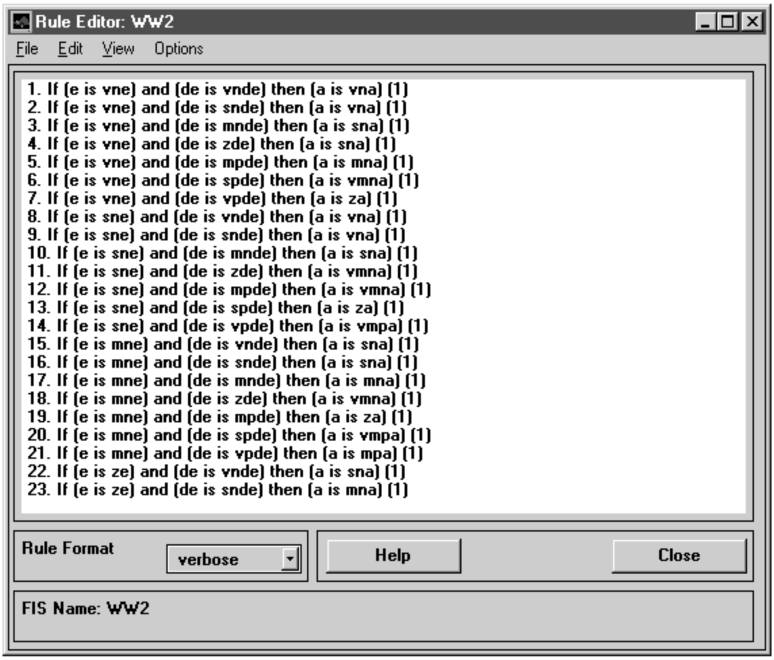


FIGURE 6.3 Forms of fuzzy membership functions in FLT.



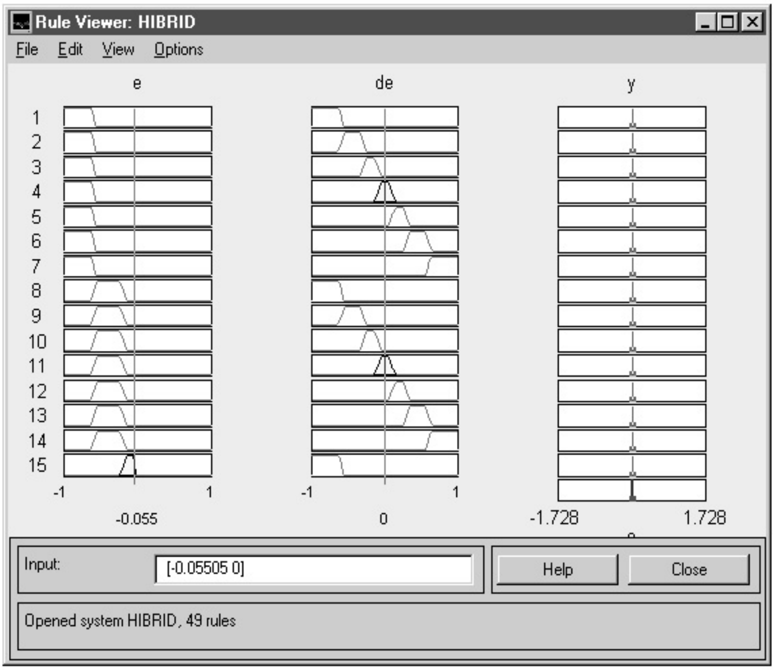
```
1. If (e is vne) and (de is vnde) then (a is vna) (1)
2. If (e is vne) and (de is snde) then (a is vna) (1)
3. If (e is vne) and (de is mnde) then (a is sna) (1)
4. If (e is vne) and (de is zde) then (a is sna) (1)
5. If (e is vne) and (de is mpde) then (a is mna) (1)
6. If (e is vne) and (de is spde) then (a is vmna) (1)
7. If (e is vne) and (de is vpde) then (a is za) (1)
8. If (e is sne) and (de is vnde) then (a is vna) (1)
9. If (e is sne) and (de is snde) then (a is vna) (1)
10. If (e is sne) and (de is mnde) then (a is sna) (1)
11. If (e is sne) and (de is zde) then (a is vmna) (1)
12. If (e is sne) and (de is mpde) then (a is vmna) (1)
13. If (e is sne) and (de is spde) then (a is za) (1)
14. If (e is sne) and (de is vpde) then (a is vmpa) (1)
15. If (e is mne) and (de is vnde) then (a is sna) (1)
16. If (e is mne) and (de is snde) then (a is sna) (1)
17. If (e is mne) and (de is mnde) then (a is mna) (1)
18. If (e is mne) and (de is zde) then (a is vmna) (1)
19. If (e is mne) and (de is mpde) then (a is za) (1)
20. If (e is mne) and (de is spde) then (a is vmpa) (1)
21. If (e is mne) and (de is vpde) then (a is mpa) (1)
22. If (e is ze) and (de is vnde) then (a is sna) (1)
23. If (e is ze) and (de is snde) then (a is mna) (1)
```

Rule Format: verbose

FIS Name: WW2

FIGURE 6.4 Rule editor.

of fuzzy inference, yield different contributions to the crisp fuzzy controller output, which can be simultaneously registered in the Rule Viewer window. This tool allows the designer to make analysis of the inference system and decide about the controller parameter settings.



Input: [-0.05505 0]

Opened system HIBRID, 49 rules

FIGURE 6.5 Rule Viewer.

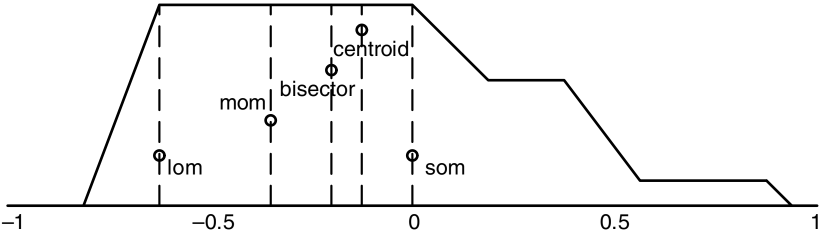


FIGURE 6.6 Defuzzification methods in FLT.

6.1.5 Defuzzification Methods in FLT

The FLT provides a set of five defuzzification methods to choose from: *centroid*, *mom*, *lom*, *som*, and *bisector* (see Figure 6.6). Moreover, FLT allows the designer to create his or her own defuzzification methods. For the Takagi–Sugeno type of DISO controllers the *centroid* defuzzification is the most appropriate. The outcome of centroid defuzzification will depend on the selected type of aggregation. The usage of *max*-type aggregation leads to the COA defuzzification method described with (2.20), while the *sum*-type aggregation leads to the COG defuzzification method described with (2.21).

TABLE 6.1
Several FLT Commands

Command	Command description
showfis(w)	Display the whole structure of the fuzzy controller
plotmf(w, 'input', 1)	Display all membership functions for input variable 1
getfis(w)	Get fuzzy system properties
plotfis(w)	Display FIS input–output diagram
gensurf(w)	Generate FIS output surface
deffuzzdm	Defuzzification methods
rmmf	Remove membership functions from FIS matrix
newfis	Create new FIS document
writefis	Save FIS document
rmvar	Remove variable from FIS matrix

6.1.6 FLT Commands

The FLT stores fuzzy controller data into a $*$.*fis* file. All parameters can be modified simply by editing the $*$.*fis* file. Several FLT commands that belong to the core of the FLT command set are given in Table 6.1. The reader can find more about FLT commands in Reference 1.

6.2 Hybrid Fuzzy Controller Super-Block for Matlab

The detailed description of hybrid fuzzy controllers is given in Section 4.1. They usually have several operating modes and therefore switching between operating modes occurs. In order to prevent a possible chattering problem, implementation of a hybrid fuzzy controller as a MATLAB super-block must provide bumpless switching of modes.

Figure 6.7 shows a structure of a hybrid fuzzy controller super-block designed for angular speed control of a vector-controlled chopper-fed PMSM drive described in detail in Example 4.1. The fuzzy controller function block contained within the hybrid fuzzy controller super-block is a standard FLT function block. One can notice the actual values of fuzzy controller inputs scaling factors, as well as the actual boundary values of inputs zero subsets ZE and ZDE, respectively. These values dictate the switching of operating modes. In general, they should be set to match the dynamic characteristics of the target control system as much as possible.

The bumpless transition between operating modes is accomplished in a way that the crisp controller output value from the preceding operating mode becomes the initial controller output value for the current operating mode.

Figure 6.8 shows the simulation block scheme of the hybrid fuzzy angular speed control system of the PMSM drive described in Example 4.1. The values of block parameters are related to the rated values of system parameters.

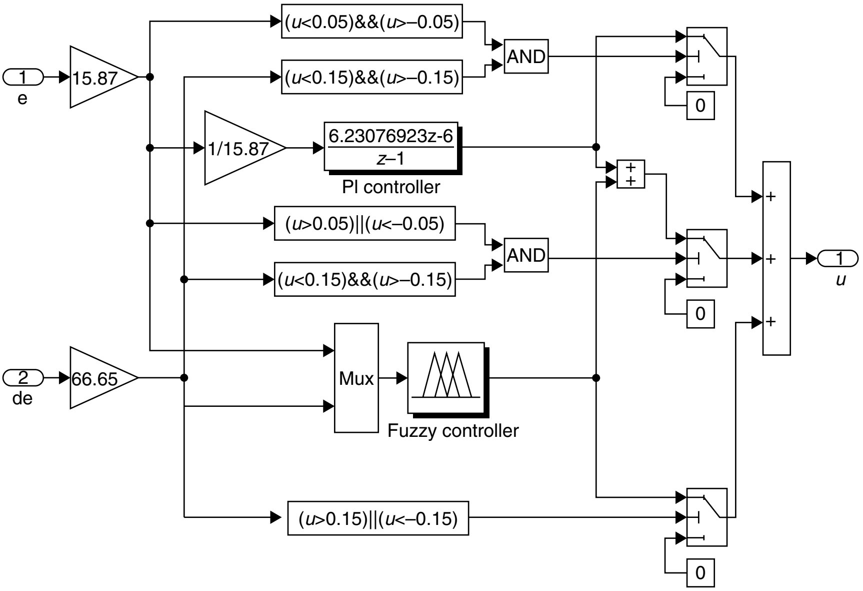


FIGURE 6.7 The structure of the hybrid fuzzy controller superblock for MATLAB+Simulink.

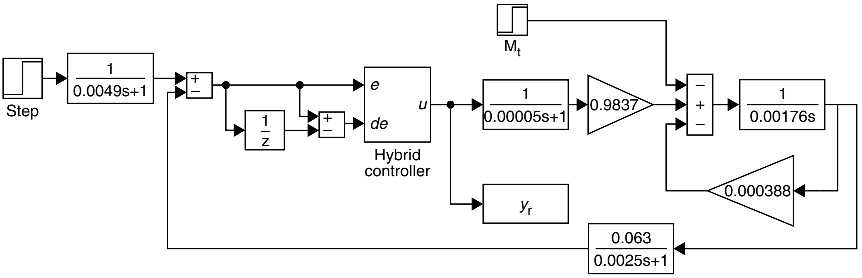


FIGURE 6.8 The simulation scheme of the studied hybrid fuzzy angular speed control system.

Figure 6.9 and Figure 6.10 show the measured angular speed and hybrid fuzzy controller output responses in the case of the stepwise change of the reference input. Figure 6.11 shows a time flow of controller output signals obtained by following the sequence of induced operating mode changes. According to expectations, at the beginning of transient response only the fuzzy controller is active, then fuzzy and PI controllers work together, and eventually, the PI controller takes over control in the steady state. The analysis of responses proves that transitions from one operating mode to another are smooth, without abrupt changes in the controller output signal.

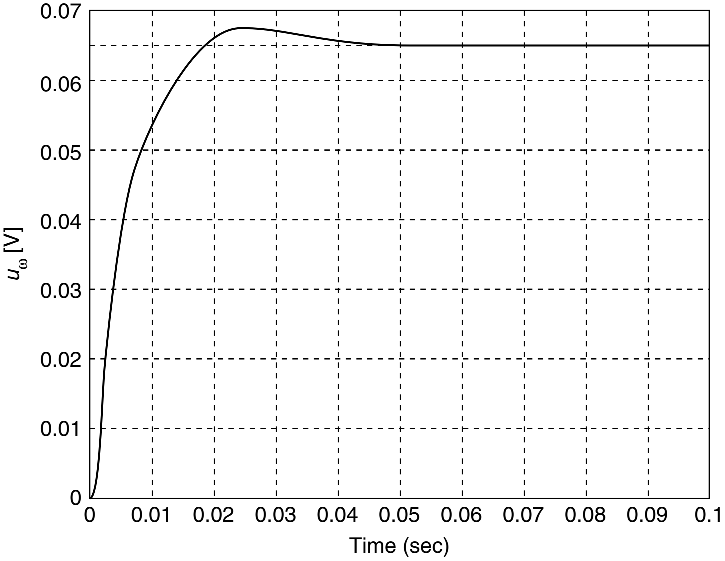


FIGURE 6.9 The measured angular speed response of a hybrid fuzzy control system.

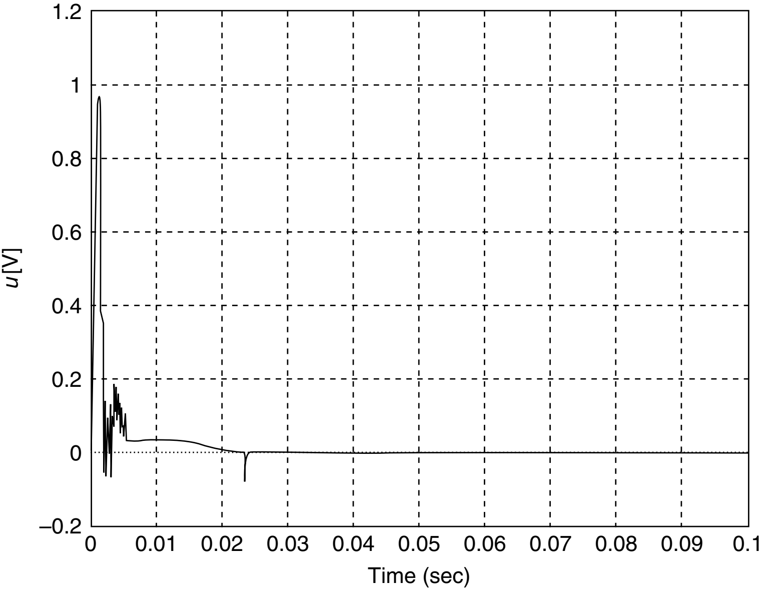


FIGURE 6.10 The hybrid fuzzy controller output response.

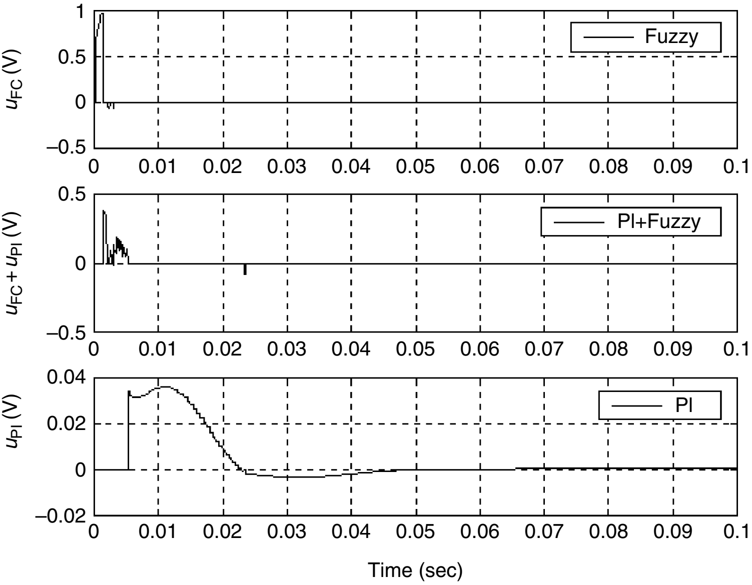


FIGURE 6.11 The hybrid fuzzy controller output response during the change of operating modes.

6.3 POLYNOMIAL-BASED PSLFLC MATLAB SUPER-BLOCK

In this section, we demonstrate the performance of a PD type polynomial-based self-learning fuzzy logic controller (PSLFLC) described in Section 5.2, which has been implemented as a function block for MATLAB+Simulink [3]. The learning laws (5.49) and (5.57) implemented in the block are based on the usage of a second-order reference model (3.28) and a third-degree model tracking error polynomial. The emphasis in this chapter is put on the description of function block parameters and the way the block is used.

The concept of the PSLFLC block allows control of unknown inherently stable static and astatic nonlinear systems if the desired closed-loop behavior can be represented with a linear second-order reference model. The basic structure of the PSLFLC block contains a PD-type fuzzy controller and a feedforward control element (Figure 6.12):

$$u(k) = \Gamma\,[e(k), \mathrm{d}e(k), \boldsymbol{\lambda}] + \xi_{\mathrm{s}} k_3 \cdot u_{\mathrm{r}}(k) = \sum_{i=1}^{r} A_i \cdot \phi_i\,[e(k), \mathrm{d}e(k)] + \xi_{\mathrm{s}} k_3 \cdot u_{\mathrm{r}}(k) \tag{6.1}$$

where ξ_{s} is a Boolean type parameter ($\xi_{\mathrm{s}} = 1$ denotes a static type of system).

The user must be aware that most of the FIS matrix parameters including input and output universes of discourses, as well as the number, size, and shape of the fuzzy input sets have been selected and determined before starting using the block.

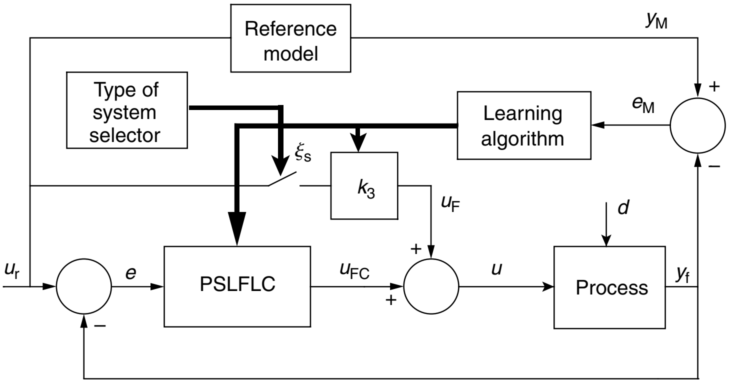
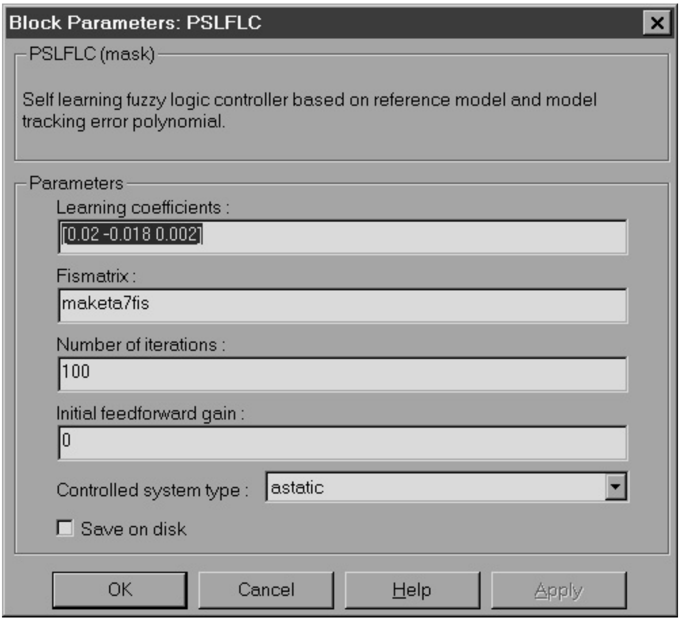


FIGURE 6.12 The structure of a system with PSLFLC. (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *KoREMA 11th Intl. Conf. Electr. Drives Power Electr.*, 93–98, 2000. With permission.)


Block Parameters: PSLFLC

PSLFLC (mask)

Self learning fuzzy logic controller based on reference model and model tracking error polynomial.

Parameters

Learning coefficients : [0.02 -0.018 0.002]

Fismatrix : maketa7fis

Number of iterations : 100

Initial feedforward gain : 0

Controlled system type : astatic

☐ Save on disk

FIGURE 6.13 The PSLFLC function block mask. (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *KoREMA 11th Intl. Conf. Electr. Drives Power Electr.*, 93–98, 2000. With permission.)

The PSLFLC function block is written as a CMEX S-function stored as the *pslflc.dll* file. The block has a mask in the form of a standard dialog box (Figure 6.13). The user can set several parameters displayed in Table 6.2 that influence operation of the block.

TABLE 6.2
PSLFLC Parameters That User Can Set

PSLFLC parameter	Parameter description
Learning coefficients (type: ⟨*real*⟩)	Values of learning coefficients γ_1, γ_2, γ_3 that satisfy stability conditions (5.59)
FISMATRIX (type: ⟨*name*⟩)	Name of the FIS matrix. The FIS matrix has a form of a standard Fuzzy Logic Toolbox v.2.0 fismatrix [1]
Number of iterations (type: ⟨*real*⟩)	Maximum number of learning iterations. Value = 0 means that learning is disabled
Initial feedforward gain value (type: ⟨*real*⟩)	If the controlled system is static ($\xi_s = 1$), then an initial value of the feedforward gain k_3 (default value = 0) can be defined. More accurate k_3 will speed up the learning process
Controlled system type (type: ⟨*options dialog*⟩)	Choice of system type with two options (i) static or (ii) astatic
Save on disk (type: ⟨*check-box*⟩)	Checking this option enables the creation of files which store particularly interesting results of simulation for a subsequent analysis

The internal structure of the PSLFLC super-block is shown in Figure 6.14. It can be seen that, besides the PSLFLC function block, the super-block contains also blocks for scaling of fuzzy controller inputs and a reference model block.

The reference model block represents a unity gain second-order system implemented according to the reference model Equation (3.28). The reference model block has a mask (dialog box), which allows the user to define essential parameters: peak time (time of first maximum) t_m, overshoot in response σ_m, and the value of control interval. After ZOH-discretization, the resulting coefficients of the reference model equation (3.28) are available for user's convenience at the reference model block output.

Let us demonstrate the effectiveness of the PSLFLC super-block on a model of a PMSM drive taken from the MATLAB–Simulink library of ready-to-use demo examples (the directory is denoted in Figure 6.15). The blocks in the model have been taken from the Power Systems Toolbox that is also the part of the MATLAB–Simulink software package. This model contains an outer angular speed control loop and an inner stator current control loop with a PWM stator current controller and a chopper-fed PMSM.

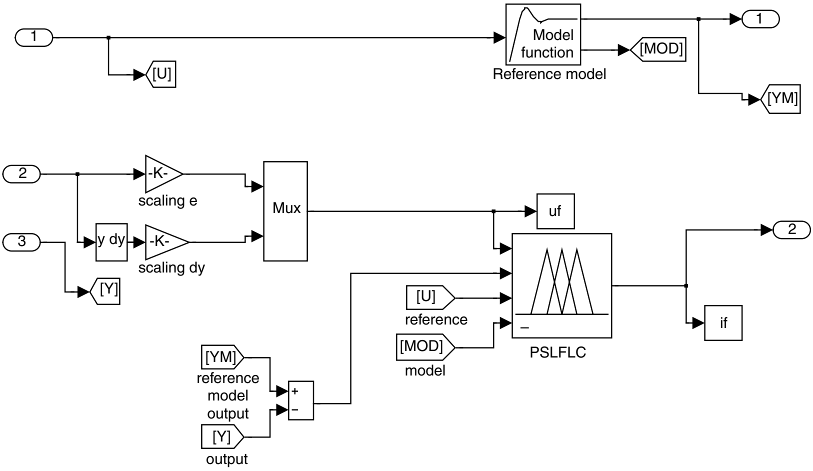


FIGURE 6.14 The structure of the PSLFLC super-block. (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *KoREMA 11th Intl. Conf. Electr. Drives Power Electr.*, 93–98, 2000. With permission.)

The intention of this example is not to discuss the properties of the MATLAB PMSM drive demo model, but to point out that this model is nonlinear and originally controlled by a linear PI angular speed controller. In the new simulation scheme shown in Figure 6.15, the original PI controller block has been replaced with the nonlinear PSLFLC super-block.

Seven linearly distributed triangular membership functions have been determined for both PSLFLC inputs $e(k)$ and $\Delta e(k)$: NLE, NME, NSE, ZE, PSE, PME, PLE and NLDE, NMDE, NSDE, ZDE, PSDE, PMDE, PLDE, respectively. Simulations were performed with the simulated change of the angular speed reference input $\Delta u_\mathrm{r} = \pm 150\ \mathrm{sec}^{-1}$. Fuzzy input scaling coefficients were estimated at $k_\mathrm{e} = 1/300$, $k_{\Delta\mathrm{e}} = 1/300$. Learning coefficients were set to the following values: $\gamma_1 = 0.02$, $\gamma_2 = -0.018$, $\gamma_3 = 0.002$. The initial feedforward gain k_3 value was set to zero. Desired performance indices of the reference model were as follows: $\sigma_\mathrm{m} = 0.5\%$, $t_\mathrm{m} = 0.01$ sec (sampling interval T_d is 0.0001 sec).

Figure 6.16 and Figure 6.17 show the reference input, system output, and reference model responses obtained after two and after six iterations of learning, respectively. It may be seen that after completion of learning the system follows the reference model very closely.

Figure 6.18 and Figure 6.19 show the reference model tracking error responses at the beginning and the end of learning, respectively. In the initial phase, the tracking error exceeds 100 sec^{-1}, while in the end it drops below 30 sec^{-1} (less than 10% of the imposed change of reference input).

Figure 6.20 and Figure 6.21 show the PSLFLC output responses at the beginning and at the end of learning, respectively. The PSLFLC output has a very acceptable nonoscillatory form, which clearly reflects the nonlinear character of the target system.

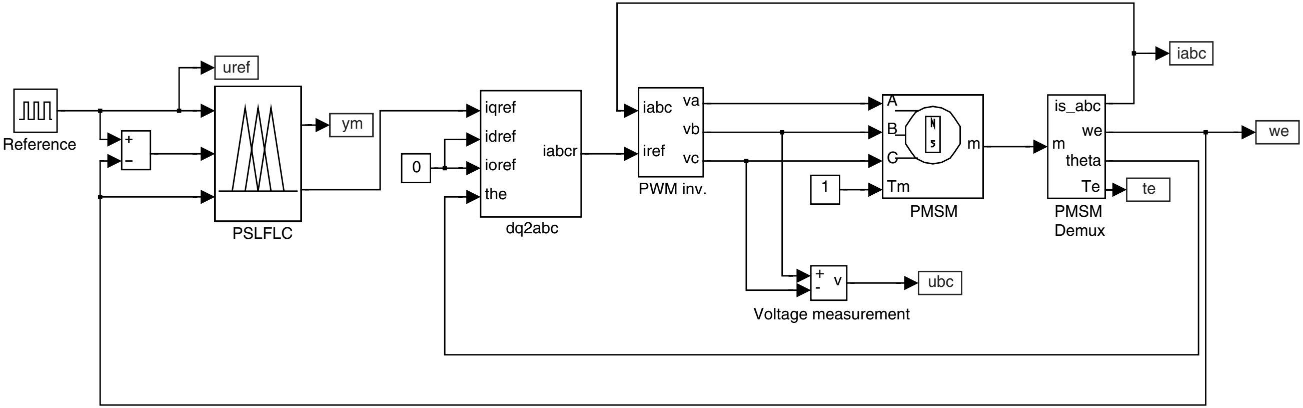


FIGURE 6.15 The MATLAB simulation scheme of the closed-loop PMSM angular speed control system (directory: \matlab\toolbox\ powersys\ powerdemo\psbpmmotor.mdl). (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *KoREMA 11th Intl. Conf. Electr. Drives Power Electr.*, 93–98, 2000. With permission.)

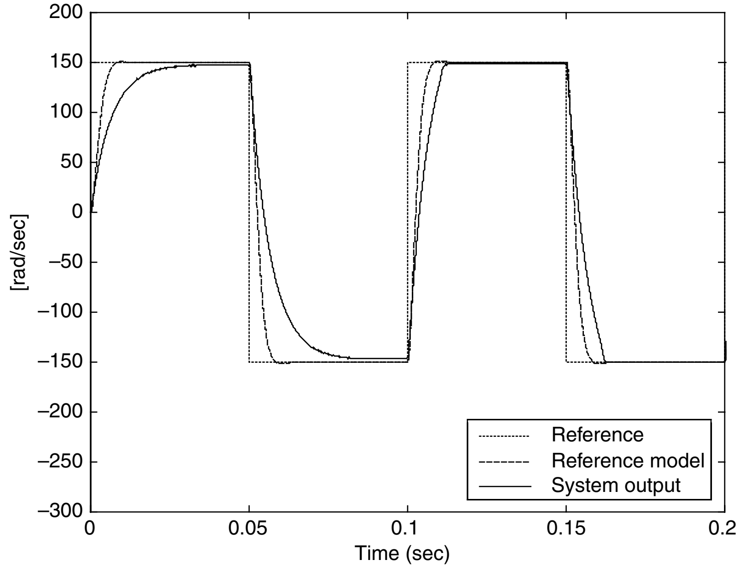


FIGURE 6.16 Start of learning: the reference input, system output, and reference model output responses of the PMSM angular speed control system. (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *KoREMA 11th Intl. Conf. Electr. Drives Power Electr.*, 93–98, 2000. With permission.)

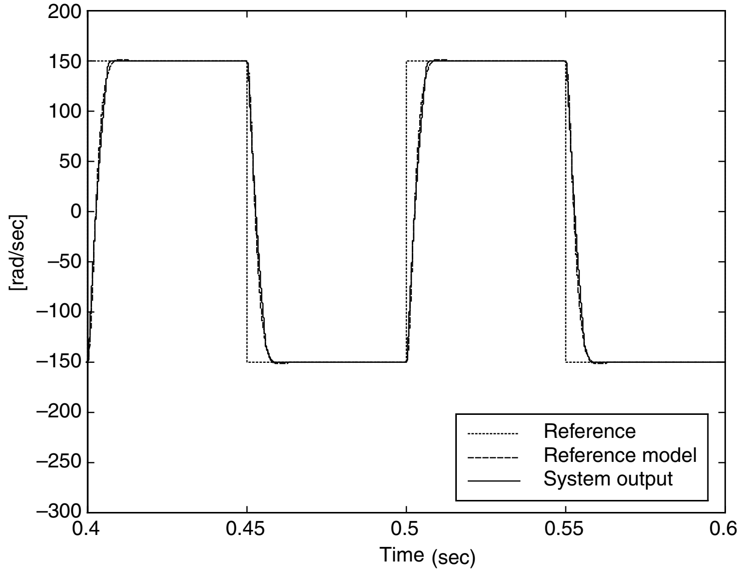


FIGURE 6.17 End of learning: the reference input, system output, and reference model output responses of the PMSM angular speed control system. (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *KoREMA 11th Intl. Conf. Electr. Drives Power Electr.*, 93–98, 2000. With permission.)

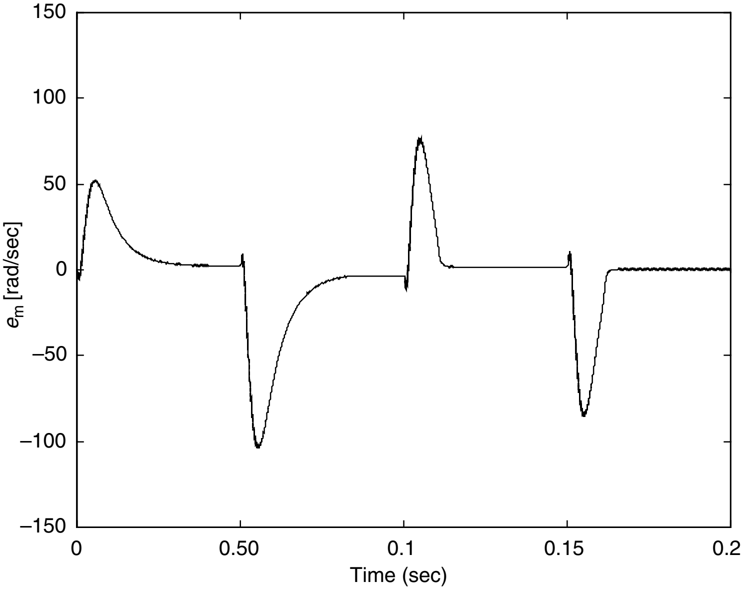


FIGURE 6.18 Start of learning: the reference model tracking error responses. (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *KoREMA 11th Intl. Conf. Electr. Drives Power Electr.*, 93–98, 2000. With permission.)

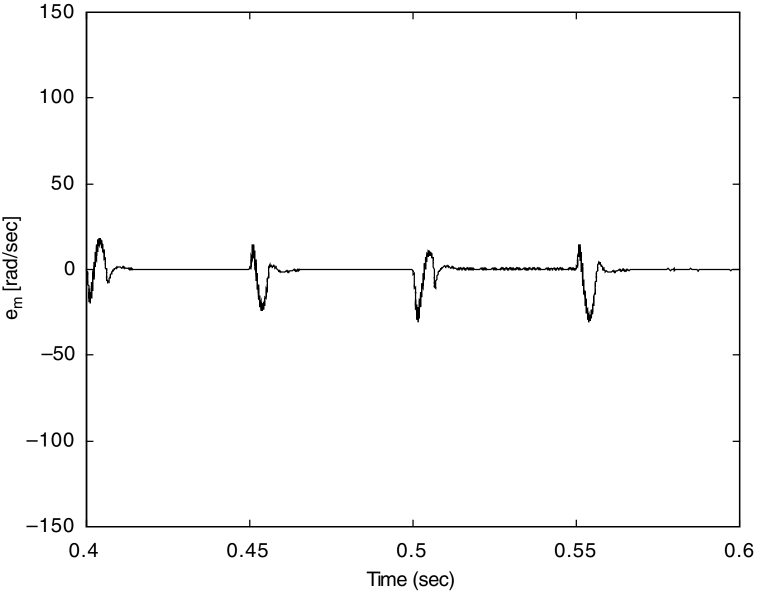


FIGURE 6.19 End of learning: reference model tracking error responses. (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *KoREMA 11th Intl. Conf. Electr. Drives Power Electr.*, 93–98, 2000. With permission.)

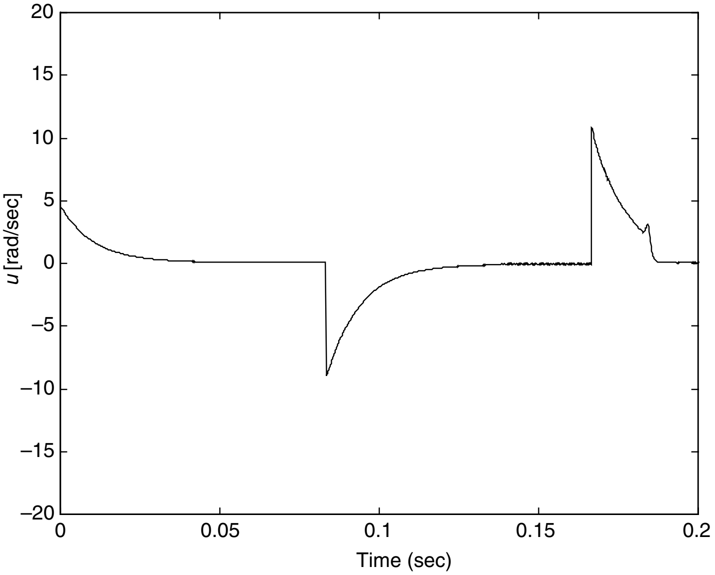


FIGURE 6.20 Start of learning: the PSLFLC output responses. (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *KoREMA 11th Intl. Conf. Electr. Drives Power Electr.*, 93–98, 2000. With permission.)

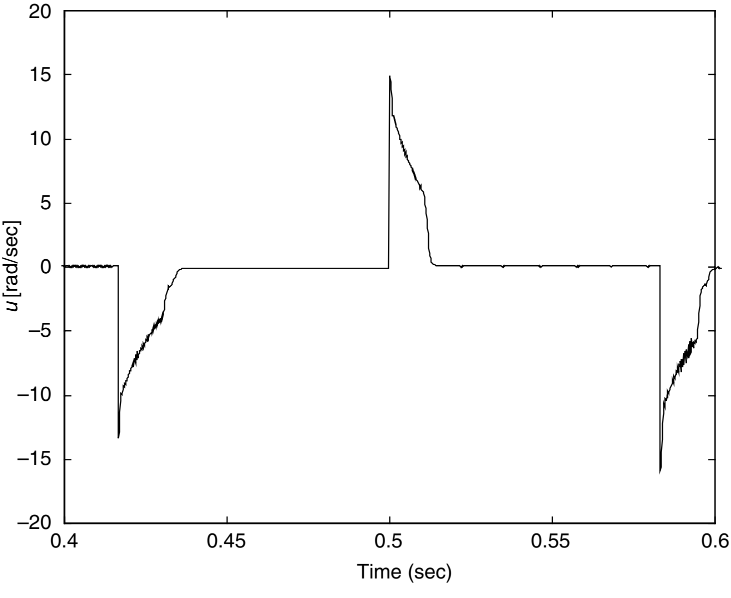


FIGURE 6.21 End of learning: the PSLFLC output responses. (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *KoREMA 11th Intl. Conf. Electr. Drives Power Electr.*, 93–98, 2000. With permission.)

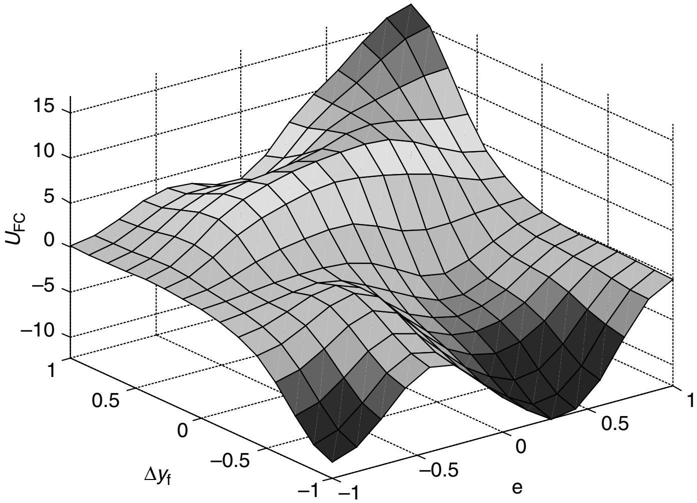


FIGURE 6.22 End of learning: the graph of the PSLFLC control surface. (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *KoREMA 11th Intl. Conf. Electr. Drives Power Electr.*, 93–98, 2000. With permission.)

Figure 6.22 shows the PSLFLC control surface that was generated after completion of self-organization. It gives a further insight into the nonlinear character of the controller.

6.4 SENSITIVITY MODEL-BASED SLFLC MATLAB SUPER-BLOCK

In this section, we demonstrate the performance of a PD type self-learning fuzzy logic controller (SLFLC) described in Section 5.3, implemented as a function block for MATLAB+Simulink [4]. The learning laws (5.88) and (5.89) implemented in the block are based on the usage of a second-order reference model (3.28) and a sensitivity model built with respect to fuzzy controller parameters. Similarly as for the PSLFLC block, the emphasis is put on the description of function block parameters and the way the block is used.

The concept of the SLFLC block allows control of unknown inherently stable static and astatic nonlinear systems providing that a desired closed-loop system behavior can be represented with a linear second-order reference model. The fuzzy controller has two inputs $e(k)$ and $\Delta y_f(k)$, and one output $u_{FC}(k)$. The basic structure of the SLFLC block contains a PD-type fuzzy controller, a feedforward control element and a P controller (Figure 6.23):

$$\begin{aligned} u(k) &= \Gamma[e(k), \Delta y_f(k), \boldsymbol{\lambda}] + \xi_s k_3 \cdot u_r(k) + (1-\xi_s)k_p e(k) \\ &= \sum_{i=1}^{r} A_i \cdot \phi_i[e(k), \Delta y_f(k)] + \xi_s k_3 \cdot u_r(k) + (1-\xi_s)k_p e(k) \end{aligned} \tag{6.2}$$

whereas in (6.1), ξ_s refers to the type of system (1 — static, 0 — astatic).

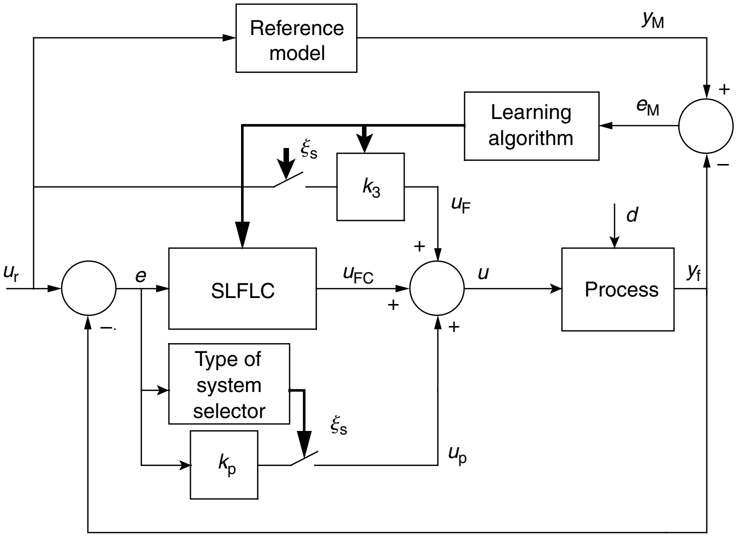


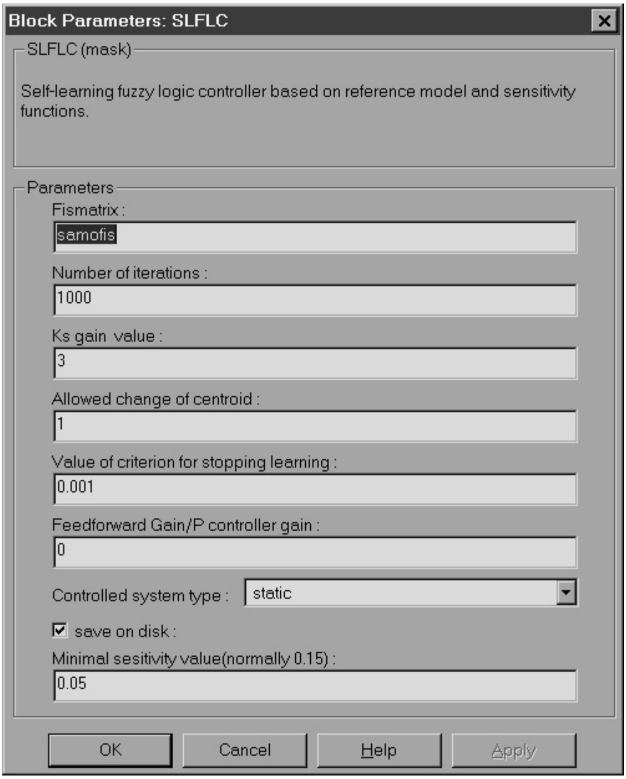
FIGURE 6.23 The structure of a system with SLFLC. (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *8th IEEE Mediterr. Conf. Contr. Autom. MED'00*, TC-2.5, 2000. With permission.)

Quite the same as with the PSLFLC block, the SLFLC FIS matrix is fully compatible with the standard FLT FIS matrix form. Also, most of SLFLC FIS matrix parameters (input and output universes of discourse, size, and shape of fuzzy input sets) are determined before using the block. Because of the way how self-organization is performed, input fuzzy sets take only a differentiable Gaussian form (5.74). An α-cut operation is applied to all input fuzzy sets ($\alpha = 0.05$). Since output fuzzy sets are singletons, a COG defuzzification is used.

The SLFLC function block is written as a CMEX S-function stored as the *slflc.dll* file. The block has a mask in the form of a standard dialog box (Figure 6.24). The user can set several parameters displayed in Table 6.3 that influence operation of the block.

The internal structure of the SLFLC super-block is shown in Figure 6.25. The structure completely coincides with the structure of the PSLFLC super-block shown in Figure 6.14, only the PSLFLC function block is replaced with the SLFLC function block.

Let us demonstrate the effectiveness of the SLFLC super-block on a simulation model of a closed-loop engine speed control system taken from the MATLAB–Simulink library of ready-to-use demo examples (the directory is denoted in Figure 6.26). The control of an engine speed is based on the control of a throttle-valve opening. We shall not discuss the properties of the model, which is nonlinear and originally controlled by a linear PI controller. A block of the hybrid fuzzy engine speed controller is shown in Figure 6.27. It can be seen that the SLFLC super-block is acting in parallel with the existing PI controller block. This means that the SLFLC super-block acts like an adaptation mechanism, having a goal to improve the overall system performance.



Block Parameters: SLFLC

SLFLC (mask)

Self-learning fuzzy logic controller based on reference model and sensitivity functions.

Parameters

Fismatrix : samofis

Number of iterations : 1000

Ks gain value : 3

Allowed change of centroid : 1

Value of criterion for stopping learning : 0.001

Feedforward Gain/P controller gain : 0

Controlled system type : static

☑ save on disk :

Minimal sesitivity value(normally 0.15) : 0.05

FIGURE 6.24 The mask of the function block SLFLC. (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *8th IEEE Mediterr. Conf. Contr. Autom. MED'00*, TC-2.5, 2000. With permission.)

Seven Gaussian linearly distributed membership functions have been determined for both SLFLC inputs $e(k)$ and $\Delta y_f(k)$: NLE, NME, NSE, ZE, PSE, PME, PLE and NLDYF, NMDYF, NSDYF, ZDYF, PSDYF, PMDYF, PLDYF, respectively. Simulations were performed with a simulated change of the engine speed reference input $\Delta u_r = \pm 1000$ rpm. Fuzzy input scaling coefficients were estimated at $k_e = 1/100$, $k_{\Delta y_f} = 1/300$. A minimal value of the sensitivity function has been set to $\beta_{\min} = 0.05$, the limit for an allowed change of singletons has been set to $\chi = 1$, while gain coefficient K_s has been set to 3.0. The proportional gain k_p has been set to zero, because a PI controller is already in the loop. Desired performance indices of the reference model were as follows: $\sigma_m = 0.5\%$, $t_m = 0.5$ sec (sampling interval T_d is 0.01 sec).

Figure 6.28 and Figure 6.29 show the reference input, system output, and reference model responses obtained after three and after thirteen iterations of learning, respectively. It can be seen that after completion of learning the system follows the reference model much better than only with the PI controller. The original system with a PI controller reaches the steady state after four seconds, while the hybrid self-learning fuzzy controller enforces system to reach the steady state in less than

TABLE 6.3
SLFLC Parameters That User Can Set

SLFLC parameter	Parameter description
FISMATRIX (type: ⟨*name*⟩)	Name of the FIS matrix. The FIS matrix has a form of a standard Fuzzy Logic Toolbox v.2.0 fismatrix [1]
Number of iterations (type: ⟨*real*⟩)	Maximum number of learning iterations. Value = 0 means that learning is disabled (only feedforward or P control is active)
K_S gain value (type: ⟨*real*⟩)	Gain value determining the relation between the reference model and the process approximation. Larger K_S will make learning slower, but smoother and more stable
Allowed change of centroid (type: ⟨*real*⟩ = χ)	Setting the value of χ according to the following law: $K_\chi = \mathrm{e}^{(\chi \lvert e_M \rvert)}$, defines constraints on singleton changes in one learning iteration, where K_χ denotes a value, which divides calculated changes of singletons. Larger K_χ makes learning slower. This parameter is partly supplementary to the parameter K_S
Value of criterion for stopping learning (type: ⟨*real*⟩)	Threshold value δ_0 of the IAE criterion (5.91) that serves for stopping learning (for prevention of over-learning). This criterion is recalculated in each learning iteration
Initial feedforward gain/P controller gain value (type: ⟨*real*⟩)	For the static type of a controlled system the feedforward gain k_3 is defined. For the astatic type of a controlled system the proportional gain k_p is defined. If $k_3 = 0$, then k_3 will change according to the learning law (5.89) and contribute to the crisp output of the SLFLC. If $k_p = 0$, then only fuzzy logic control is active
Controlled system type (type: ⟨*options dialog*⟩)	Choice of system type with two options (i) static or (ii) astatic. This reflects the interpretation of the preceding SLFLC parameter (k_3 or k_p)
Save on disk (type: ⟨*check-box*⟩)	Checking this option enables the creation of files, which store particularly interesting results of simulation for a subsequent analysis
Minimal sensitivity value (type: ⟨*real*⟩)	Definition of a threshold value $\beta_{\min}$, which separates sensitivity functions that contribute to changes of singletons from those sensitivity functions (whose maximal values are below the threshold) that do not contribute

a second. This proves that addition of the SLFLC in parallel to the PI controller has contributed to a much better quality of the system response.

Figure 6.30 and Figure 6.31 show the hybrid self-learning fuzzy controller outputs at the beginning (only PI controller is active) and at the end of learning (both controllers are active), respectively. The final form of the hybrid self-learning fuzzy controller output clearly indicates that the target system is very nonlinear. Figure 6.32 shows the SLFLC control surface that was generated after completion of learning. Its form clearly shows that the resulting input–output mapping function that copes with a nonlinear control process has also a very nonlinear character.

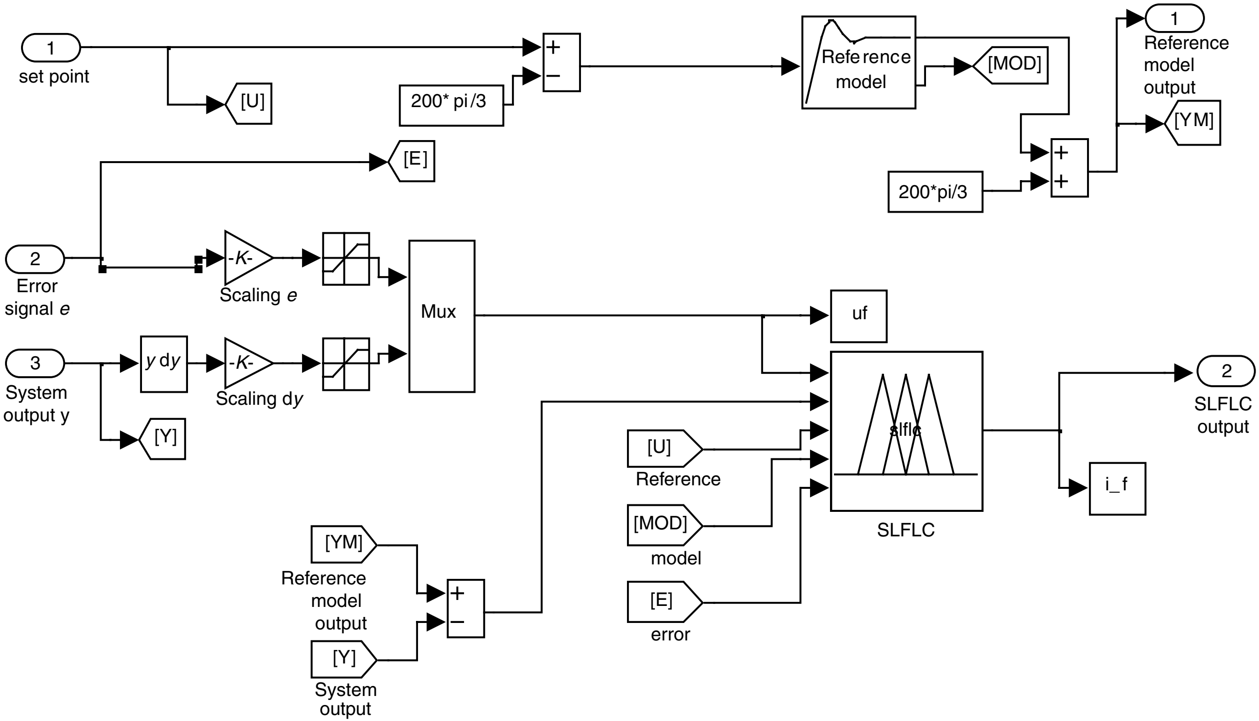


FIGURE 6.25 The structure of the SLFLC super-block. (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *8th IEEE Mediterr. Conf. Contr. Autom. MED'00*, TC-2.5, 2000. With permission.)

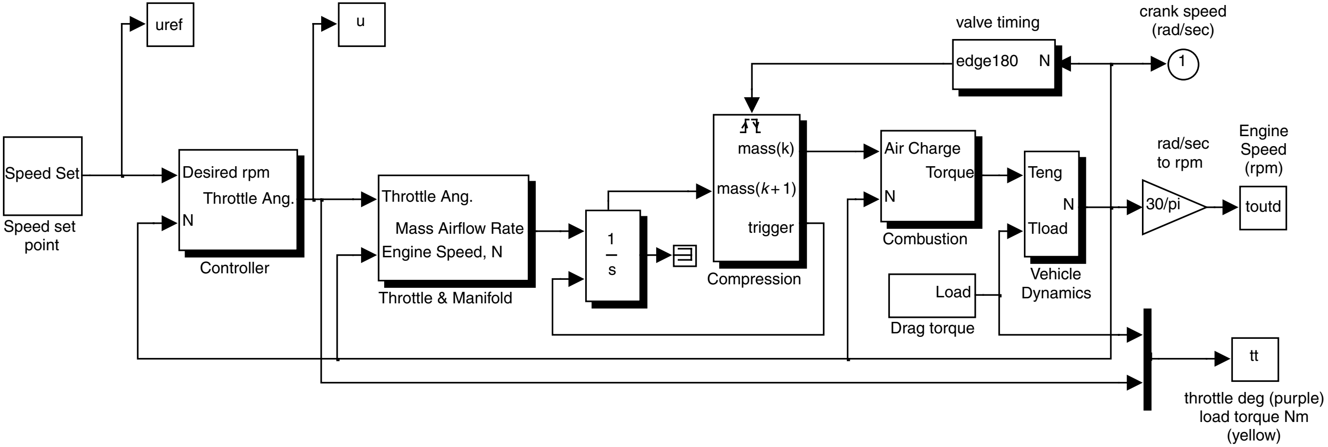


FIGURE 6.26 The MATLAB simulation scheme of the closed-loop engine speed control system (directory: \matlab\toolbox\simulink \simdemos\engine.mdl). (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *8th IEEE Mediterr. Conf. Contr. Autom. MED'00*, TC-2.5, 2000. With permission.)

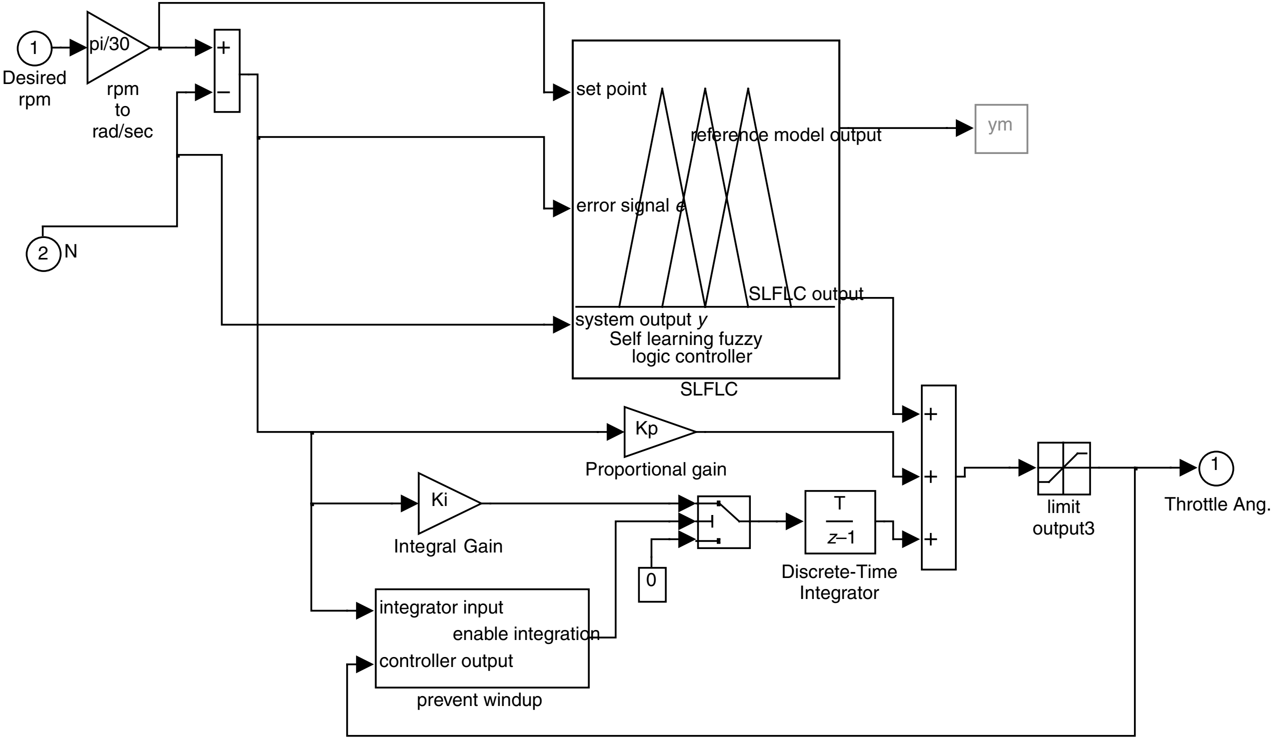


FIGURE 6.27 The structure of a hybrid self-learning fuzzy controller (PI + SLFLC). (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *8th IEEE Mediterr. Conf. Contr. Autom. MED'00*, TC-2.5, 2000. With permission.)

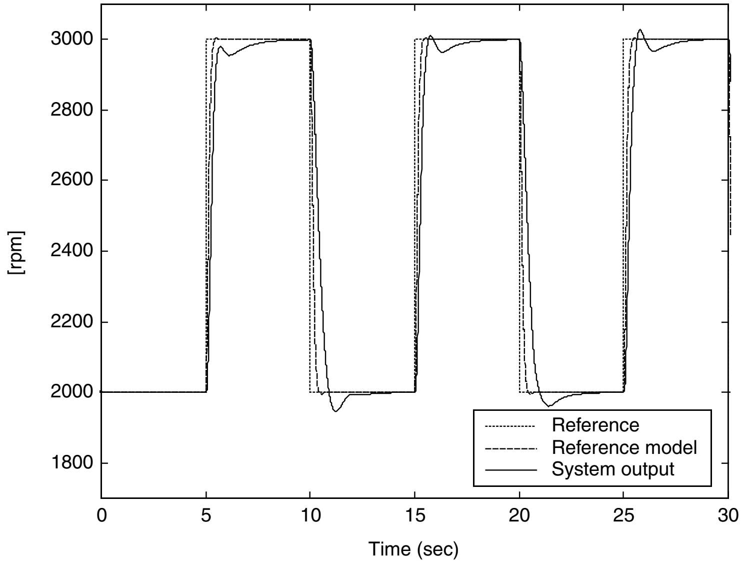


FIGURE 6.28 Start of learning: the reference input, system output, and reference model output responses of the engine speed control system. (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *8th IEEE Mediterr. Conf. Contr. Autom. MED'00*, TC-2.5, 2000. With permission.)

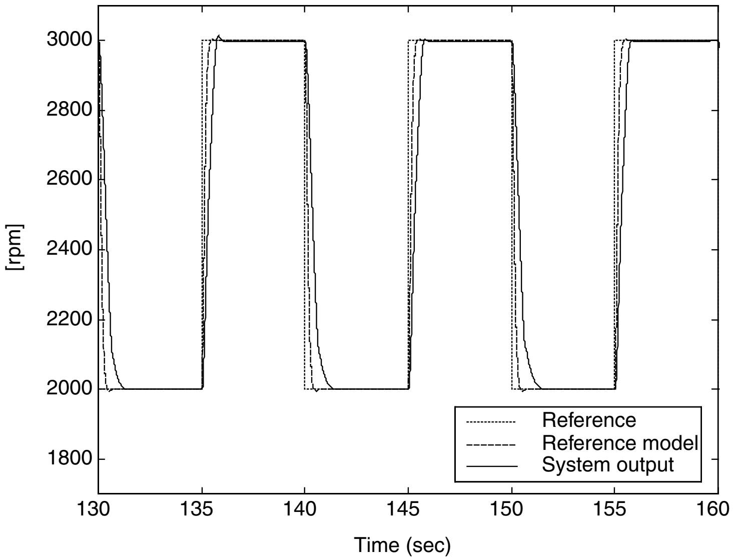


FIGURE 6.29 End of learning: the reference input, system output, and reference model output responses of the engine speed control system. (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *8th IEEE Mediterr. Conf. Contr. Autom. MED'00*, TC-2.5, 2000. With permission.)

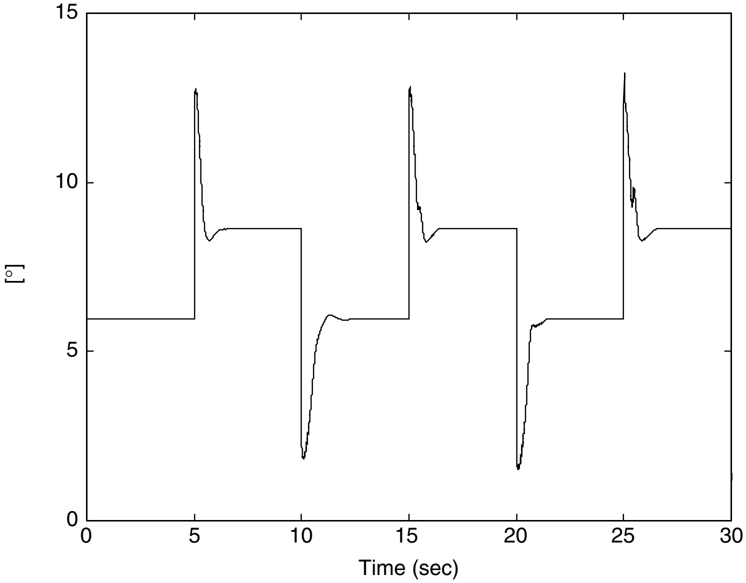


FIGURE 6.30 Start of learning: the hybrid self-learning fuzzy controller output responses. (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *8th IEEE Mediterr. Conf. Contr. Autom. MED'00*, TC-2.5, 2000. With permission.)

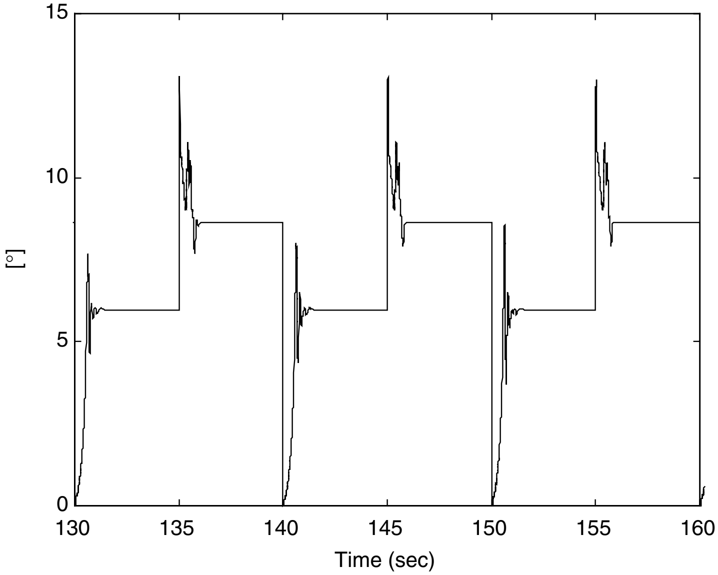


FIGURE 6.31 End of learning: the hybrid self-learning fuzzy controller output responses. (From Kovačić, Z., Bogdan, S., and Reichenbach, T., *8th IEEE Mediterr. Conf. Contr. Autom. MED'00*, TC-2.5, 2000. With permission.)

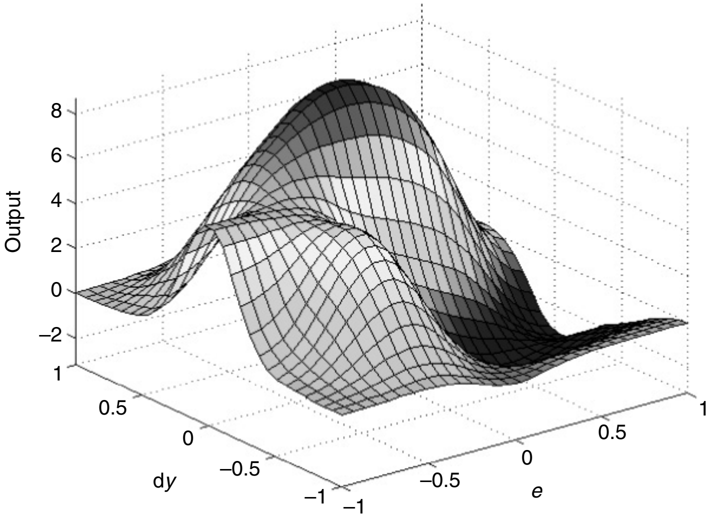


FIGURE 6.32 End of learning: the graph of the SLFLC control surface.

6.5 Design Project: Fuzzy Control of a Electro-Hydraulic Servo System

MATLAB offers a very convenient environment for testing new designs and control solutions before their practical implementation in real control systems. This refers also to the design of standard or self-organizing fuzzy controllers described in previous sections. In this chapter we give a description (mathematical and simulation model) of a nonlinear electro-hydraulic servo system. The intended project task is to use the MATLAB FLT and design a fuzzy controller which will control the given electro-hydraulic servo system. For this purpose the standard FLT fuzzy controller block or PSLFLC and SLFLC super-blocks can be used. Although fuzzy control does not depend on a mathematical model of the controlled process, it can use it as a source of useful information for creation of a knowledge base. In our case we also need a mathematical model for a better (realistic) presentation of an unknown, inherently stable real control process. The idea is to design a fuzzy controller for an unknown process and thereafter make experiments for simulated changes of given control process parameters. It can be helpful to make a plan of experiments in advance so that a later on analysis of results becomes easier.

6.5.1 Mathematical Model of a Control Process

A given electro-hydraulic servo system shown in Figure 6.33 is a nonlinear control process with time-varying parameters. The servo system contains a double-acting 200 mm stroke cylinder loaded by a mechanical part of a given mass, elasticity, and damping, and a current-driven servo valve that controls the flow of the fluid in the cylinder [5].

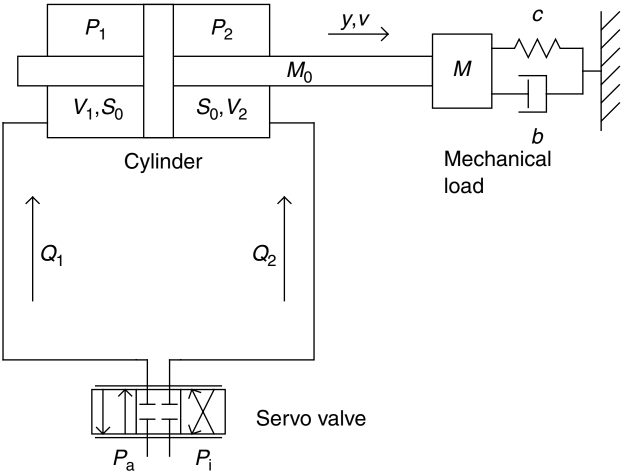


FIGURE 6.33 Electro-hydraulic system.

The electrical part of the electro-hydraulic servo valve can be described with a second-order transfer function:

$$\frac{Y_{\mathrm{u}}(s)}{U(s)} = \frac{-k\omega_{\mathrm{n}}^2}{s^2 + 2\zeta_{\mathrm{n}}\omega_{\mathrm{n}}s + \omega_{\mathrm{n}}^2} \tag{6.3}$$

where Y_{u} is the servo valve spool position, m; U the servo valve input current, A; k the servo valve gain coefficient, m/A; ω_{n} the servo valve natural frequency, rad/sec; and ζ_{n} the servo valve damping coefficient.

The spool of the servo valve slides in a sleeve, which controls the flows of the fluid in cylinder chambers. The mass flow rate Q is proportional to the effective cross-section area S and the square root of pressure difference between the two points $\sqrt{\Delta P}$. Since the effective area S is proportional to the servo valve spool position y_{u} (S changes with changes of y_{u}), the mass flow rate through the servo valve is described with the following equations:

$$Q_1(y_{\mathrm{u}}, P_1) = \begin{cases} y_{\mathrm{u}} \cdot \sqrt{|P_i - P_1|} \cdot \mathrm{sign}(P_i - P_1) & \text{for } y_{\mathrm{u}} \geq 0 \\ y_{\mathrm{u}} \cdot \sqrt{|P_1 - P_a|} \cdot \mathrm{sign}(P_1 - P_a) & \text{for } y_{\mathrm{u}} < 0 \end{cases} \tag{6.4}$$

$$Q_2(y_{\mathrm{u}}, P_2) = \begin{cases} -y_{\mathrm{u}} \cdot \sqrt{|P_2 - P_a|} \cdot \mathrm{sign}(P_2 - P_a) & \text{for } y_{\mathrm{u}} \geq 0 \\ -y_{\mathrm{u}} \cdot \sqrt{|P_i - P_2|} \cdot \mathrm{sign}(P_i - P_2) & \text{for } y_{\mathrm{u}} < 0 \end{cases} \tag{6.5}$$

where P_1 is the pressure in the left-hand and P_2 in the right-hand cylinder chamber.

Servo valve gain coefficient k already includes a proportional gain associated with the mass flow rate, so it is omitted in Equations (6.4) and (6.5). P_i is the pressure supplied by the hydraulic pump, and P_a is the tank pressure. An assumption is made that P_i and P_a have constant values.

For the symmetric servo valve flow stage mass flow rates Q_1 and Q_2 are equal:

$$Q_1(y_\mathrm{u}, P_1) = -Q_2(y_\mathrm{u}, P_2) \tag{6.6}$$

The cylinder is described with a following thermodynamic equation:

$$\frac{V}{B} \cdot \frac{\mathrm{d}P}{\mathrm{d}t} + \frac{\mathrm{d}V}{\mathrm{d}t} = Q \tag{6.7}$$

Parameter B is the bulk isotherm modulus of the fluid (oil), while V, P, and Q are volume, pressure, and mass flow rate in a cylinder chamber, respectively. Volumes of two cylinder chambers change with the cylinder rod position y in a way that:

$$\begin{aligned} V_1 &= V_0 + S_0 y \\ V_2 &= V_0 - S_0 y \end{aligned} \tag{6.8}$$

where S_0 is the cylinder rod effective area and V_0 is the half-volume.

By insertion of (6.8) into (6.7) we get a description of pressure behavior in two chambers of the cylinder:

$$\frac{\mathrm{d}P_1}{\mathrm{d}t} = \frac{B}{V_0 + S_0 y}(Q_1 - S_0 v) \tag{6.9}$$

$$\frac{\mathrm{d}P_2}{\mathrm{d}t} = \frac{B}{V_0 - S_0 y}(Q_2 + S_0 v) \tag{6.10}$$

where $v = \mathrm{d}y/\mathrm{d}t$ is the cylinder rod speed.

The mechanical part of the system is described with the following dynamic equation:

$$(M + M_0) \cdot \frac{\mathrm{d}^2 y}{\mathrm{d}t^2} = S_0 \cdot P_1 - S_0 \cdot P_2 - b \cdot \frac{\mathrm{d}y}{\mathrm{d}t} - c \cdot y - F_\mathrm{f} \tag{6.11}$$

where M_0 is the cylinder rod mass, M is the mass of the mechanical part, b and c are viscous coefficient and elasticity of the mechanical part. In the given simulation model, friction force F_f is neglected.

6.5.2 Simulation Model

The electro-hydraulic system described in Reference 5 has parameter values displayed in Table 6.4.

Figures 6.34 to 6.37 show MATLAB–Simulink simulation models of the electro-hydraulic system, electro-hydraulic valve, double-acting hydraulic

TABLE 6.4
Parameters of the Electro-Hydraulic System

Variable	Variable description	Values	SI units
ω_{n}	Servo valve natural frequency	500	rad/sec
ξ_{n}	Servo valve damping coefficient	0.4	
k	Servo valve gain coefficient	5.1e−5	m/A
P_i	Hydraulic pump pressure	280e−5	Pa
P_a	Tank pressure	1e−5	Pa
V_0	Cylinder half-volume	5e−4	m^3
M_0	Cylinder rod mass	50	kg
S_0	Cylinder rod effective area	1.53e−3	m^2
B	Bulk isotherm modulus	7e−8	Pa
M	Mechanical part mass	20	kg
c	Mechanical part elasticity coefficient	1e−5	N/m
b	Mechanical part viscous coefficient	5000	Nsec/m

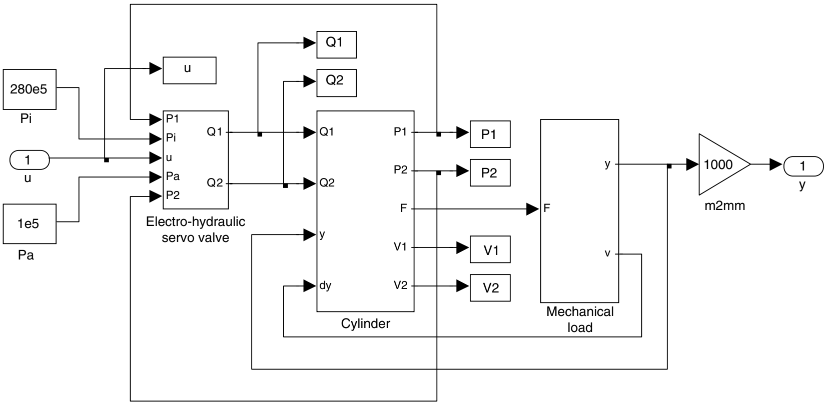


FIGURE 6.34 Simulation model of the electro-hydraulic system.

cylinder, and the mechanical part of the system created based on the described mathematical model.

6.5.3 Fuzzy Controller Design Specifications

For the given electro-hydraulic system the following design specifications should be taken into account:

1. The range of the signal u fed into the simulation model (see Figure 6.34) is [−20, 20], which corresponds to the servo valve input current in milliamps.

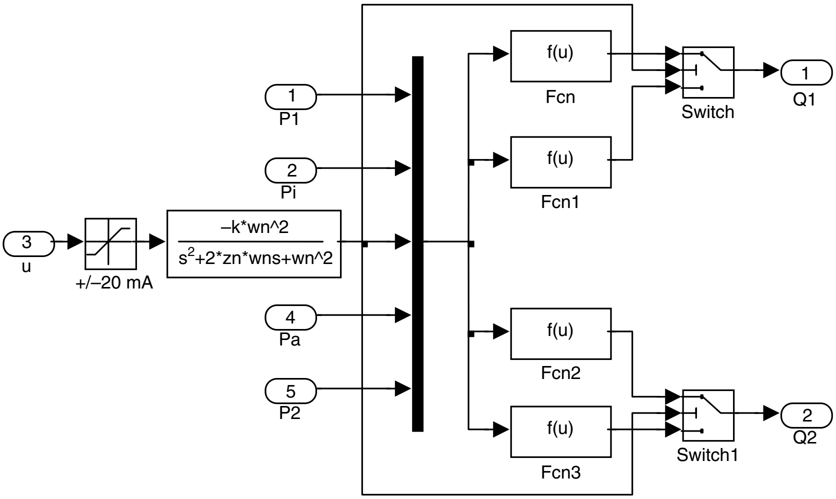


FIGURE 6.35 Simulation model of the electro-hydraulic servo valve.

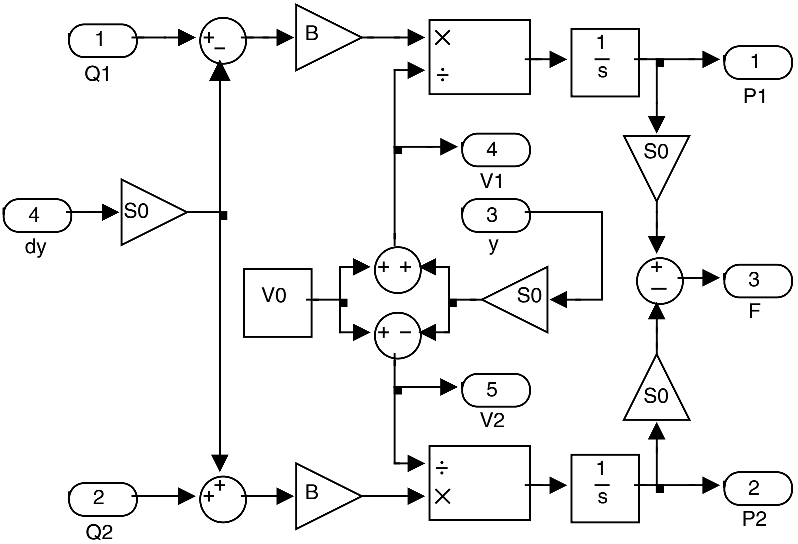


FIGURE 6.36 Simulation model of the double-acting hydraulic cylinder.

2. The range of the cylinder rod position signal y at the output of the simulation model (Figure 6.34) is $[-300, 300]$, which corresponds to the cylinder rod position in millimeters.
3. When designing a fuzzy controller the following steps must be taken:
   - Determination of the character of the controlled process (whether it is static or astatic). This implies the type of the fuzzy controller

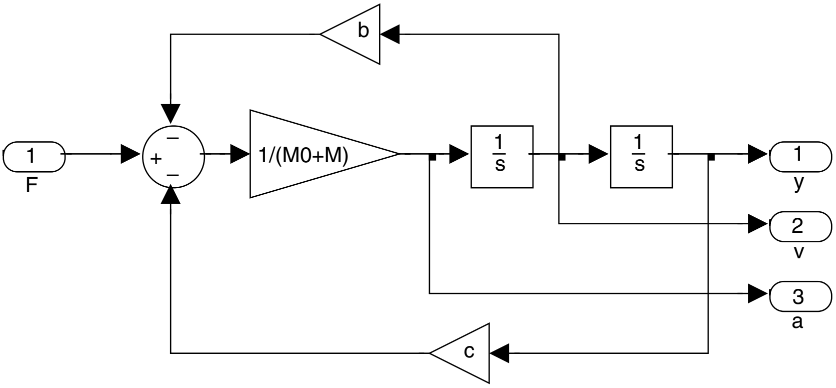


FIGURE 6.37 Simulation model of the mechanical part of the system.

to be designed, a PI-type or a PD-type, respectively. The character of the process can be identified by observing an open-loop process response.

- Determination of a desired control quality. This can be done by setting second-order reference model parameters (σ_m, t_m).
- Definition of the universes of discourse of input and output variables of the fuzzy controller. In order to preserve the universality of the controller, let the fuzzy controller have a normalized form, that is, let the universe of discourse of each input be [−1, 1]. It can be useful to observe $e - \Delta e$ or $e - \Delta y_\mathrm{M}$ trajectories of the reference model for better estimation of the boundaries of fuzzy controller input universes of discourse.
- Decision about the number of fuzzy sets (usually 5 or 7 for each input).
- Selection of the type of fuzzy membership functions (Gaussian type by default for the SLFLC super-block).
- Definition of fuzzy control rules (normally up to 49 rules).
- Selection of the fuzzy inference mechanism.
- Selection of the defuzzification method (usually COG).
- Determination of a control interval T_d. The idea is to set its value so that the values of Δe or Δy_M have an impact on the fuzzy control rules. For too small T_d, Δe or Δy_M will loose that impact as they will also be too small. A reference model response can be taken as a basis for good estimation of T_d (e.g., $t_\mathrm{m}/20 < T_\mathrm{d} < t_\mathrm{m}/10$).

The control problem can be solved by designing a standard digital DISO type fuzzy controller or PSLFLC controller with two inputs, error signal $e(k)$ and change of error $\Delta e(k)$, and one output denoted as $u(k)$. Most often the signals

TABLE 6.5
Input and Output Universes of Discourse for Different Scaling Factor Values

Resolution of ADC and DAC	Input scaling factor K1	Output scaling factor K2	Input variables range	Output variables range
8-bit	12.8	0.078125	[−128, 128]	[−128, 128]
12-bit	204.8	0.004882812	[−2048, 2048]	[−2048, 2048]

are fed into the digital fuzzy controller through A/D converters (ADC). Accordingly, in the simulation model shown in Figure 6.38, the scaled reference input and system feedback signals are also fed into the fuzzy controller through ADCs and a multiplexer. If the SLFLC function super-block is used, then inputs fed into the controller become error signal $e(k)$ and change of the system output $\Delta y_{\mathrm{f}}(k)$.

Assuming that the ranges of physical input values are matched with the input range of ADCs, the value of input scaling factors depends on the actual ADC resolution. The scaling factor value for the unipolar 8-bit ADC used in the simulation scheme shown in Figure 6.38 is K1 = 1/0.078125 = 12.8, where 0.078125 V represents the physical value of the least significant bit. With 12-bit ADCs this coefficient takes value K1 = 1/0.004882812 = 204.8. Output scaling is done in a similar way (e.g., coefficient K2 = 1/12.8 = 0.078125). By deployment of such scaling factors, the input and output universes of discourse are set as displayed in Table 6.5.

A better resolution of ADCs can help achieve a better control quality. Higher resolution also provides better granularity of fuzzy input values, which in return yields more effective control.

Once the fuzzy controller design is finished, the plan of experiments may contain the following steps:

- Fuzzy controller sensitivity to process parameter variations (small, modest, large)
- Variation of the number of input and output fuzzy sets
- Variation of the shape of input membership functions
- Variation of the distribution of input and output fuzzy sets (linear or nonlinear)
- Variation of fuzzy rules (changes of consequent parts)
- Variation of fuzzy implication operators and defuzzification methods
- Modification of input and output scaling factors
- Variation of the resolution of A/D and D/A converters (8-, 10-, 12-bit)
- Variation of the process structure (including new “unknown” processes).

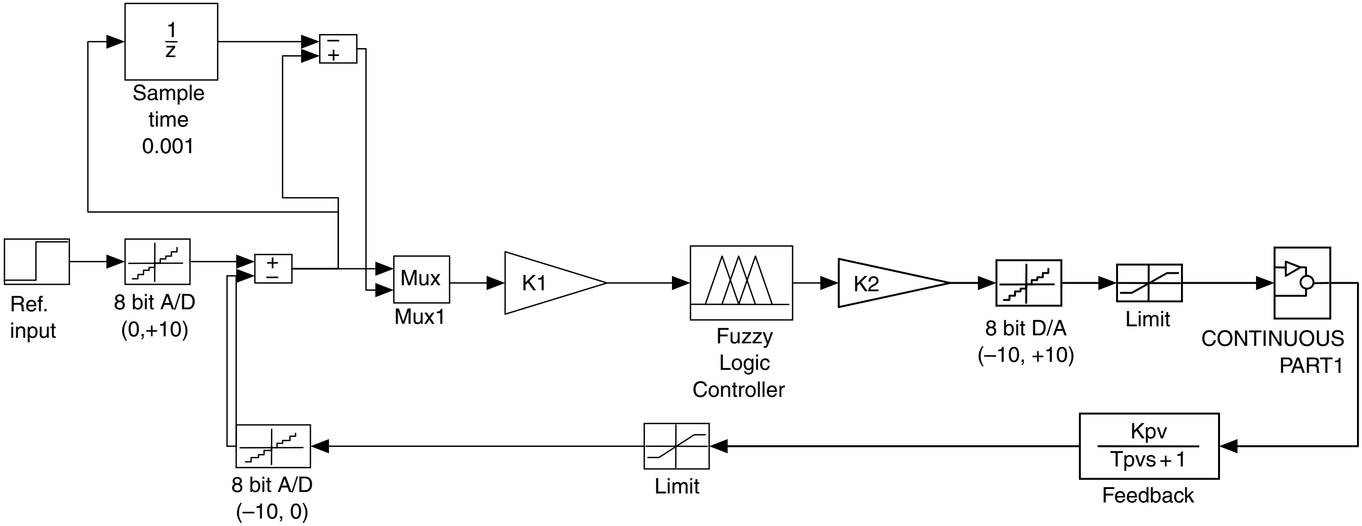


FIGURE 6.38 Example of a simulation model of a fuzzy controlled system.

References

1. Jang, J.-S.R. and Gulley, N., “Fuzzy logic toolbox — for use with Matlab,” The MathWorks Inc., 1995.
2. http://flrcg.rasip.fer.hr/slflc/: “Self-learning fuzzy controller function super-blocks for Matlab 5.2, 5.3 and 6.5,” LARICS — Laboratory for robotics and intelligent control systems, Faculty of Electrical Engineering and Computing, University of Zagreb, 2003.
3. Kovačić, Z., Bogdan, S., and Reichenbach, T., “A class of self-learning fuzzy logic controllers designed as a function block for Matlab+Simulink™ environment,” *The 11th International Conference on Electrical Drives and Power Electronics EDPE’00*, Dubrovnik, Croatia, pp. 93–98, 2000.
4. Kovačić, Z., Bogdan, S., and Reichenbach, T., “Demonstration of self-learning fuzzy logic controller performance in the Matlab+Simulink™ environment,” *The 8th IEEE Mediterranean Conference on Control and Automation MED’00*, CD-ROM Proceedings, TC-2.5, Patras, Greece, 2000.
5. Pommier, V., Sabatier, J., Lanusse, P., and Oustaloup, A., “Crone control of a nonlinear hydraulic actuator,” *Control Engineering Practice*, 10, 391–402, 2002.

7 Implementation of Fuzzy Controllers for Industrial Applications

The fuzzy control technology has emerged as one of the most effective nonlinear control technologies used in industrial applications. Everything began in 1970s with pioneering work of E.H. Mamdani and first successful application of fuzzy logic for steam engine control [1], followed by contributions of many other eminent fuzzy control scientists and engineers [2–7]. Since then the spectrum of fuzzy control applications is steadily growing. Thanks to the joint efforts of university and industry research teams all over the world, in 1980s and 1990s, a “fuzzy boom” happened in various application fields. For example, fuzzy control found its way in traffic control and transportation [8–10], process industry [11–15], robotics [16,17], flight control [18–20], but even more important, in short time the fuzzy control has become a standard solution in a large number of consumer products such as photo cameras, washing machines, air-conditioners, and many others. In this early fuzzy control stage, Japan was leading in the number of registered patents and launched new products (especially leading companies such as Omron, Fuji, Hitachi, and Matsushita). Excellent reviews on the historical development of fuzzy control engineering and numerous applications can be found in References 21 and 22.

These days fuzzy control applications are so widely spread that describing them all would be impossible. Instead, we shall stay focused on generic fuzzy controller implementation concepts, which, we believe, might be more helpful to the reader in his or her future fuzzy control designs and implementations of fuzzy controller structures described in the book.

In this chapter, we also describe few selected applications, which show the versatility of fuzzy control solutions, from the control of a road tunnel ventilation system to the control of anesthesia during demanding surgical operations [23,24].

7.1 Brief Overview of Industrial Fuzzy Controllers

When analyzing the features that an industrial fuzzy controller should have, we find that it should provide:

- Fast processing of a large number of inputs, outputs, and fuzzy control rules
- Configurability of controller’s structure

- Programmability of controller parameters
- Short control intervals
- Sufficient resolution of the controlled variable
- Possibility to work in parallel with other types of controllers
- Low power consumption of the fuzzy processing circuitry
- Support for common communication standards (RS232, RS485, CAN, Profibus, Modbus, Ethernet, Internet, etc.)
- Low-cost control solution

In parallel with the development of fuzzy control theory and new control applications an intensive development of hardware for fuzzy control processing is going on. Two concepts of fuzzy controllers' implementation, analog and digital, are competing with each other (see Figure 7.1). As a result, there is a variety of analog, digital, and mixed signal fuzzy controller architectures at the designer's disposal for solving both simple and very complex control problems [25–27].

The analog fuzzy controller is interesting in those applications such as the above mentioned mass produced consumer products where inputs and outputs are analog signals. Some controller structures employ analog circuitry only for input and output conversion, while fuzzy processing is done digitally with general-purpose microprocessors or application-specific integrated circuits. Other structures, such as the complementary metal oxide semiconductors (CMOS) analog fuzzy controller structure developed by Prof. Yamakawa exploits the functional capabilities of the MOS transistor to implement the fuzzy operators with very simple circuitry [28]. Although the fuzzy processing is analog, such a controller is programmed

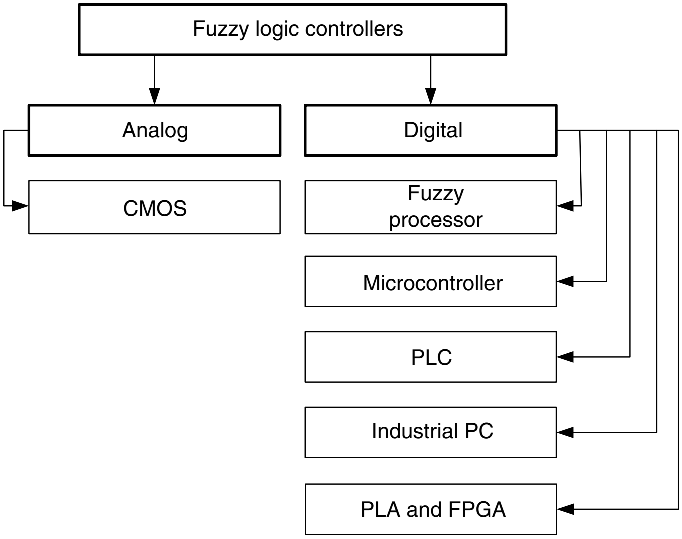


FIGURE 7.1 Different platforms for implementation of a fuzzy controller.

digitally. The successive CMOS analog fuzzy controller structures were able to achieve larger operation speeds and lower power consumption [29–39].

Even though very attractive and widely used, analog fuzzy controllers are characterized by limited precision at one hand, and weak structural flexibility on the other. The former can be overcome by a careful circuit design, but larger flexibility and programmability can be achieved only with a fully digital implementation.

As shown in Figure 7.1, digital fuzzy controllers can be designed using different implementation techniques and different hardware platforms.

The architecture and functionality of fuzzy processors are very similar to those of standard microprocessors. The only difference is in enhancement of the processor core with additional digital circuits and microcoded instructions for execution of main fuzzy controller operations. Among many different types of fuzzy processors, let us mention a few: NeuraLogix NLX-230 Fuzzy Microcontroller, Togai InfraLogic FC110 digital fuzzy processor, Omron FP-3000 fuzzy processor, SGS Thomson Weight Associative Rule Processor (WARP), and Siemens FUZZY 166 microcontroller (this is a fuzzified version of the 16-bit Siemens microcontroller family 80C166) [26,40,41]. Fuzzy processors are programmable like standard microprocessors, but the execution of fuzzy operations is much faster (shorter control intervals). Only the cost of such processors is slightly higher due to objectively lower quantities being produced.

General purpose programmable logic controllers (PLCs) designed for closed-loop process control applications usually have a fuzzy controller function block as an option that can be purchased separately. The Klöckner–Moeller Corporation made a fuzzy PLC based on the Siemens FUZZY 166 microcontroller [42], but otherwise fuzzy PLCs are very rare.

On the other hand, PC-based process control solutions are more and more present in today's control practice. The increased reliability of PC operating systems and the availability of a wide spectrum of PC modules and instruments at the world market allow an easy integration of a PC-based control system. A large number of software tools make this option even more attractive. PC-based fuzzy controllers are designed by using some of the commercially available development software programs. Among the many popular tools, let us mention just a few: fuzzyTech from INFORM Corporation, TILShell from Togai InfraLogic, and CubiCalc from HyperLogic Corporation. These and similar tools do not only allow the user to define and tune a fuzzy controller but also generate a high level language source code or an optimized assembly code for the PC and various standard microcontroller families.

The significant increase in operating speed and ability to process a large number of inputs, outputs, and fuzzy control rules can be accomplished by programming fuzzy controller operations in programmable logic arrays (PLA) and field programmable gate arrays (FPGA). The PLA can be used for implementation of a customized fixed structure of a fuzzy controller [43]. The FPGA is a digital integrated circuit that can be programmed to do any type of digital function. It may have more than million gates that allow building of large and complex processing units. Compared to a standard microprocessor, the FPGA can be programmed on the fly. This feature is very convenient for execution of customized high-speed digital functions [44].

7.2 Implementation Platforms for Industrial Fuzzy Logic Controllers

As we have mentioned above, nowadays, there are too many fuzzy control applications in the industry to describe all of them. The reader is suggested to refer for further information about specific application fields in numerous application-oriented books, journals, and conference proceedings. This book will stay focused on the elaboration of generic implementation concepts suitable for different implementation platforms. For this purpose, each implementation concept and platform included in the chapter are illustrated with an accompanying worked-out example.

We do not pay attention to fuzzy controllers made with commercial fuzzy controller design tools as such implementations must conform to the given design rules that are specific for each tool. In such a case, a generated code, even if it is optimized, is actually constrained by the design procedure and the prescribed output format.

First we describe an implementation of a fuzzy controller by using a low-cost 8- or 16-bit microcomputer and executable code written in the assembly language. Such an implementation platform allows very fast execution of fuzzy control algorithms providing in turn very short control intervals. This becomes very interesting in control applications with fast dynamics as in the control of servo systems.

Installations of PLCs are widely present in today's process industry. Manufacturers build PLCs according to international standards, which enable an easy buildup of various control solutions. PLC programming can be done with several standard programming techniques such as ladder diagrams (LD), statement lists (STL), and function block diagrams (FBD). These techniques allow the PLC programmer to introduce new control functions by using existing PLC elements or by creating new ones. In this chapter, we describe a standard PLC-based implementation of the fuzzy controller in the case of controlling a condenser level in the thermal power plant.

Besides standard PLC solutions, there are more and more industry applications based on so-called soft PLC solutions. The main characteristic of a soft PLC is that its program, usually developed on a PC with adequate programming tool, can be downloaded as a finalized project into different target platforms (e.g., embedded microcontrollers, PLCs, or industrial PCs). In this chapter, we describe an implementation of a sensitivity model-based self-learning fuzzy controller (see Section 5.3) as a new soft PLC function block. We pay more attention to the description of a multimode operation concept that best suits the needs of a process operator.

Thanks to a large number of specialized programming tools, high level language programming (mostly in C and C++) has almost replaced the assembly programming. Regardless from the type of programming language, an implemented fuzzy controller would be more or less unique. Namely, such a way of implementation gives the programmer a lot of freedom to create a unique fuzzy controller structure.

7.2.1 Microcomputer-Based Fuzzy Controller Implementation

As we have mentioned above, high level language programming of popular 8- and 16-bit microcontrollers has become a common practice. Though advantages of high level language programming are unquestionable, programming in the microcontroller assembly language can be even more advantageous when issues like processing of a large number of fuzzy control rules as well as switching between coarse and fine control regimes are considered. The techniques described here include a page-wise scaling of inputs and outputs, register-oriented fuzzy encoding of binary input values, and simple direct-access scanning of almost any size of a fuzzy rule-table.

In order to demonstrate the above mentioned assembly language-based techniques, let us consider an implementation of the hybrid fuzzy controller described in Section 4.1.

A standard 8- or 16-bit microcomputer-based digital DISO fuzzy controller that we want to implement performs the following basic operations:

1. Analog-to-digital (A/D) conversion of controller inputs (we assume that we use a 10-bit ADC).
2. Digital-to-fuzzy conversion of controller inputs (let us suppose the most frequent fuzzification to seven fuzzy sets), each fuzzy set is normal and regular, that is, the corresponding membership functions are symmetrical and they have a unity maximum membership value.
3. Scanning of active fuzzy control rules (we assume that not more than 4 out of 49 rules can be activated in one control interval).
4. Fuzzy inference.
5. Fuzzy-to-digital conversion (defuzzification), that is, computation of the crisp controller output according to the center of gravity principle (let us suppose that magnitudes of the controller output are quantized to 15 singleton values).
6. Digital-to-analog (D/A) conversion of the controller output.

Usually, A/D conversion of controller inputs is performed by using A/D converters, but there are other types of A/D conversion, too. For example, in digitally controlled electrical drives and servo systems, incremental encoders are used for a digital measurement of position and angular speed. Also, a digital measure of an analog signal can be obtained by successive voltage-to-frequency (V/f) and frequency-to-binary (f/D) conversions. Regardless of the type of A/D conversion method, the final result is always a digital number with a determined resolution.

Suppose that we use a 10-bit bipolar A/D converter. Then controller inputs $e(k)$ and $\Delta e(k)$ create a phase plane $e(k) - \Delta e(k)$ where trajectory points $[e(k), \Delta e(k)]$ take values from the ±9-bit interval (±512). The lower the input values, the finer the control. In order to discern fine from coarse control, let us split the phase plane into a set of disjunctive areas (pages) nested within each other. Similar concept with nested fuzzy rule-tables for coarse and fine control was presented in Reference 45. The phase plane shown in Figure 7.2 has four nested pages: the most inner page (1st page), the middle pages (2nd and 3rd pages), and the most outer page (4th page). Fine control is active only in the first page. Coarse control becomes active for trajectory points $[e(k), \Delta e(k)]$ lying in other pages.

One can see that pages are binary proportional (the second page is two times larger than the first one; the third page is two times larger than the second one, etc.). Splitting the phase-plane into binary proportional pages can significantly simplify the switching between fine and coarse control. That is, in such a concept a fuzzy controller is designed only for the fine control region. When we deal with coarse control, first we scale trajectory points $[e(k), \Delta e(k)]$ lying in outer (coarse control) regions into the fine control page. With binary proportional pages scaling means division by the power of two (two, four, etc.). A most distant trajectory point

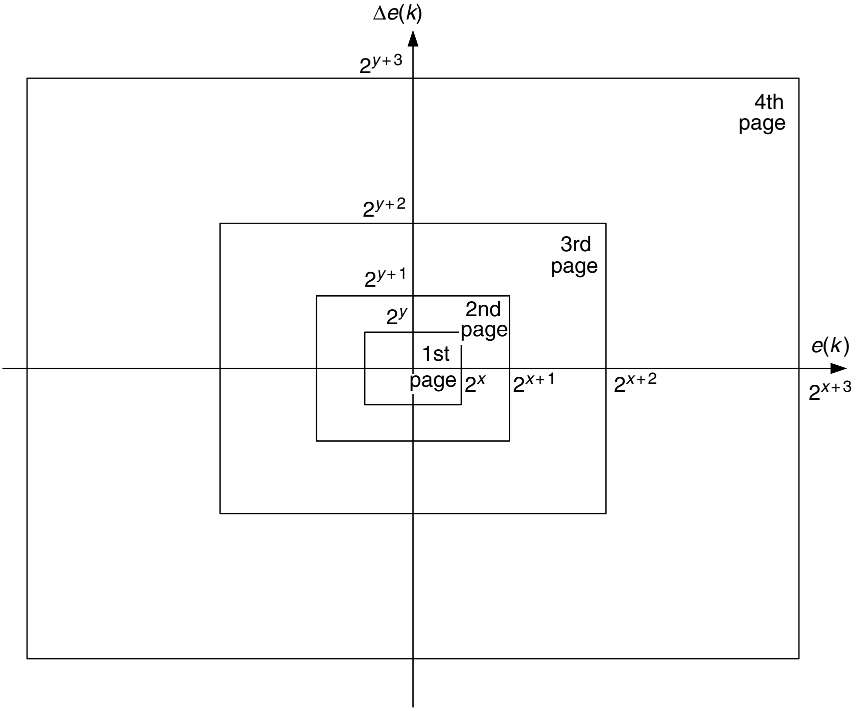


FIGURE 7.2 A page-wise representation of a phase plane $e(k) - \Delta e(k)$ in the case of using a 10-bit bipolar A/D converter.

determines a scaling factor for all trajectory points. After execution of the fuzzy control algorithm, we return the crisp output value $u_{\mathrm{FC}}(k)$ reversely scaled with the same factor. Reverse scaling means multiplication by the same scaling factor. In terms of assembly programming, scaling and reverse scaling mean execution of simple shift to the right and shift to the left assembly instructions, respectively. When the controlled process is far from a steady-state condition and the trajectory points belong to outer pages, division of integer values by the powers of two may slightly affect the coarse control resolution, but the fine control resolution remains unaffected.

Reverse scaling of the fuzzy controller output obtained in the fine control region actually means that the designed nonlinear fuzzy control function is linearly extended to the coarse control region. It must be noted that such way of handling fine and coarse control appears very natural and easy to implement for different types of A/D converters. In our example, the first page would correspond to the usage of an 8-bit A/D converter and likewise, five pages could be nested in the case of using a 12-bit A/D converter. In general, pages may have freely determined dimensions (e.g., 128 × 128 or 128 × 16). The size of the first page will primarily depend on the desired quality of the control.

The DISO fuzzy controller under consideration has seven fuzzy sets defined for both inputs. Therefore, the fuzzy rule-table is a 7 × 7 matrix, which may potentially contain 49 fuzzy control rules. The number of rules that must be processed in each control interval is usually a limiting factor, which affects the speed of microcomputer-based computation. Assuming that only adjacent sets overlap, maximally 4 out of 49 control rules may contribute to the crisp controller output.

Now let us describe a simple way of scanning a 7 × 7 fuzzy rule-table. Seven fuzzy sets of each input can be encoded by using 3 bits of the allocated 8-bit register or memory location. Since both inputs may simultaneously belong to two fuzzy sets, we use two allocated registers for storing information about the fuzzification (see Figure 7.3): bits 0, 1, and 2 for the change of error $\Delta e(k)$ and bits 3, 4, and 5

REGISTER 1

7	6	5	4	3	2	1	0	
X	X	0	0	0	X	X	X	ZE
X	X	0	0	1	X	X	X	PSE
X	X	0	1	0	X	X	X	PME
X	X	0	1	1	X	X	X	PLE
X	X	1	0	0	X	X	X	NSE
X	X	1	0	1	X	X	X	NME
X	X	1	1	0	X	X	X	NLE

REGISTER 2

7	6	5	4	3	2	1	0	
X	X	X	X	X	0	0	0	ZDE
X	X	X	X	X	0	0	1	PSDE
X	X	X	X	X	0	1	0	PMDE
X	X	X	X	X	0	1	1	PLDE
X	X	X	X	X	1	0	0	NSDE
X	X	X	X	X	1	0	1	NMDE
X	X	X	X	X	1	1	0	NLDE

FIGURE 7.3 Encoding of fuzzy input sets by using two 8-bit registers.

		$e(k)$,	$\Delta e(k)$
Ad_0	A_{00}	ZE,	ZDE
+1	A_{01}	ZE,	PSDE
+2	A_{02}	ZE,	PMDE
+3	A_{03}	ZE,	PLDE
+4	A_{04}	ZE,	NSDE
+5	A_{05}	ZE,	NMDE
+6	A_{06}	ZE,	NLDE
+7	not used		
+8	A_{10}	PSE,	ZDE
+9	A_{11}	PSE,	PSDE
	...		
+55	A_{66}	NLE,	NLDE

FIGURE 7.4 Fuzzy rule-table addressing in the form of a look-up table.

for the error input $e(k)$. Two spare bits 6 and 7 may be used for encoding of up to 15 fuzzy input sets.

The information packed in the registers actually describes the antecedent parts of currently activated fuzzy rules. In addition, the contents of registers can be used to create an index, which, when added to the base look-up table address points at the memory location with the corresponding output singleton value (Figure 7.4). Then we can scan the table in the following way

$$\mathrm{Ad} = \mathrm{Ad}_0 + \mathrm{Ad}_1 \tag{7.1}$$

where Ad is the total look-up table address, Ad_0 the base look-up table address, and Ad_1 is the index defined by logical combinations of the contents of two registers (REGISTER 1 and 2).

In case of using seven fuzzy subsets, only 56 bytes of memory (Figure 7.4) are sufficient to store all singletons from the fuzzy rule-table. In case of using more than seven subsets (e.g., 11), four bits can be used to scan the fuzzy rule-table.

The aim of the described assembly language-based techniques and solutions is to simplify time-critical fuzzy controller operations and thus enable the usage of standard low-cost microcomputers for implementation of fuzzy controllers. Page-wise scaling and register-oriented fuzzy encoding of binary input values along with direct access scanning of the fuzzy rule-table was successfully implemented in the microcomputer-based hybrid fuzzy controller described in Section 4.2.5. This controller was implemented with a 32-bit VME-based microcomputer hardware (Motorola 68000) and 12-bit A/D converters, and experimental results proved that the above mentioned implementation techniques were very effective.

7.2.2 PLC-Based Fuzzy Gain Scheduling Control of Condensate Level

Programmable logic controllers have become dominant control devices in process automation. Capable of operating as standalone devices or elements of distributed control system networks, PLCs represent a very good basis for various control solutions. Standard PLCs are almost regularly equipped with standard types of controllers: P, PI, or PID. Some PLC manufacturers offer fuzzy controllers as options, which can be ordered separately at additional cost. The reason why fuzzy controllers are not so widely present in PLC-based applications lies in the fact that the setting of fuzzy controller parameters requires expertise in fuzzy control design and adequate commissioning tools.

Different ways of PLC programming allow the programmer to use existing PLC elements (e.g., function blocks) or to follow instructions and create new ones. In this chapter, we describe a PLC-based fuzzy controller implementation by using existing PLC elements.

The fuzzy controller implementation is demonstrated on the problem of condensate level control in the thermal power plant at KTE Jertovec, Croatia. Due to a nonlinear geometry of the drum and nonlinearities induced by a pump and a valve, the control process is highly nonlinear. In order to avoid these problems and to improve the control quality, a fuzzy gain scheduling control scheme is designed and implemented with an industrial PLC Siemens Simatic S7-216 [46].

As one of important components of a thermal power plant, a condenser serves for indirect heat exchange that occurs during heat transfer from the warmer fluid (steam) to the cooler fluid (water). During this process, the turbine exhaust steam is condensing and the condensate is accumulating in a hot well at the bottom of the condenser and flowing out through a shell fluid outlet [47].

Besides its main function to condense the turbine exhaust steam, the condenser's role is also to maintain a specified vacuum value at the turbine exhaust in order to keep the power cycle efficiency high, releasing the heat at the lowest possible temperature. Among various factors that influence that pressure, the level of condensate plays an important role. The condensate level value should be constant across the whole range of operating conditions. There are two additional reasons for keeping the condensate level constant. One of them is that the change of level can cause a cavitation effect in the pump located behind the hot well, and due to this effect, the pump can be seriously damaged if the condensate level drops bellow the lower level limit. Another negative effect can be caused by an uncontrolled increase of the condensate level. At some point the condensate can reach the level of condenser pipes and thus impair their cooling ability.

Usually the standard condensate level control loop incorporates a simple controller such as a three-state controller (TSC) with a hysteresis. Even though the level control loop is slow and the process is well known, in some cases nonlinearities encountered in the system may trigger problems. In that case, commonly used standard control structures experience difficulties in maintaining the desired control quality.

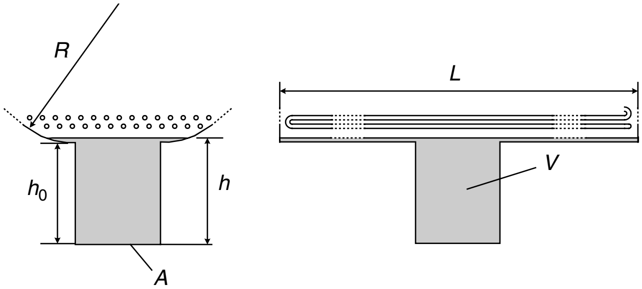


FIGURE 7.5 The schematic diagram of the condenser. (From Bogdan, S., Kovačić, Z., Lončar, D., and Lukačević, D., *KoREMA 9th Mediterr. Conf. Contr. Autom.*, Session FM1-A, 2001. With permission.)

The operating conditions of the power plant influence dynamics of the condensate level. In order to compensate this influence, measurement of process variables that enable estimation of the change of level must be provided. This will allow the usage of a fuzzy logic algorithm as a gain scheduler [48].

The schematic diagram of the condenser in KTE Jertovec is shown in Figure 7.5. The condenser consists of two parts: the shell with pipes and the hot well at the bottom of the shell. Cold water provided by an outside source (river) flows through the pipes and carries the heat away. The condensed water is removed from the hot well by the pump.

7.2.2.1 The Condenser Model

As can be seen from Figure 7.5, due to condenser geometry the cross section area is changing with the condensate level. From the control point of view this should be considered as the main source of process nonlinearity.

In the simulation analysis that follows, an assumption is made such that condensate level dynamics are primarily affected by a flow difference between the inlet steam flow condensed into fluid and the outlet condensate flow, while other effects such as cooling water leakages have been neglected.

Under these assumptions [49], we may write the condensate mass balance equation as

$$\rho \frac{\mathrm{d}V}{\mathrm{d}t} = m_{\text{in}} - m_{\text{out}} \tag{7.2}$$

where ρ is the condensate density (kg/m^3), V the condensate volume (m^3), m_{in} the inlet steam flow (kg/sec), and m_{out} is the outlet condensate flow (kg/sec).

Since the condensate volume is a function of the level, Equation (7.2) can be rewritten as:

$$A(h)\frac{\rho \mathrm{d}h}{\mathrm{d}t} = m_{\text{in}} - m_{\text{out}} \tag{7.3}$$

where h is the condensate level, in m.

$$A(h - h_0) = \begin{cases} A, & \text{for } h \leq h_0 \\ 2L\sqrt{2Rh - h^2}, & \text{for } h > h_0 \end{cases} \tag{7.4}$$

where $A(h)$ is the area under the condensate (m^2), A the hot well area (m^2), L the length of the condenser (m), R is the radius of the condenser (m), and h_0 is the hot well height (m).

From Equation (7.4) it is clear that for the condensate level higher than h_0 mass flow differential equation (7.3) becomes nonlinear.

The water, condensed in the hot well, leaves the condenser through a pump, which has a nonlinear $Q - h$ characteristic. The third source of nonlinearity is a valve placed behind the pump, which controls the outlet condensate flow. Assuming that condensate density is constant and valve cross section vs. valve stroke has a linear characteristic, flow equation assumes the form

$$m_{\text{out}} = \alpha A_{\text{v}}(x)\sqrt{2\rho\Delta p} = K_{\text{v}}\sqrt{\Delta p} \tag{7.5}$$

where A_{v} is the valve cross section (m^2), K_{v} the valve coefficient, x the valve position (m), α the flow coefficient, and Δp is the pressure difference (N/m^2).

The process is modeled in MATLAB® with Equations (7.2) to (7.5) and the nonlinear $Q - h$ characteristic of the pump taken into account. Mass flow m is further replaced with notation Q.

7.2.2.2 Standard Condensate Level Control

Standard condensate level control is shown in Figure 7.6. The level h is measured by a differential pressure gauge and filtered prior to comparison with a reference level h_{ref}. Based on the level error, a TSC sets a voltage polarity of a DC servo motor driving the valve.

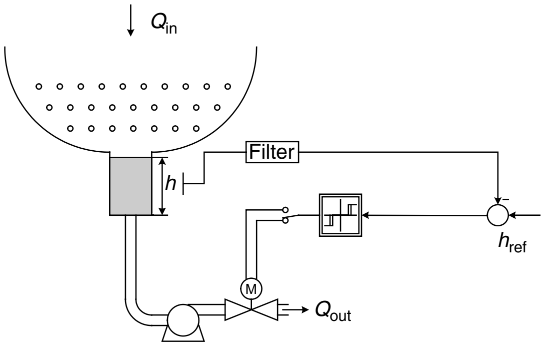


FIGURE 7.6 The standard condensate level control scheme. (From Bogdan, S., Kovačić, Z., Lončar, D., and Lukačević, D., *KoREMA 9th Mediterr. Conf. Contr. Autom.*, Session FM1-A, 2001. With permission.)

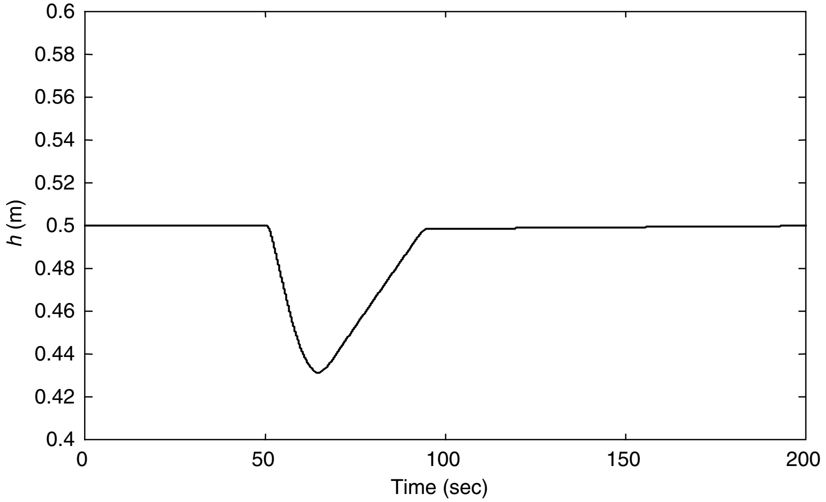


Figure 7.7 The condensate level response for a 5% decrease of steam flow. (From Bogdan, S., Kovačić, Z., Lončar, D., and Lukačević, D., *KoREMA 9th Mediterr. Conf. Contr. Autom.*, Session FM1-A, 2001. With permission.)

The process parameters in the steam power plant at KTE Jertovec, Croatia are as follows: $A = 0.196$ m^2, $L = 4.75$ m, $R = 1.4$ m, $h_0 = 0.51$ m, $h_{\text{ref}} = 0.5$ m. Inlet steam temperature is 450°C. The inlet steam flow rate for the nominal load is $Q_{\text{in}0} = 13.8889$ kg/sec, $x_0 = 40\%$.

The level transient response in case of a 5% negative change in steam flow is shown in Figure 7.7. The standard controller compensates the change of level in approximately 50 sec. Another case is shown in Figure 7.8, where the change in steam flow is positive. It can be seen that the level response is much slower than in the case of a decreasing steam flow. The speed of level change at the beginning is comparable with the one shown in Figure 7.7, but once the condensate passes the line between the hot well and the shell the speed is reduced. The standard controller needs almost 20 min to compensate a change in the level.

By comparing two responses (Figure 7.7 and Figure 7.8) it is evident that system nonlinearities have a significant influence on the control quality.

In order to get a better insight into process dynamics it is interesting to see how the TSC controller handles variations in the steam flow under different operating conditions.

First we test the performance of the standard controller by changing inlet flow in the range ±5% of the operating point value (50 t/h). The level reference point is $h_{\text{ref}} = 0.5$ m. The condensate level and controller output responses are shown in Figure 7.9(b) and Figure 7.9(c). As can be seen from Figure 7.9(b) the level remains within the range 0.45 to 0.54 m. It can be noticed that level dynamics are very fast for $h < h_0$, while for $h > h_0$ the response is much slower.

The second test is related to the situation when flow changes from nominal value to 50% of it and recovers back to the nominal value. The obtained results are shown in Figure 7.10.

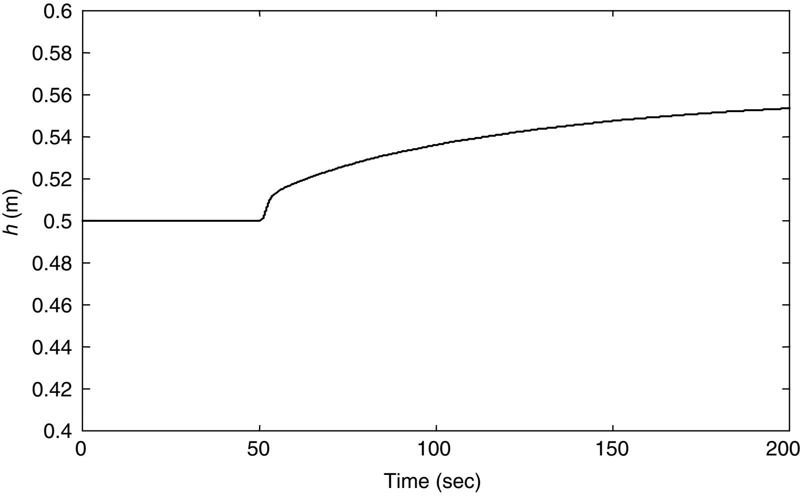


FIGURE 7.8 The condensate level response for a 5% increase of steam flow. (From Bogdan, S., Kovačić, Z., Lončar, D., and Lukačević, D., *KoREMA 9th Mediterr. Conf. Contr. Autom.*, Session FM1-A, 2001. With permission.)

The level drop caused by flow decrease is 9 cm, while during flow recovery level goes up for 9 cm. As for the particular system level high limit is 0.6 m (otherwise alarm occurs and control is suspended) the standard control scheme cannot deal with significant flow change. That is why in such case an operator must switch from automatic to manual mode of operation.

From Figure 7.10(c), one can notice high valve activity, especially during flow recovery. Since frequent switching affects the actuator durability, one of the criteria for controller design is its capability of handling the controlled variable in a desired range with low exertion of the actuator.

7.2.2.3 Fuzzy Gain Scheduling Condensate Level Control

The fuzzy gain scheduling (FGS) level control structure is shown in Figure 7.11. The controller comprises of a gain scheduler, a level derivation estimator, and a controller with two tunable gains. The controller output is connected with the input of the standard TSC. Two additional signals are used: measurement of live steam flow (which is for this particular case equal to inlet steam flow) and measurement of condensate flow. These two measurements already existed in the standard control scheme and were used before only for the monitoring purpose.

The fuzzy gain scheduler has two inputs, the steam flow Q_{in} and the change in steam flow ΔQ_{in}, and two outputs, gain coefficients $K_{\mathrm{d}h}$ and K_{e}. The fuzzy logic-based gain scheduler determines new values of gain coefficients by using a fuzzy rule-table with nine fuzzy rules. Inputs have three fuzzy sets with triangular membership functions (Figure 7.12), while outputs are represented with singletons. Calculation of outputs is performed according to the center of gravity (COG) principle described with Equation (2.22).

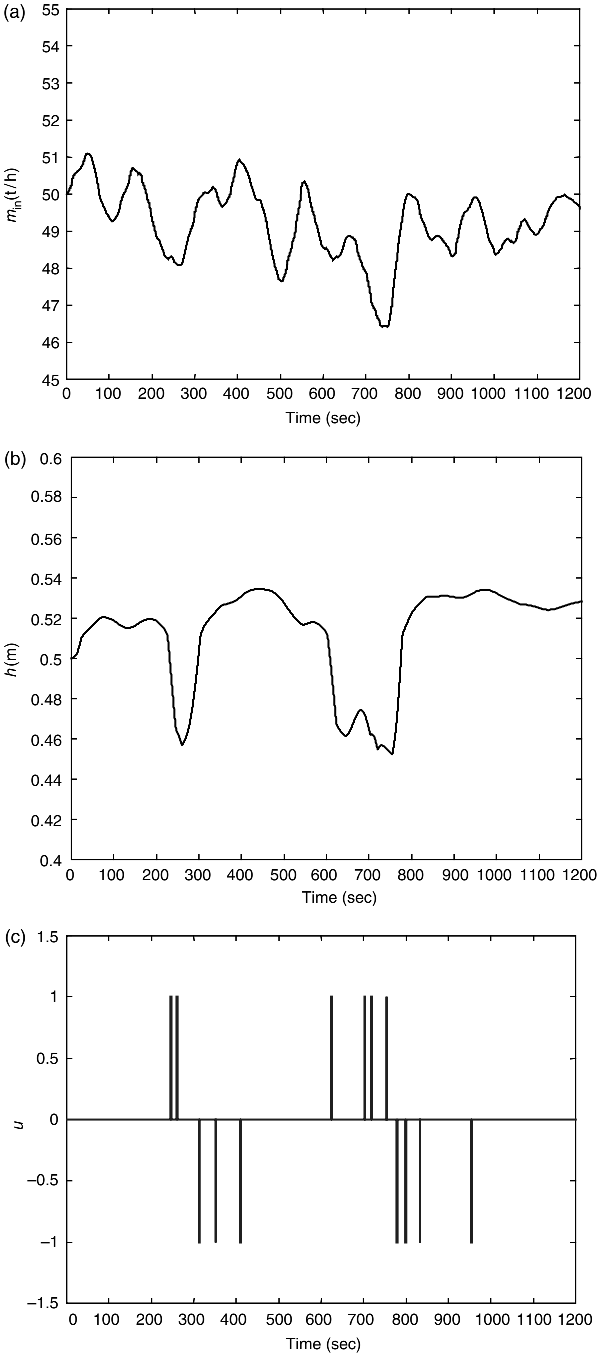


FIGURE 7.9 The level (b) and TSC output (c) responses in case of minor variations of steam flow (a). (From Bogdan, S., Kovačić, Z., Lončar, D., and Lukačević, D., *KoREMA 9th Mediterr. Conf. Contr. Autom.*, Session FM1-A, 2001. With permission.)

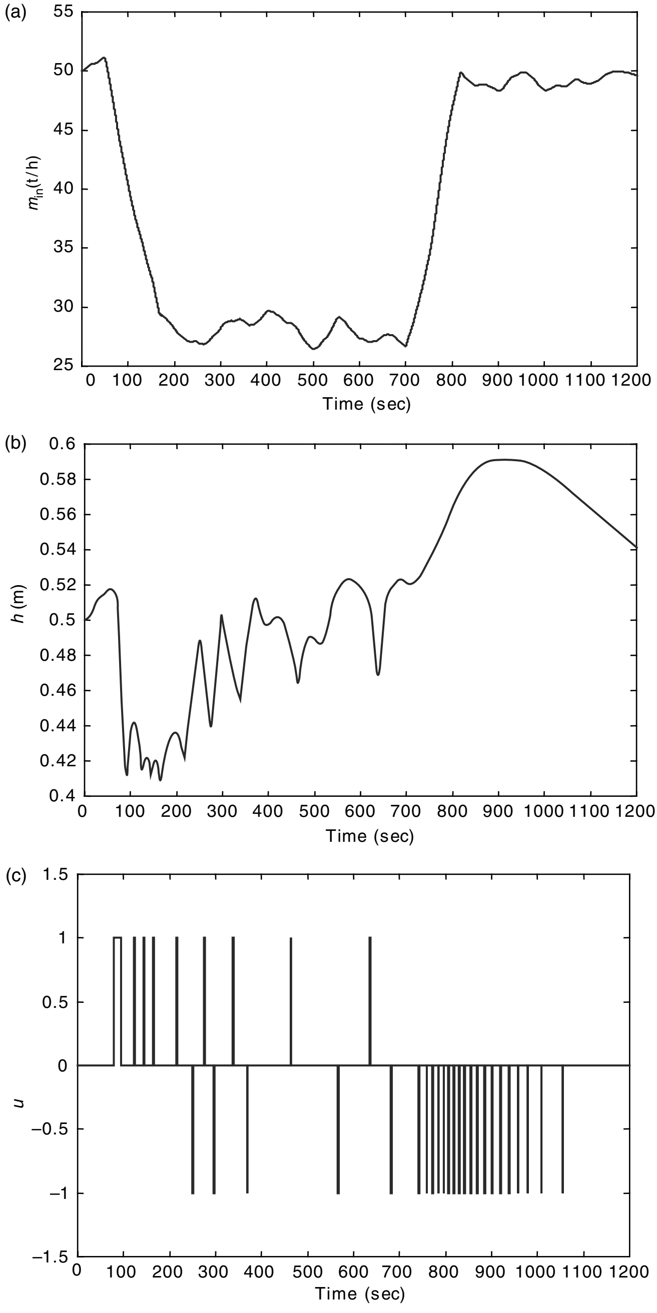


FIGURE 7.10 The level (b) and TSC output (c) responses in case of major variations of steam flow (a). (From Bogdan, S., Kovačić, Z., Lončar, D., and Lukačević, D., *KoREMA 9th Mediterr. Conf. Contr. Autom.*, Session FM1-A, 2001. With permission.)

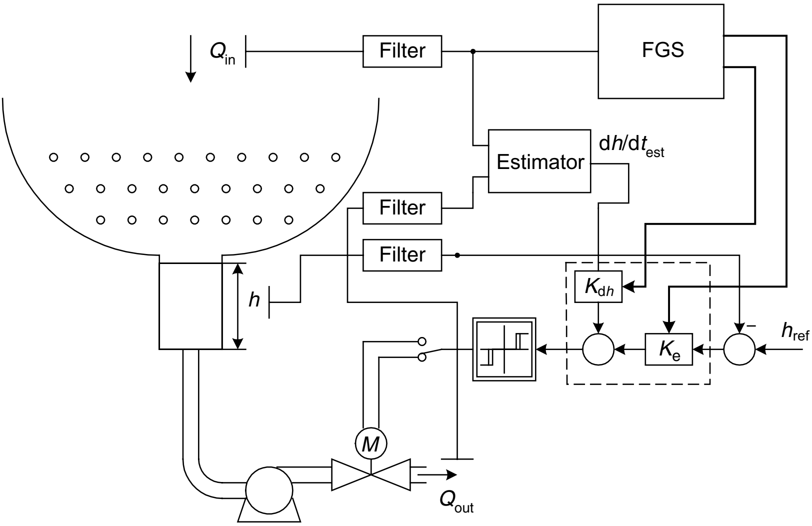


FIGURE 7.11 The FGS condensate level control scheme. (From Bogdan, S., Kovačić, Z., Lončar, D., and Lukačević, D., *KoREMA 9th Mediterr. Conf. Contr. Autom.*, Session FM1-A, 2001. With permission.)

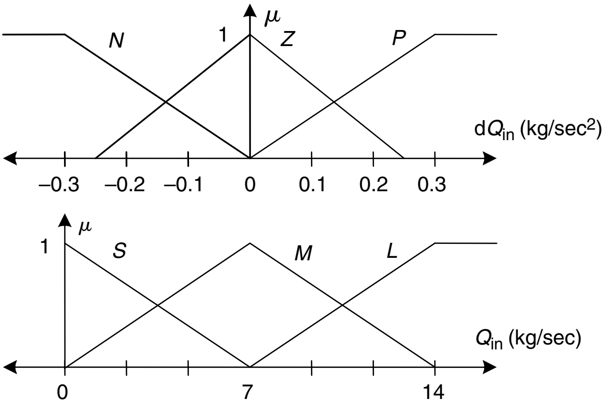


FIGURE 7.12 Fuzzy membership functions for a gain scheduling algorithm. (From Bogdan, S., Kovačić, Z., Lončar, D., and Lukačević, D., *KoREMA 9th Mediterr. Conf. Contr. Autom.*, Session FM1-A, 2001. With permission.)

The estimator calculates the change of the level, dh/dt, based on flow measurements. The calculated value is multiplied with gain K_{dh}, and added to the signal formed from the error between the level reference h_{ref} and the measured level h. All measured variables are filtered with a first-order filter.

The simulation results obtained for the same steam flow variations as in the case of a TSC examination are shown in Figure 7.13.

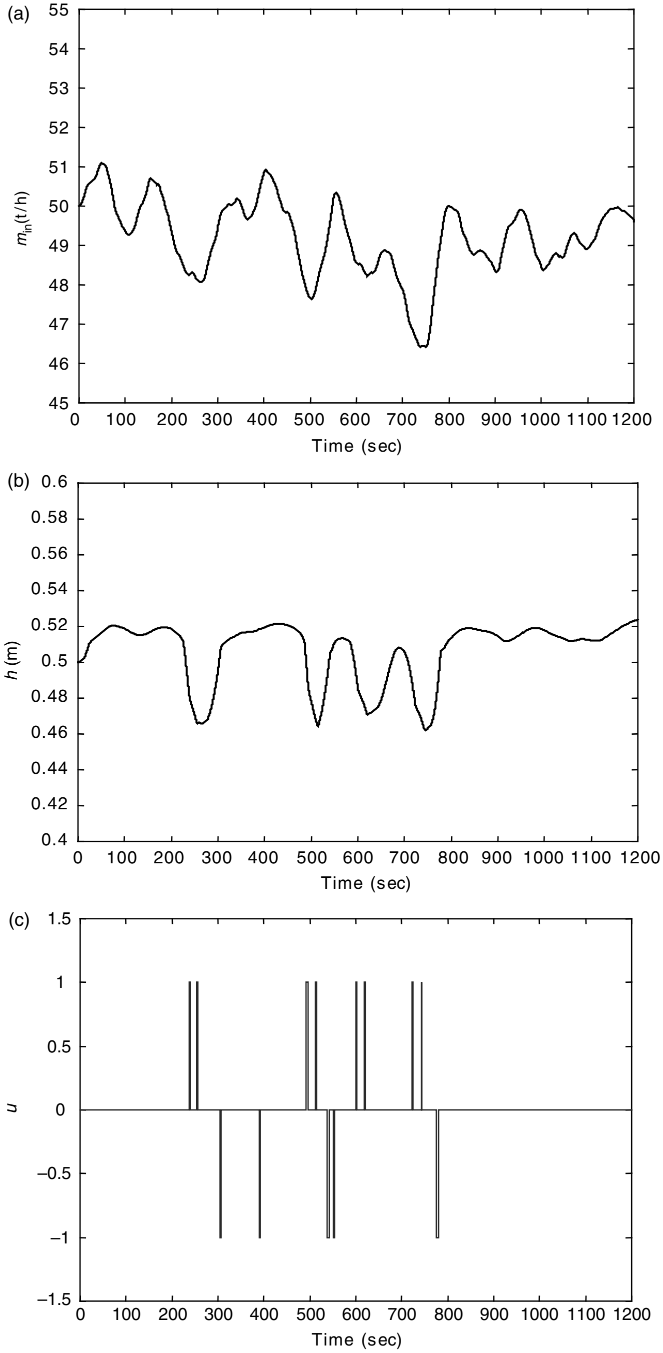


FIGURE 7.13 The level (b) and fuzzy adaptive controller output (c) responses in case of minor variations of steam flow (a). (From Bogdan, S., Kovačić, Z., Lončar, D., and Lukačević, D., *KoREMA 9th Mediterr. Conf. Contr. Autom.*, Session FM1-A, 2001. With permission.)

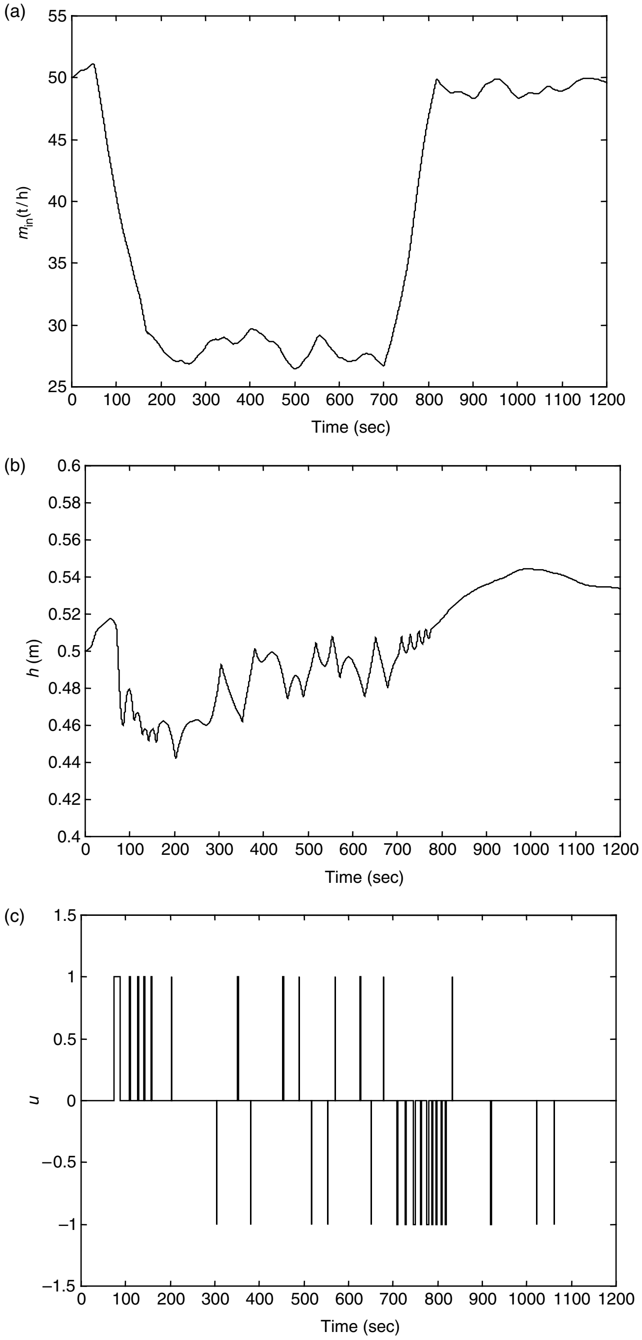


FIGURE 7.14 The level (b) and fuzzy adaptive controller output (c) responses in case of major variations of steam flow (a). (From Bogdan, S., Kovačić, Z., Lončar, D., and Lukačević, D., *KoREMA 9th Mediterr. Conf. Contr. Autom.*, Session FM1-A, 2001. With permission.)

After application of FGS control, variations of condensate level have remained within the range 0.46 to 0.52 m, which is 30% less than in the case of using the TSC (Figure 7.9). Furthermore, the square error integral value with the FGS control is 0.3970, while with the TSC, this value is 0.8855. This is more than 50% reduction without any increase in actuator (servo valve) effort (in both cases the valve was switched 13 times).

As already mentioned, the main problem in standard condensate level control is a significant change of steam flow. The results obtained with FGS control in the case of a large steam flow change are shown in Figure 7.14. One can see that the condensate level drops 6 cm (9 cm with standard control — see Figure 7.10), while in the case of steam flow increase the level reaches 0.545 m (0.59 m with standard control — see Figure 7.10). The value of the integral quality criterion is 1.0035 for FGS control and 3.5320 for standard TSC, which indicates a reduction of more than three times.

In the same time the number of relay switches decreases from 33 in the case of standard TSC to 30 in the case of FGS control. The results of comparison of two methods are shown in Table 7.1.

TABLE 7.1
Comparison of Two Controllers

	Variations Q_{in} = ±5%		Increase/decrease Q_{in} 50%	
Method	**# Switching**	$\int e(t)^2 dt$	**# Switching**	$\int e(t)^2 dt$
TSC controller	13	0.8855	33	3.5320
FGS controller	13	0.3970	30	1.0035

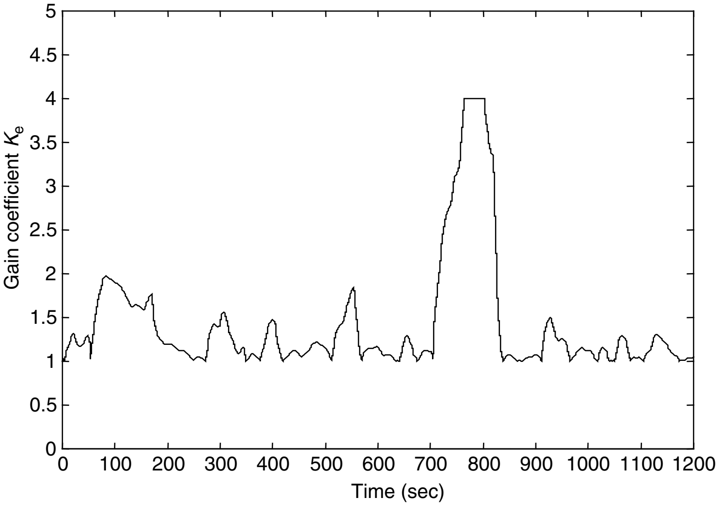


FIGURE 7.15 Transition of gain coefficient K_e during large changes of steam flow. (From Bogdan, S., Kovačić, Z., Lončar, D., and Lukačević, D., *KoREMA 9th Mediterr. Conf. Contr. Autom.*, Session FM1-A, 2001. With permission.)

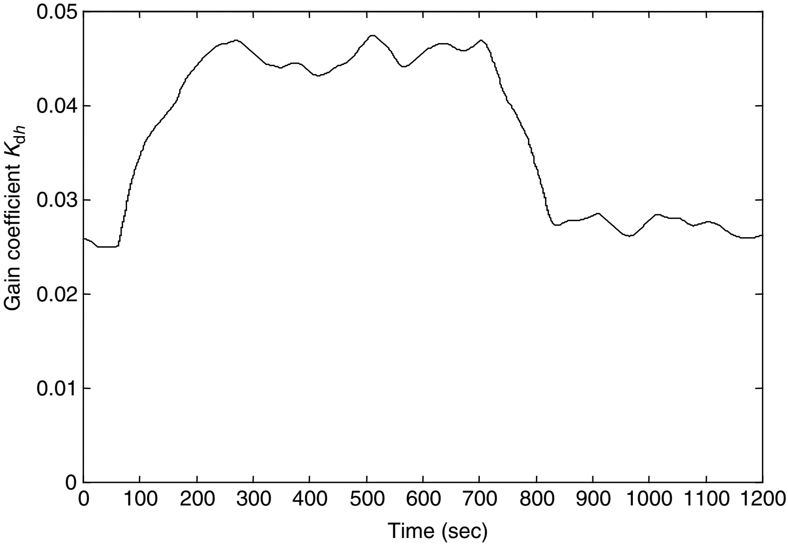


FIGURE 7.16 Transition of gain coefficient K_{dh} during large changes of steam flow. (From Bogdan, S., Kovačić, Z., Lončar, D., and Lukačević, D., *KoREMA 9th Mediterr. Conf. Contr. Autom.*, Session FM1-A, 2001. With permission.)

The scheduling of gain coefficients K_{e} and $K_{\mathrm{d}h}$ is shown in Figure 7.15 and Figure 7.16, respectively.

Presented results and field tests showed that the standard TSC loop was unable to hold the level in the desired region in case of a significant change in inlet steam flow.

7.2.2.4 PLC Siemens Simatic S7-216 Step 7 Program of FGS Condensate Level Control

The FGS control structure has been implemented with an industrial PLC Siemens Simatic S7-216 and successfully tested in the KTE Jertovec.

The structure of the PLC program that executes a FGS control algorithm is shown in Figure 7.17. It is followed by a PLC program written in Step 7 (trademark of Siemens) programming tool that is presented in its original PLC project form.

Experimental results obtained in the plant under the same operating conditions as for the standard TSC confirmed that the usage of FGS controller indicatively improved control quality while keeping the lowest possible number of actuator switching.

7.2.3 PLC-Based Self-Learning Fuzzy Controller Implementation

The goal of any new control methodology is to get successfully transferred into the control practice. Only real control applications can confirm a real value of new control methods. Having that in mind, herein we describe an

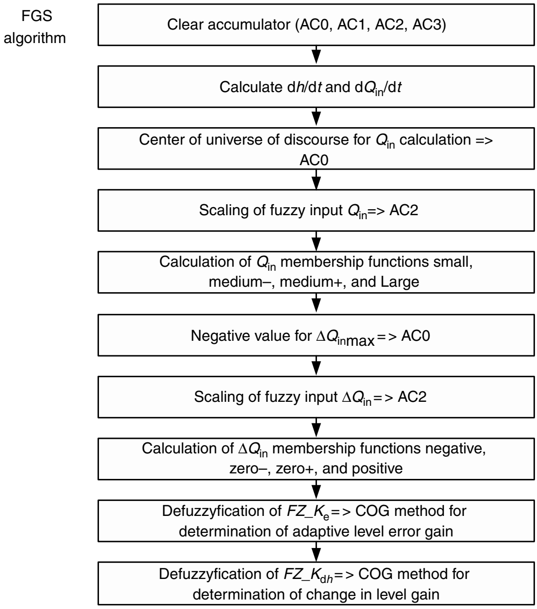
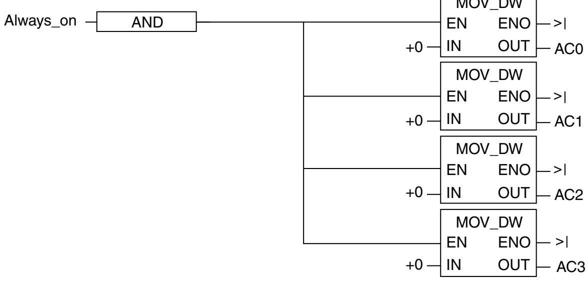

FGS algorithm
Clear accumulator (AC0, AC1, AC2, AC3)
Calculate dh/dt and dQ_{in}/dt
Center of universe of discourse for Q_{in} calculation => AC0
Scaling of fuzzy input Q_{in}=> AC2
Calculation of Q_{in} membership functions small, medium–, medium+, and Large
Negative value for ΔQ_{inmax} => AC0
Scaling of fuzzy input ΔQ_{in} => AC2
Calculation of ΔQ_{in} membership functions negative, zero–, zero+, and positive
Defuzzyfication of FZ_K_e => COG method for determination of adaptive level error gain
Defuzzyfication of FZ_K_{dh} => COG method for determination of change in level gain

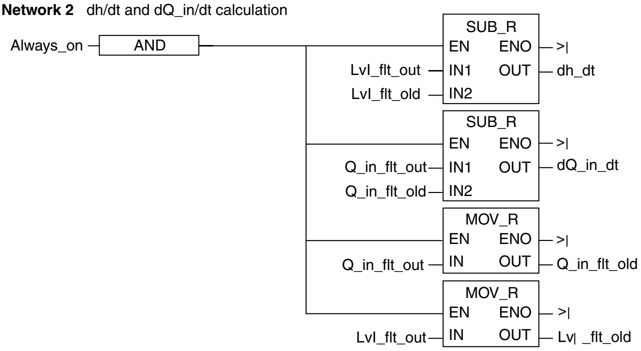
FIGURE 7.17 Flow-chart of the PLC-based fuzzy gain scheduling control algorithm.

Here follows a PLC program presented in its original PLC project written in Siemens step 7

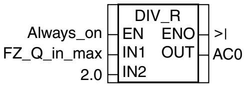
Network 1 clear accumulator



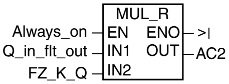
Network 2 dh/dt and dQ_in/dt calculation



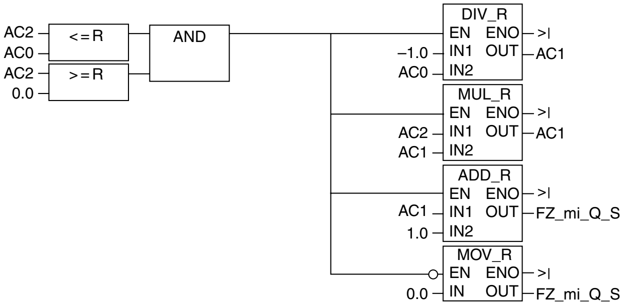
Network 3 Center of universe of discourse for Q_in calculation => AC0



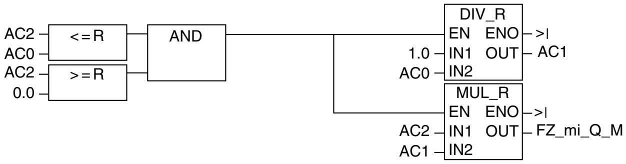
Network 4 fuzzy input scaling => AC2



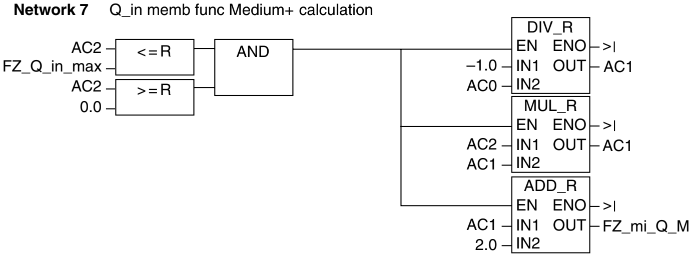
Network 5 Q_in memb func small calculation



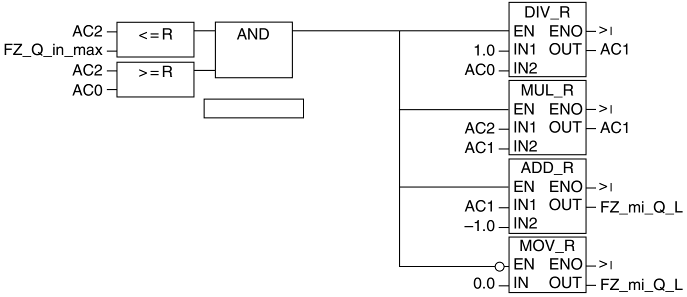
Network 6 Q_in memb func medium– calculation



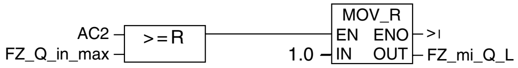
Network 7 Q_in memb func Medium+ calculation



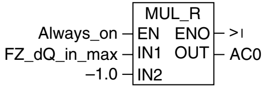
Network 8 Q_in memb func Large calculation



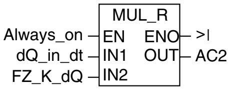
Network 9 Q_in memb func Large calculation



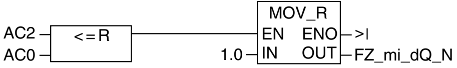
Network 10 Negative value of dQ_in_max calculation=>AC0



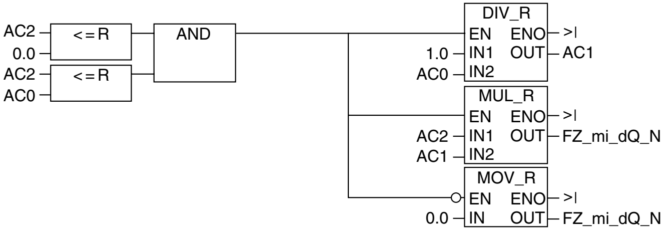
Network 11 Fuzzy input scaling => AC2



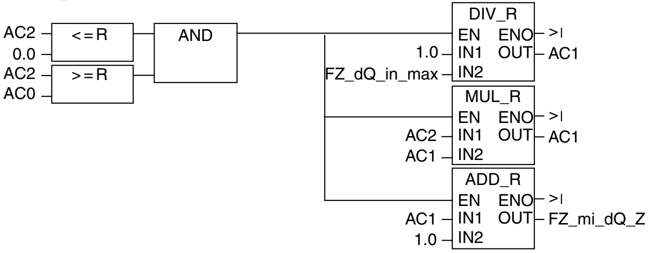
Network 12 dQ_in memb func Negative calculation



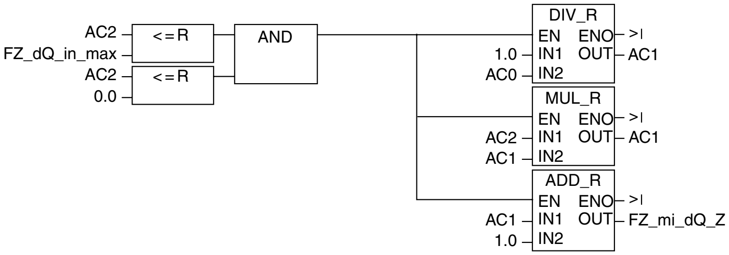
Network 13 dQ_in memb func Negative calculation



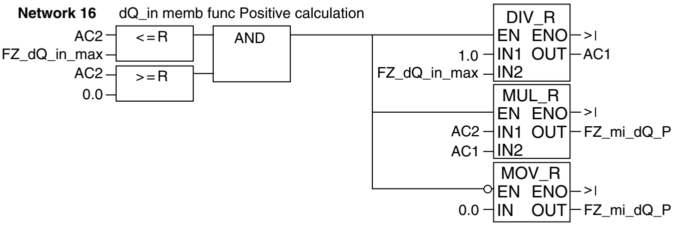
Network 14 dQ_in memb func Zero– calculation



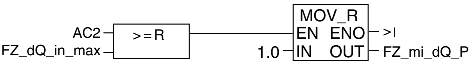
Network 15 dQ_in memb func Zero+ calculation



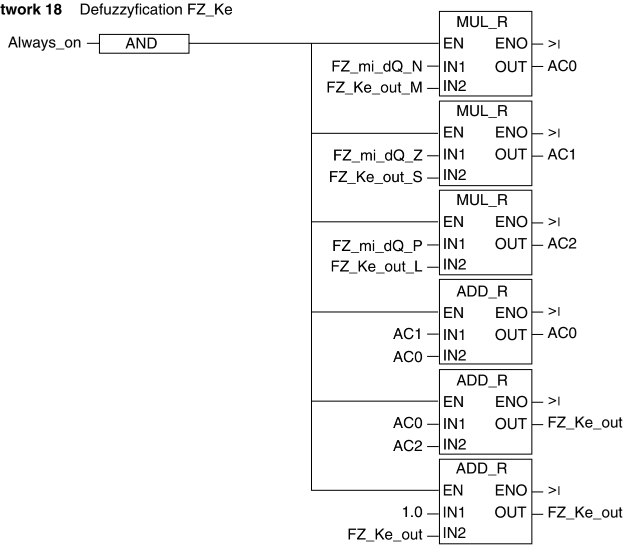
Network 16 dQ_in memb func Positive calculation



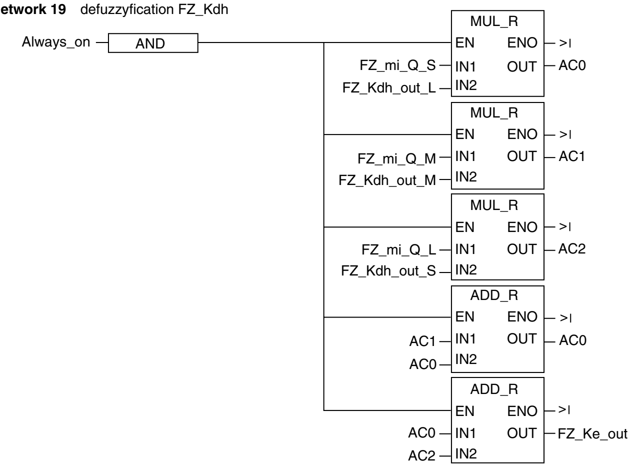
Network 17 dQ_in memb func Positive calculation



Network 18 Defuzzyfication FZ_Ke



Network 19 defuzzyfication FZ_Kdh



implementation of the sensitivity model-based self-organizing fuzzy controller described in Section 5.3, implemented in the form of a function block for a soft PLC. Besides self-organization, the block also contains an algorithm for a phase plane-based presetting of the fuzzy control surface described in Section 3.3. A newly formed function block called phase plane-based self-organizing fuzzy controller (PPSOFC) is programmed with a soft PLC programming tool IsaGRAF in accordance to the international standard IEC61131-3 [50]. Since we deal here with a controller function block designed for open PLCs and industrial PCs, we pay more attention to the function block structure and its parameters as well as to the description of several modes of controller operation. Since multimode operation requires carefully planned control overhead, we provide detailed explanations of each control mode and conditions for proper switching of modes.

The concept of the PPSOFC function block assumes operation with floating point instructions due to the complexity of the control algorithm and specifications on the accuracy of control.

7.2.3.1 PPSOFC — Self-Organizing Fuzzy Controller Function Block

The control practice says that many nonlinear high-order systems are controlled with some form of a PID controller. Being aware of that, we logically come to the conclusion that integration of a standard PID controller into a self-learning fuzzy controller block PPSOFC is a smart move. This opens the possibilities to use the PPSOFC block as a direct replacement for the already installed PID controllers. Once we have a PID controller within the block, all necessary conditions for the implementation of the phase plane-based presetting algorithm are there (see Section 3.3). As discussed in Section 3.3, the phase plane-based presetting method requires online recording and analysis of controller inputs and output values during a regular PID control of the target system. Thus obtained error phase plane trajectories and corresponding control curves are then used for creation of a fuzzy control surface that should mimic the original PID control.

The operation of the PPSOFC block depends on the second-order reference model Equation (3.28), which describes a desired dynamic behavior of the target system (see Section 3.2). In the PPSOFC block a second-order reference model is defined with five parameters:

- Magnitude of the imposed change of reference model input $\Delta u_{\mathrm{M}} = \Delta u_{\mathrm{r}}$
- Magnitude of the change of reference model output $\Delta y_{\mathrm{M}} > 0$
- Magnitude of the maximum change (peak value) of reference model output Δy_{Mm}
- For oscillatory model responses: peak time of the reference model output t_{m}
- For aperiodic model responses ($\Delta y_{\mathrm{M}} = \Delta y_{\mathrm{Mm}}$): settling time of the reference model output t_{s}
- Control interval T_{d}

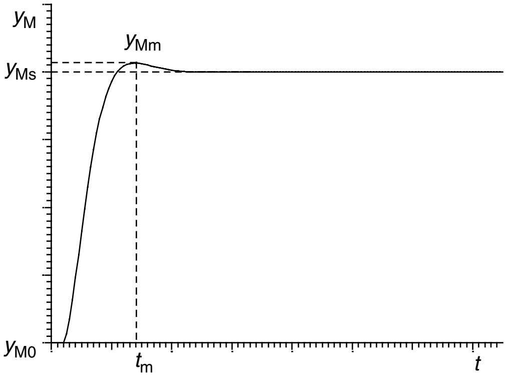


FIGURE 7.18 The reference model response.

The way of setting the reference model parameters is illustrated with the example shown in Figure 7.18. Parameter values provided by the user are as follows: $\Delta u_M = \Delta u_r = 100$, $\Delta y_M = y_{Ms} - y_{M0} = 200$, $\Delta y_{Mm} = y_{Mm} - y_{M0} = 210$ (5% overshoot), $t_m = 4.5$ sec, and $T_d = 0.2$ sec. These values are used for calculation of second-order model parameters: $a_{M1} = 1.7201$, $a_{M2} = -0.7524$, and $b_{M1} = 0.0646$.

The substitution of the PID control with the self-learning fuzzy control should not jeopardize the continuity of real-time process control; special attention must be paid to bumpless switching between different operating modes. Integration of both linear PID and self-organizing fuzzy controllers within the complex function block PPSOFC ensures elegant switching between two controllers without abrupt changes of the control value. This should help to make the function block PPSOFC attractive for standard PID controller users.

The outlook of the PPSOFC function block is shown in Figure 7.19. As mentioned above, the block has several operating modes, which depend on the states of external control signals. We assume that the block is connected to the outer world via 12-bit A/D and D/A converters. All signals and parameters of the PPSOFC block are displayed in Tables 7.2 and 7.3.

The PPSOFC function block should be initially put up into manual mode of operation. Recommended initial settings of PPSOFC inputs, outputs, and parameters are displayed in Table 7.4.

Setting of reference model parameters depends on acquired knowledge about the dynamics of the controlled process. Dynamic characteristics of typical process variables in different industry applications are shown in Table 7.5. If the user does not have enough knowledge about the process, it may happen that inappropriate values of the reference model parameters are entered. In such case, it is advised to start setting of the model with larger values of the peak time or settling time, as they can be gradually decreased during the progress of learning. In addition, setting of aperiodic reference model response or response with a lower overshoot will make

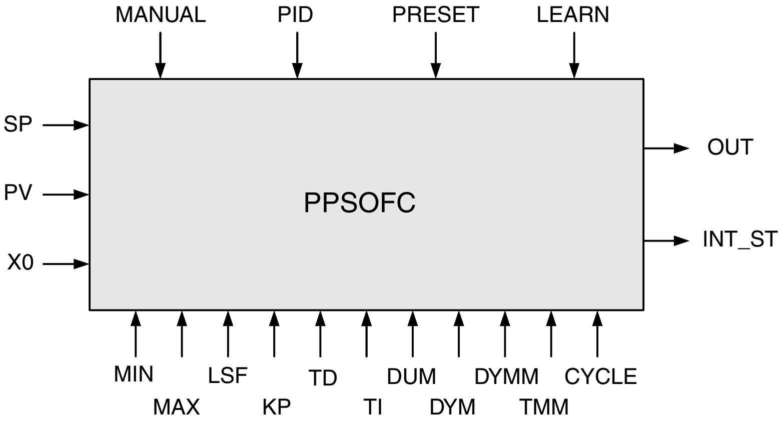


FIGURE 7.19 Soft PLC function block PPSOFC.

TABLE 7.2
PPSOFC Signals

Signal	Characteristics
MANUAL (C)	Manual control
PID (C)	Standard PID control with user-defined controller parameters
PRESET (C)	Preset fuzzy control or phase plane-based presetting of a fuzzy control surface in conjunction with the PID signal
LEARN (C)	Learning (self-organization) of fuzzy controller. A sequence of varying set point values is expected
X0 (I)	Manually set control output → OUT = X0. It can assume values from MIN to MAX
SP (I)	Set point expressed as a 12-bit number. It can assume values from 0 to 4095
PV (I)	Process variable (feedback signal) expressed as a 12-bit number. It can assume values from 0 to 4095
OUT (O)	Controller output expressed as a 12-bit number. It can assume values from MIN to MAX
INT_ST (O)	Internal status: 0 — manual, 1 — PID preset, 2 — PID, 3 — preset fuzzy, 4 — learn, and 5 — self-organizing fuzzy control

C: control; I: input; O: output.

the convergence of learning process slightly faster (because of faster convergence of singletons, which lie close to the phase plane origin, i.e., ZE–ZDY area).

Each operating mode of the controller is determined by the current status of control inputs MANUAL, PID, PRESET, and LEARN. These control inputs have different levels of priority, as shown in Table 7.6. The control signal MANUAL

TABLE 7.3
PPSOFC Parameters

Parameter	Explanation
CYCLE	Controller interval
MAX	Upper limit for the control signal, MAX $\leq$ 100
MIN	Lower limit for the control signal, 0 $\leq$ MIN $<$ MAX
LSF	Learning safety factor, whose value is directly related to the estimated range of process gain coefficient variations in order to ensure the stability of learning. An increase of LSF slows down the speed of learning
DUM	Change of the reference model input
DYM	Change of the reference model output ($>$0)
DYMM	Maximum change of the reference model output
TMM	Peak time of the reference model output. For aperiodic model response DYM = DYMM, TMM is a settling time of the reference model output
KP	Gain coefficient of a PID controller
TI	Integration time of a PID controller
TD	Derivation time of a PID controller

TABLE 7.4
Recommended Initial Status of the PPSOFC Function Block

Parameter	Explanation
MANUAL	TRUE
PID	FALSE
PRESET	FALSE
LEARN	FALSE
X0	0
CYCLE	From TMM/5 to TMM/10 (or from TMM/25 to TMM/50 for aperiodic response)
MAX	100
MIN	0
LSF	From 10 to 30
DUM = DYM	100 (the reference model gain coefficient is mostly set to 1)
DYMM	From 100 (suggested) to 130 (from 0 to 30% overshoot in response)

has the highest priority in order to give the user full control over the controller and the entire control system. The control signal PID has the second highest priority, which means that the active state of the signal PID enforces PID control regardless from the current states of control signals PRESET and LEARN. The control signal PRESET has the second lowest priority, while the control signal LEARN has the

TABLE 7.5
Dynamic Characteristics of Typical Control Objects (Processes)

Process	Transport delay	Time constant
Temperature		
Small oven	0.5–1 min	5–15 min
Big oven	1–3 min	10–20 min
Distillation column	1–7 min	40–60 min
Steam heater	cca 2 min	—
Room heating	1–5 min	10–60 min
Pressure		
Gas pipeline	—	0.1 sec
Masoot steam boiler	—	cca 150 sec
Flow		
Pipe	0–5 sec	0.2–10 sec
Level		
Steam boiler	0.5–1 min	—
Angular speed		
Small electric drive	—	0.2–10 sec
Large electric drive	—	5–40 sec
Voltage		
Small generator	—	1–5 sec
Large generator	—	5–10 sec

TABLE 7.6
Levels of Priority of the PPSOFC Control Inputs

Level of priority	Control input
1	MANUAL
2	PID
3	PRESET
4	LEARN

lowest priority of all. This means that active status of the signal PRESET can interrupt the learning process at any time during the automatic mode of operation and enforce system control with a preset fuzzy controller.

Although operating modes are normally determined by the current status of control signals, one of available operating modes, so called *user-authorized PID control and phase plane-based presetting* mode of operation will be activated only by following a certain protocol. The main reason for introduction of such

TABLE 7.7
Modes of Operation in the PPSOFC Functional Block

Manual	PID	Preset	Learn	Mode of operation
TRUE	T/F	T/F	T/F	Manual control (INT_ST = 0)
FALSE	FALSE	FALSE	FALSE	Routine self-organizing fuzzy control (INT_ST = 5): If both the user-authorized mode and self-organizing mode have not been started, then the controller output will contain only a feedforward component (open-loop)
FALSE	FALSE	TRUE	T/F	Preset fuzzy control (INT_ST = 3): If the user-authorized phase plane-based presetting mode has not been started yet, then the controller output will contain only a feedforward component (open-loop)
FALSE	FALSE	FALSE	TRUE	Self-organizing fuzzy control (INT_ST = 4) starting from a blank fuzzy rule-table or the last preset fuzzy rule-table (depending on whether PRESET was active before or it was not)
FALSE	TRUE	T/F	T/F	PID control (INT_ST = 2)
FALSE	TRUE	TRUE	T/F	User-authorized PID control and phase plane-based presetting (INT_ST = 1) (calculation) of a fuzzy control surface (conditional, i.e., if preceded by TRUE-to-FALSE transition of the MANUAL control signal). Before initiating the presetting procedure, a steady-state of a PID controlled system must be achieved and maintained for some time

a protocol is the online character of the phase plane-based presetting of a fuzzy control surface. Namely, the presetting of the fuzzy control surface is always performed during closed-loop control, while a request for presetting must be issued by the user, that is, in the manual (open-loop) control mode. Since very often it may happen that the user has defined different values of the set point input SP and the manually set operating point PV (although they are expected to be the same), switching from open-loop control to closed-loop PID control will then provoke a transient response of the system output. Without any protocol, the difference SP−PV could affect the system response internally induced by the stepwise change of the set point for the purpose of phase plane-based presetting of the fuzzy control surface.

In order to provide a quick reference to available operating modes, a spectrum of operating modes and accompanying conditions is displayed in Table 7.7. Some modes have the same characteristic, which has finally resulted in one manual and five automatic operating modes available to the user.

A more detailed description of the protocol for selection of operating modes available in the PPSOFC function block is presented along with further explanations related to possible industry applications.

Manual Control (Open-Loop), MANUAL = TRUE

In the *manual mode (INT_ST = 0)* of control the value of the control output OUT is determined by the value of the external input X_0, which is directly supplied by the user. The control signal MANUAL, due to the highest level of priority, makes other control signals in this mode completely ineffective. Each time when the *manual mode* is reentered and abandoned, the reference model parameters that are used in a model reference-based learning mechanism are recalculated.

User-Authorized PID Control and Phase Plane-Based Presetting of the Fuzzy Controller, First FALSE-to-TRUE Transition of MANUAL, then PID = PRESET = TRUE, LEARN = FALSE or TRUE, then TRUE-to-FALSE Transition of MANUAL while Keeping PID = PRESET = TRUE

The user enters the PID control and phase plane-based presetting mode (INT_ST = 1) by switching to manual mode (FALSE-to-TRUE transition of the MANUAL signal) and by setting control signals PID = PRESET = TRUE. After switching to automatic mode (and keeping PID = PRESET = TRUE), true states of PID and PRESET will initiate PID control and successive phase plane-based presetting of the fuzzy controller. The presetting procedure requires generation of internal perturbation of the set point (with respect to the manually set process value PV) for induction and recording of error phase plane trajectories and corresponding control output curves, used in direct reconstruction of the fuzzy control surface. After completion of the fuzzy controller presetting procedure, the PID controller will stay in full control of the system (switching to INT_ST = 2).

It must be noted that each request for this mode of operation has to be authorized by the user (i.e., it must be issued from the manual mode). In the PID and phase plane-based presetting control mode, interrupting the execution of presetting procedure, before it is completed (i.e., before a PID controller overtakes system control) is banned.

PID Control (Closed-Loop), MANUAL = FALSE, PID = TRUE, PRESET and LEARN = TRUE or FALSE

In the *PID control mode (INT_ST = 2)* closed-loop control is performed by a PID controller with user-defined parameters. Setting of PRESET or LEARN signal into the true state will be ignored, because of higher priority level of the PID control signal. If the user has not defined any of the PID controller parameters, then the controller output will remain constant (equal to the last control value OUT).

Self-Organizing Fuzzy Control (Closed-Loop), MANUAL = PID = PRESET = FALSE, LEARN = TRUE

The self-organizing fuzzy control mode (INT_ST = 4) is entered only when the lowest priority control signal LEARN is active and permitted. There are several possible ways to enter this mode: directly from the *manual mode* when self-organizing fuzzy control will always continue from a blank fuzzy rule-table; after being in the *user-authorized PID and presetting mode* when self-organizing fuzzy control will continue from the preset fuzzy rule-table; and finally, it may be entered or reentered from other operating modes (PID, preset, or self-learning) and self-organizing fuzzy control will continue from the fuzzy rule-table, which was last active.

If the *self-organizing fuzzy control mode* was interrupted by setting the control signal PRESET into the true state, then the fuzzy rule-table obtained during self-organization would be cleared (lost) and the latest preset fuzzy rule-table would overtake control.

Preset Fuzzy Control (Closed-Loop), MANUAL = PID = FALSE, PRESET = TRUE, LEARN = TRUE or FALSE

The user would enter the preset fuzzy control mode (INT_ST = 3) if it was preceded at least once by the user-authorized PID and presetting control mode. In such case, closed-loop system control is performed according to the fuzzy rule-table that was created and stored during last presetting operation. Setting of LEARN signal into the true state does not have any effect due to its lower level of priority. If the *user-authorized PID and presetting control mode* has not been started yet, then the controller output will contain only a feedforward component (i.e., open-loop control).

If the *preset fuzzy control mode* was entered during the *self-organizing fuzzy control mode*, then the fuzzy rule-table obtained during self-organization would be cleared and the preset fuzzy rule-table would overtake control.

Routine Self-Organizing Fuzzy Control (Closed-Loop), MANUAL and PID and PRESET and LEARN = FALSE

The user may enter the *routine self-organizing fuzzy control mode (INT_ST = 5)* either after completion of the *self-organizing fuzzy control mode (INT_ST = 4)* or after abandoning all other operating modes. In such case, closed-loop control is performed by the “frozen” form of the self-organizing fuzzy controller. If the *self-organizing fuzzy control mode* has not been started at all, then the fuzzy controller output will contain only a feedforward component featuring open-loop control.

Depending on the tolled states of control signals PID, PRESET, and LEARN, *the routine fuzzy control mode* can switch again to the *PID*, *preset*, or *self-organizing mode* of operation.

7.3 Examples of Fuzzy Controller Applications in Process Control

In Section 7.2 we have treated fuzzy logic applications from the implementation technology point of view. The presented application examples have illustrated many possibilities and potential advantages of each implementation platform.

As already mentioned, fuzzy control has proved to be very effective in numerous applications in most different application areas. In this chapter, we present two fuzzy process control solutions. The first one deals with the problem of controlling the road tunnel ventilation system in the tunnel at Učka, Croatia, that connects Northern Adriatic region Kvarner Bay with Istria. The second one describes the usage of fuzzy control for control of anesthesia, so important during demanding surgical operations, which is normally performed by a highly-skilled human operator — anesthetist.

7.3.1 PC-Based Fuzzy-Predictive Control of a Road Tunnel Ventilation System

Vehicles passing through a tunnel produce various types of poison gasses as well as soot, especially in the case of heavy vehicles with diesel engines [51]. With new legislations and demands from tunnel users who are concerned for their health and safety, more and more sophisticated equipment needs to be installed in the tunnel: video cameras, refined traffic-sensing devices, more reliable fire detection systems, and high-sensitive pollution sensors [52]. High standards for air quality and the need for good visibility require an advanced ventilation system for management and control of pollution.

Two objectives, opposite in nature, have to be fulfilled simultaneously by the ventilation system (a) the system should keep visibility (opacity, OP) on a required level and make certain that pollutants remain within admissible margins and (b) energy (costs) used for objective (a) should be minimal. Under some circumstances it is difficult to meet both objectives concurrently by using simple control algorithms; hence, recently, the procedures that combine artificial intelligence and predictive control are implemented for system supervision [53–55]. Usually, together with these new algorithms, longitudinal ventilation is used, mostly due to its acceptable cost.

As a solution of the above defined control problem, we describe a fuzzy-predictive control scheme that includes a feedforward loop based on the traffic and weather data. The aim of the fuzzy-predictive controller is to replace the existing ventilation controller in the tunnel at Učka, Croatia [63].

7.3.1.1 The Structure of a Fuzzy-Predictive Controller

The structure of a fuzzy-predictive controller is shown in Figure 7.20. As can be seen, the structure consists of predictive, fuzzy and jet-fans controllers, pollutants measurement, and a reference generation module.

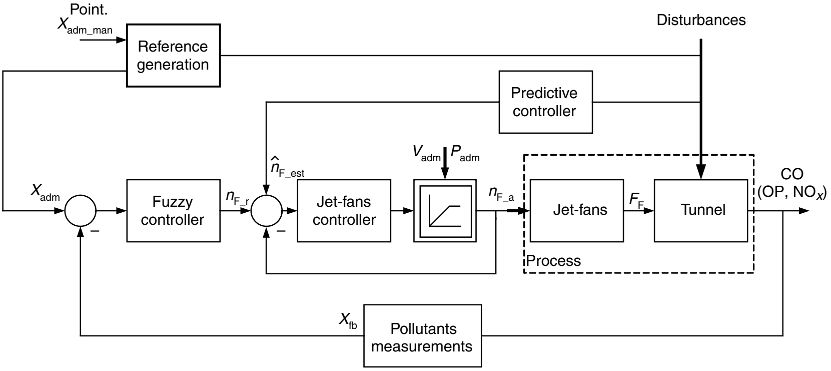


FIGURE 7.20 A structure of the fuzzy-predictive controller. (From Bogdan, S. and Birgmajer, B., *IEEE Intl. Sympos. Industr. Electron. ISIE 2005*, 4, 151–156, 2005. With permission.)

Since the tunnel ventilation system simultaneously takes care of three different pollutants (CO, NO$_x$, and soot), the reference generation module determines the actual set point of the control system. The pollutant that differs most from its required level overtakes the set point. The operator is allowed to override the automatic generation of the system set point.

The task of the *predictive controller* is to determine required air flow, which depends on traffic type, traffic intensity, and weather conditions. Based on that prediction, an estimation of a number of jet fans, $\hat{n}_{F\_est}$, which would produce a thrust force sufficient to provide calculated air flow, is carried out. Since the tunnel model, which is a part of the predictive controller, describes the real tunnel only to some extent, the *fuzzy controller* compares the required level of pollutant, X_{adm}, with the measured value, X_{fb}, and adjusts the jet-fans prediction in order to keep the pollution close to the predefined level. The fuzzy controller output, n_{F_r}, and predictive controller output, $\hat{n}_{F\_est}$, are fed into the *jet-fans controller*. Their sum is compared with the current number of active jet-fans, n_{F_a}. In case $n_{F\_r} + \hat{n}_{F\_est} > n_{F\_a}$, the controller sends a request for a jet-fan switch-on; in case $n_{F\_r} + \hat{n}_{F\_est} < n_{F\_a}$, the controller sends a request for a jet-fan switch-off. As the energy consumption of the ventilation system is significant, care must be taken when switch-on requests are sent to jet-fans. In addition, air velocity within the tunnel should not rise above an adequate level. These two restrictions, consumed energy and air velocity, limit the number of currently active jet-fans.

7.3.1.2 Air Flow Prediction

The first step in air flow prediction is calculation of amounts of CO, NO$_x$, and small, visibility-reducing particles produced by traffic. These amounts depend on several parameters such as speed and type of vehicles, tunnel length, tunnel altitude, etc. A total CO produced by vehicles in the tunnel can be calculated as:

$$Q_{CO\_est} = q_{CO} \cdot N \cdot L \cdot k_{aCO} \cdot k_{gCO} \cdot k_{sCO} \frac{p_0}{p} \cdot \frac{T}{T_0} \tag{7.6}$$

where q_{CO} is the CO produced by a vehicle, m^3/veh/km, N the number of vehicles per hour, veh/h, L the tunnel length, km, $k_{\mathrm{aCO}}, k_{\mathrm{gCO}}, k_{\mathrm{sCO}}$ the correction factors for altitude, gradient, and speed, respectively, p_0 the normal pressure (1013 hPa), p the atmospheric pressure, hPa, T_0 the normal temperature (273 K), and T is the atmospheric temperature, K.

Productions of other two pollutants, NO$_x$ and small particles are determined by the following relations:

$$Q_{\mathrm{NO}_x\_\mathrm{est}} = q_{\mathrm{NO}_x} \cdot (N_{\mathrm{L}} + 10 \cdot N_{\mathrm{H}}) \cdot L \cdot k_{\mathrm{gNO}_x} \tag{7.7}$$

$$M_{\mathrm{p\_est}} = m_{\mathrm{p}} \cdot (N_{\mathrm{H}} + 0.08 \cdot N_{\mathrm{L}}) \cdot L \cdot k_{\mathrm{ap}} \cdot k_{\mathrm{gp}} \tag{7.8}$$

where q_{NO_x} is the NO$_x$ produced by a vehicle (m^3/veh/km), N_{L} the number of light vehicles per hour (veh/h), N_{H} the number of heavy vehicles per hour (veh/h), k_{gNO_x} the correction factor for gradient, m_{p} the mass of particles produced by a heavy vehicle (mg/veh/km), and k_{ap}, k_{gp} are the correction factors for altitude and gradient, respectively.

Once pollutants productions are known, the predictive controller determines a required air velocity as

$$v_{\mathrm{a\_est}} = \frac{Q_{\mathrm{a\_est}}}{A_{\mathrm{T}}} \tag{7.9}$$

where A_{T} is the tunnel cross section in square meters. The fresh air flow, $Q_{\mathrm{a\_est}}$, is calculated as

$$Q_{\mathrm{a\_est}} = \max\left(\frac{M_{\mathrm{p\_est}}}{M_{\mathrm{p\_adm}}}, \frac{Q_{\mathrm{CO\_est}} \cdot 10^6}{\mathrm{CO}_{\mathrm{adm}}}, \frac{Q_{\mathrm{NO}x\_\mathrm{est}} \cdot 10^6}{NOx_{\mathrm{adm}}}\right)$$

with $M_{\mathrm{p\_adm}}$ as admissible concentration of small particles (mg/m^3), CO$_{\mathrm{adm}}$ as admissible concentration of CO (ppm), and NO$_{x\mathrm{adm}}$ as admissible concentration of nitrogenous gases (ppm).

7.3.1.3 Prediction of Number of Jet-Fans

In order to predict a number of jet-fans required to provide a pressure rise which would establish an estimated air velocity, all forces that impact air mass within the tunnel should be taken into account. The *piston effect force*, F_{pist}, caused by a vehicle drag, has the largest influence on the air flow. Although not so significant, forces caused by *tunnel wall friction*, F_{f}, *portal pressure difference*, F_{p}, and *inlet portal losses*, F_{in}, must be integrated into the calculation for an accurate estimation of *jet-fans force*, F_{jet}. When these forces are in balance, that is, the sum of all five forces is zero; the air mass within the tunnel has a constant velocity.

The piston effect force for heavy and light vehicles can be calculated as

$$\begin{aligned} F_{\text{pist\_H}} &= N_{\text{H}} \cdot C_{\text{dH}} \cdot A_{\text{H}} \cdot \frac{\rho_{\text{a}}}{2} \cdot |v_{\text{H}} - v_{\text{a\_est}}| \cdot (v_{\text{H}} - v_{\text{a\_est}}) \\ F_{\text{pist\_}L} &= N_{\text{L}} \cdot C_{\text{dL}} \cdot A_{\text{L}} \cdot \frac{\rho_{\text{a}}}{2} \cdot |v_{\text{L}} - v_{\text{a\_est}}| \cdot (v_{\text{L}} - v_{\text{a\_est}}) \end{aligned} \tag{7.10}$$

where C_{dH} is the heavy vehicle drag coefficient, A_{H} the heavy vehicle frontal area (m^2), v_{H} the heavy vehicle velocity (km/h), C_{dL} the light vehicle drag coefficient, A_{L} the light vehicle frontal area (m^2), v_{L} the light vehicle velocity (km/h), and ρ_{a} is the air density (kg/m^3).

The force caused by a static pressure difference on the tunnel portals is equal to

$$F_{\text{p}} = A_{\text{T}} \cdot (p_{\text{in}} - p_{\text{out}}) \tag{7.11}$$

where p_{in} and p_{out} are the inlet and outlet portal pressures.

The wall friction force, which is obtained from the following equation

$$F_{\text{wf}} = -k_{\text{wf}} \cdot A_{\text{T}} \cdot \frac{\rho_{\text{a}}}{2} \cdot \frac{L}{D} \cdot v_{\text{a\_est}}^2 \tag{7.12}$$

is always opposed to the direction of tunnel air flow, as well as the force caused by flow separation at the inlet portal

$$F_{\text{in}} = -k_{\text{in}} \cdot A_{\text{T}} \cdot \frac{\rho_{\text{a}}}{2} \cdot v_{\text{a\_est}}^2 \tag{7.13}$$

where k_{wf} is the wall friction coefficient, D the tunnel hydraulic diameter (m^2), and k_{in} is the inlet loss coefficient.

As we stated earlier, the goal of the predictive controller is determination of a number of jet-fans, which establishes the tunnel air velocity equal to $v_{\text{a\_est}}$. Having calculated all forces that induce movement of the air within the tunnel, predictive controller estimates the number of jet-fans as

$$n_{\text{F\_est}} = \frac{\sum F}{k_{\text{F}} \cdot A_{\text{F}} \cdot \rho_{\text{a}} \cdot |v_{\text{F}}| \cdot (v_{\text{F}} - v_{\text{a\_est}})} \tag{7.14}$$

where k_{F} is the pressure-rise coefficient of a jet-fan, A_{F} the discharging area of a jet-fan (m^2), v_{F} the discharging speed of a jet-fan (m/sec), and

$$\sum F = F_{\text{pist\_H}} + F_{\text{pist\_L}} + F_{\text{p}} + F_{\text{wf}} + F_{\text{in}}$$

Due to difficulties with determination of tunnel parameters k_{wf} and k_{in} and due to dynamic change of drag coefficients C_{dH} and C_{dL}, with respect to the number and the type of vehicles, one-dimensional force equation describes the tunnel air mass motion with a limited accuracy. An improvement can be achieved by using heuristically obtained look-up tables for drag coefficients determination.

For a precise estimation of tunnel parameters a set of on-site experiments and simulations should be conducted [56,57].

The other way to overcome the problem of inaccurate estimation is the usage of a *forgetting factor*, K_{JF}, in the form of a simple filter

$$\hat{n}_{F\_est}(k) = K_{JF} \cdot \hat{n}_{F\_est}(k-1) + (1 - K_{JF}) \cdot n_{F\_est}(k) \qquad (7.15)$$

In case that parameters and coefficients present in estimation equations are accurate, forgetting factor K_{JF} should be set close to 0. On the other hand, if exact values are not known, K_{JF} should be just about 1. In that way, influence of the predictive controller on the final number of active jet-fans can be controlled by only one parameter, K_{JF}.

7.3.1.4 Tunnel Parameters Identification

Accurate air flow prediction depends on parameters that are present in the tunnel model. The best way to get the knowledge about parameters values is to perform experiments within the tunnel. Results obtained during such experiment are shown in Figures 7.21 to 7.24. Each experiment lasted for two hours (120 min). During

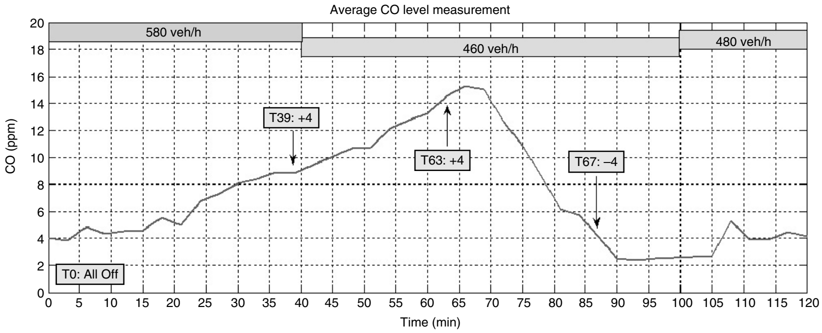


FIGURE 7.21 The average CO level measured during the day.

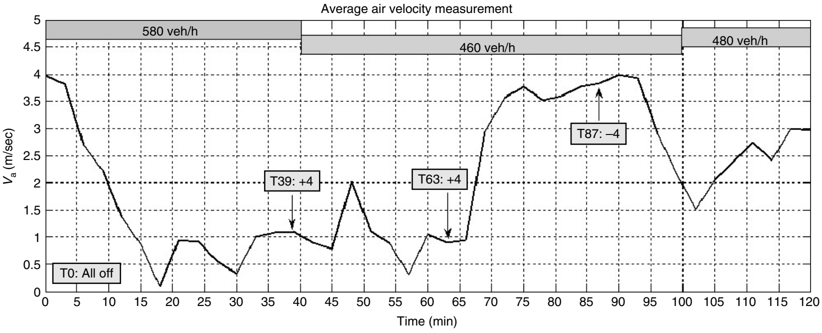


FIGURE 7.22 The average air velocity measured during the day.

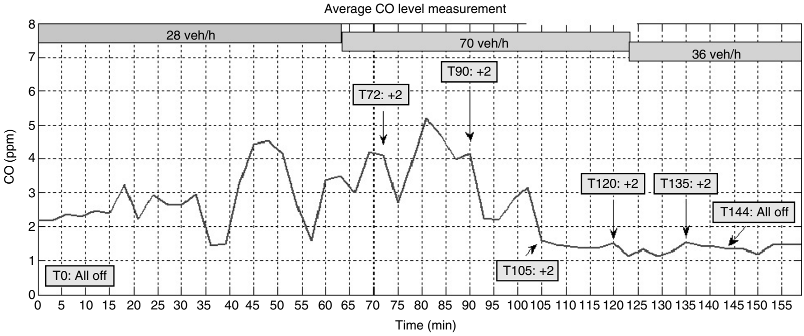


FIGURE 7.23 The average CO level measured during the night.

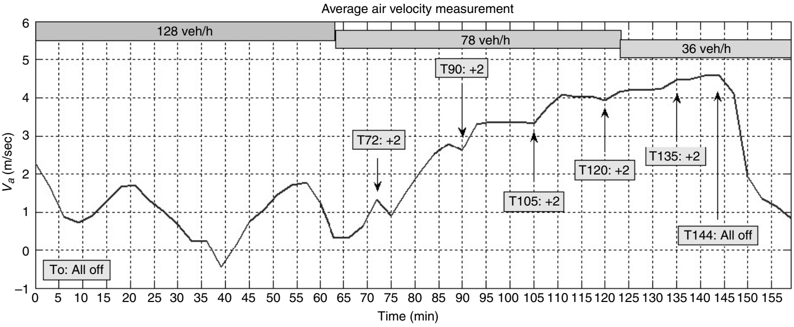


FIGURE 7.24 The average air velocity measured during the day.

that time fan groups were switched on and off, while traffic density was more or less constant. Figure 7.21 shows an average CO level measured during a daily experiment. The goal of the experiment was identification of the parameter q_{CO}, that is, the amount of CO produced by the vehicle per kilometer. At the beginning of the experiment ($t = 0$ min) all fan groups were switched off. As a consequence, the air velocity, shown in Figure 7.22, decreased from 4 to 0.75 m/sec in the next 15 min, which influenced the CO level in the tunnel; the average value started to increase. At $t = 39$ min, four fan groups were switched on, and at $t = 63$ min, additional four fan groups were switched on. The air velocity started to increase (Figure 7.22), thus reducing the level of CO from 15 to 2.5 ppm in the next 25 min. Having this experiment repeated several times, the average value of parameter q_{CO} can be calculated.

The other parameter important for design of the prediction algorithm and fuzzy controller is the pressure rise coefficient of a jet-fan, k_{F}. This parameter has high influence to the open-loop gain of the system. Results obtained by one of the conducted experiments are shown in Figure 7.23 and Figure 7.24. The experiments were undertaken during the night when the traffic density is very low, thus the influence of the traffic on the air velocity can be neglected. At the beginning of the experiment all fan groups were switched off. From Figure 7.24 it can be seen that the average value of the air velocity is approximately 1 m/sec. At $t = 72$ min, two

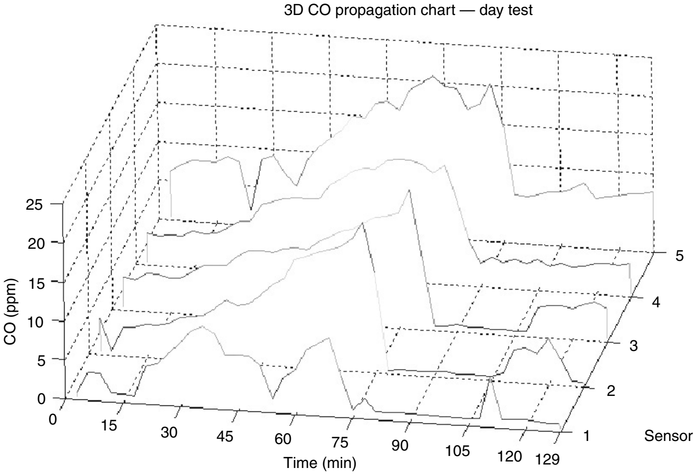


FIGURE 7.25 Average CO measurements from all five stations during the day.

fan groups were switched on. As a consequence the air velocity started to increase. For every 15 min in the next hour two additional fan groups were switched on. It can be seen that the influence of the fan group on air velocity depends on the number of currently active fan groups, that is, it depends on the actual air velocity, which is in accordance with Equation (7.14).

In order to get information about the tunnel dynamics, it is interesting to see how a pollutant propagates through the tunnel. The tunnel Učka has five CO measurement stations, whose data obtained during the daylight experiment (Figure 7.21 and Figure 7.22) are shown in Figure 7.25. At $t = 63$ min, eight fan groups were active and the average level of CO started to decrease on station 1, while it still increased on other stations (the air flow direction was from station 1 to 5).

Usually the set point of the pollutant controller is determined as the maximal value of the measured data in a particular moment (station 5 in our example measured 20 ppm). From Figure 7.25, we can conclude that actions based on that data would switch on fan groups without a real need for that since the pollutant level already started to decrease (at stations 1 and 2) and this trend would propagate through the tunnel in the next 20 min.

The new fuzzy-predictive controller takes into account current measurements of all stations and combines acquired data in order to form a set point such that it would save energy and reduce the switching of fan groups.

7.3.1.5 Fuzzy Controller

Prior to the description of the fuzzy controller, let us briefly exemplify details of the pollution control system currently used in the tunnel Učka. The tunnel, 5028 m long, is bidirectional with one lane in each direction (Istria ↔ Kvarner). There are 24 *fan groups* installed for each direction (three jet-fans form a fan group). The control algorithm of the old ventilation system is based only on pollutants measurements. Although distant stations for weather observation and loops for

TABLE 7.8
Actions Triggered on States

State	St_0	St_1	St_2	St_3	St_4	St_5
# Fan groups	all	2	2	1	1	1
Action	off	off	off	on	off	on
T (min)	0	15	15	10	30	5

From Bogdan, S. and Birgmajer, B., *IEEE Intl. Sympos. Industr. Electr. ISIE 2005*, 4, 151–156, 2005. With permission.

TABLE 7.9
Actions Triggered on Thresholds

Threshold	Tr_1	Tr_2	Tr_3	Tr_4	Tr_5
# Fan groups	all	2	2	2	1
Action	off	on	off	on	off

From Bogdan, S. and Birgmajer, B., *IEEE Intl. Sympos. Industr. Electr. ISIE 2005*, 4, 151–156, 2005. With permission.

measurement of traffic parameters exist, data collected by this equipment is not considered by the control algorithm (they are used only for monitoring and statistics). The switching of fan groups is led by pollutant thresholds and predefined states.

Since the number of thresholds and states, as well as control actions, is determined heuristically, the quality of ventilation depends on the operator's experience. At the beginning of a shift, the operator loads his/her procedure into the tables depending on the traffic parameters and weather conditions. According to Tables 7.8 and 7.9, the control algorithm works as follows: thresholds for CO are defined as $Tr_1 = 7$ ppm, $Tr_2 = 8$ ppm, $Tr_3 = 9$ ppm, $Tr_4 = 10$ ppm, and $Tr_5 = 12$ ppm. Then, CO level is in state St_0 if its measured value is less than Tr_1, that is, CO < 7 ppm. If $Tr_1 <$ CO $< Tr_2$, then carbon monoxide is in state St_1, and so on. From Table 7.8 we read that while CO level is in state St_1 the control algorithm switches off two fan groups every 15 min. If CO level is in state St_3 the control algorithm switches on one fan group every 10 min. Table 7.9 contains actions when states change. For example, transition from St_1 to St_2, that is, positive transition Tr_2, activates two fan groups, whereas changeover from St_2 to St_1 (negative transition) does not influence jet-fans. On the other hand, threshold Tr_5 activates the action only in the case of a negative transition (switches off one fan group).

Such a presentation of the control algorithm is very difficult to comprehend. The number of parameters (thresholds, states, actions, etc.) is large and it is very difficult to correlate their values with required control quality and system dynamics. Tabular approach cannot cope with a rapid change of traffic and weather conditions, as adaptation to new circumstances is done manually by the operator (assuming that the operator recognizes a new situation and has prepared a control table for it). Since the algorithm considers only direction of change (increase or decrease of CO level) without taking into account the amount of change, the time response of the ventilation system is rather slow and energy consuming. Furthermore, the assenting influence of natural ventilation is not incorporated into the algorithm, thus the number of active jet-fans, in case of a high level of CO and favorable pressure difference and traffic, is kept high for a much longer period than necessary. The first step in the improvement of previously used control algorithm is in its extension with a predictive controller, whereas the second step is substitution of thresholds and states tables with the fuzzy controller.

In order to take into account the dynamics of CO level transition, the fuzzy controller has two inputs, the pollutant deviation from a set point, e_{CO}, and the deviation rate of change, Δe_{CO}. One output, $\Delta n_{\mathrm{F\_r}}$, in the form of five singleton fuzzy sets is defined. Since the tunnel ventilation process has a static character, the final output is formed as

$$n_{\mathrm{F\_r}}(k) = n_{\mathrm{F\_r}}(k-1) + \Delta n_{\mathrm{F\_r}}(k) \tag{7.16}$$

Each input has five fuzzy sets (NL, NS, Z, PS, and PL) defined over its universe of discourse, as shown in Figure 7.26. The controller rules are shown in Table 7.10.

The fuzzy control algorithm, which uses *Mamdani* implication and calculates the crisp controller output according to the COG principle, is executed every 30 sec.

7.3.1.6 Simulation Experiments

The tunnel Učka ventilation system, controlled by the fuzzy-predictive controller, has been simulated and the results are compared with those obtained with the

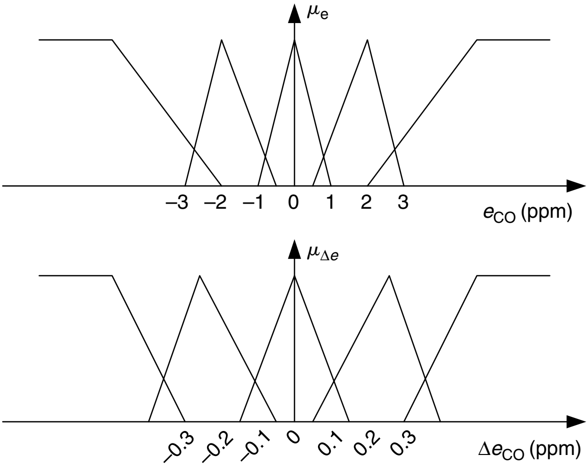


FIGURE 7.26 Membership functions of the fuzzy controller inputs. (From Bogdan, S. and Birgmajer, B., *IEEE Intl. Sympos. Industr. Electron. ISIE 2005*, 4, 151–156, 2005. With permission.)

TABLE 7.10
Fuzzy Controller Rules

	NLE	NSE	ZE	PSE	PLE
NLDE	3	3	1	1	0
NSDE	3	1	1	0	−1
ZDE	1	1	0	−1	−1
PSDE	1	0	−1	−1	−3
PLDE	0	−1	−1	−3	−3

Adapted from Bogdan, S. and Birgmajer, B., *IEEE Intl. Sympos. Industr. Electr. ISIE 2005*, 4, 151–156, 2005. With permission.

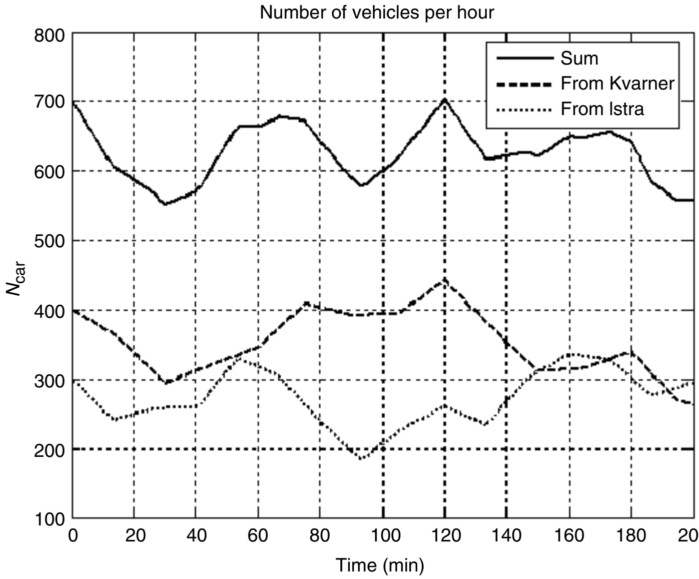


FIGURE 7.27 The traffic intensity in the tunnel during simulation. (From Bogdan, S. and Birgmajer, B., *IEEE Intl. Sympos. Industr. Electron. ISIE 2005*, 4, 151–156, 2005. With permission.)

tabular controller. Figure 7.27 shows the number of vehicles in the tunnel in both directions. The traffic intensity is changing randomly with an average value around 300 veh/h and a slightly higher concentration from the Kvarner portal. Figure 7.28 and Figure 7.29 show the CO concentration and the air velocity, respectively, in the case when control system is not active, that is, CO is diluted only by natural ventilation.

The results obtained with the longitudinal ventilation system controlled by the tabular controller are shown in Figures 7.30 to 7.32, while Figures 7.33 to 7.35 present the results in the case of fuzzy-predictive control of the tunnel ventilation.

It can be seen that in both cases the CO concentration is considerably reduced. The value of CO with inactive ventilation system soars over 40 ppm, whereas

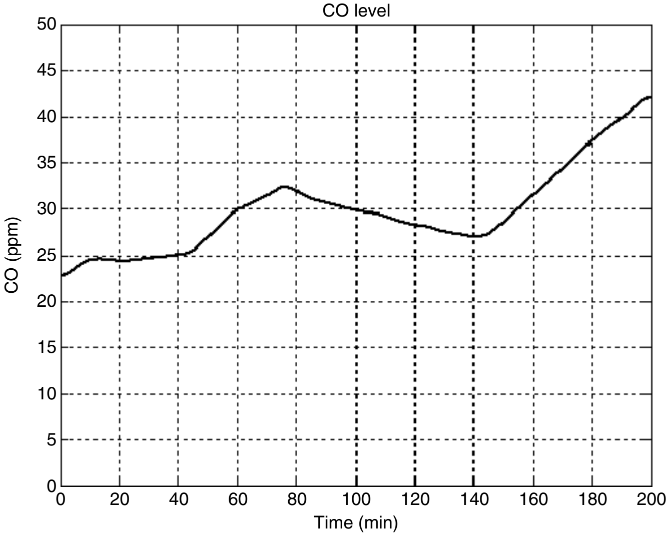


FIGURE 7.28 The CO level in the case of natural ventilation. (From Bogdan, S. and Birgmajer, B., *IEEE Intl. Sympos. Industr. Electron. ISIE 2005*, 4, 151–156, 2005. With permission.)

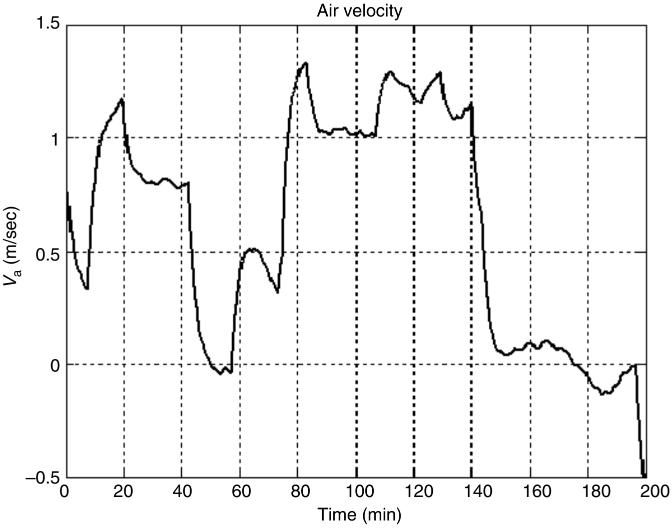


FIGURE 7.29 The air velocity in the case of natural ventilation. (From Bogdan, S. and Birgmajer, B., *IEEE Intl. Sympos. Industr. Electron. ISIE 2005*, 4, 151–156, 2005. With permission.)

tabular and fuzzy-predictive controllers keep that value around 9 to 10 ppm. The average CO concentration is 9.4 ppm for both controllers, but differences in CO level dynamics are noticeable (Figure 7.30 and Figure 7.33). The fuzzy-predictive controller reacts instantly to any changes in traffic or weather conditions,

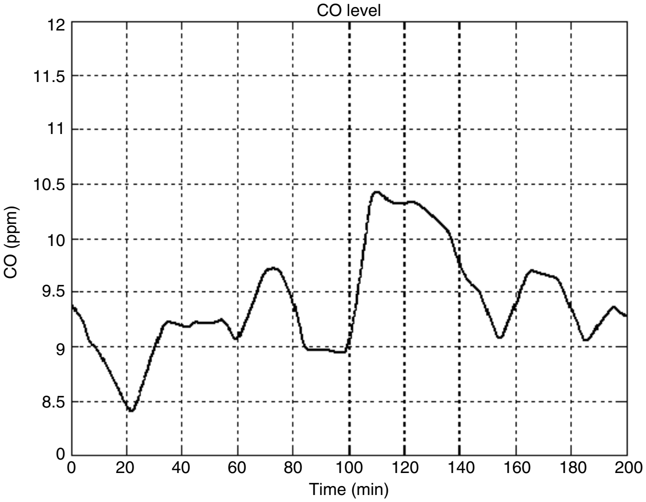


FIGURE 7.30 The CO level in the case of using the tabular controller. (From Bogdan, S. and Birgmajer, B., *IEEE Intl. Sympos. Industr. Electron. ISIE 2005*, 4, 151–156, 2005. With permission.)

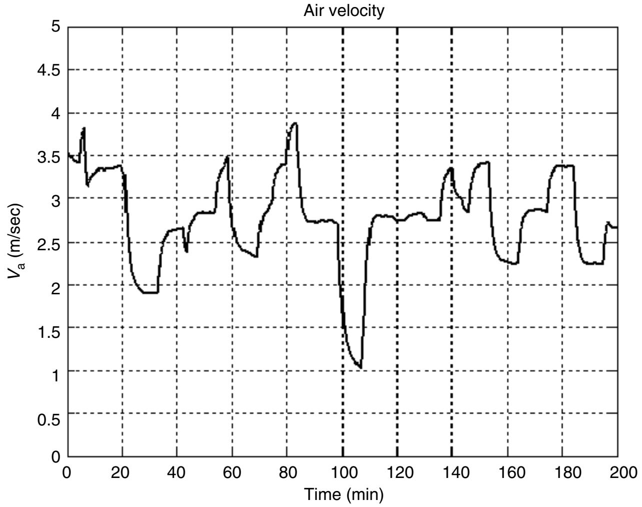


FIGURE 7.31 The air velocity in the case of using the tabular controller. (From Bogdan, S. and Birgmajer, B., *IEEE Intl. Sympos. Industr. Electron. ISIE 2005*, 4, 151–156, 2005. With permission.)

before these changes significantly affect the CO concentration. The installed power of each fan group is 80 kW. During 200 min, which was the period of simulation, jet-fans, in the case of the tabular controller, have consumed 645 kWh. In the same time, energy used by the fan groups controlled with the fuzzy-predictive controller, was 615 kWh, which is around 5% less than in the first case. When

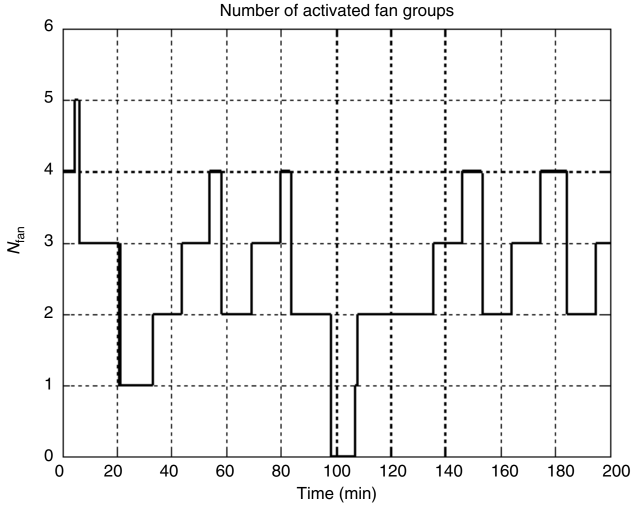


FIGURE 7.32 The number of active fan groups in the case of using the tabular controller. (From Bogdan, S. and Birgmajer, B., *IEEE Intl. Sympos. Industr. Electron. ISIE 2005*, 4, 151–156, 2005. With permission.)

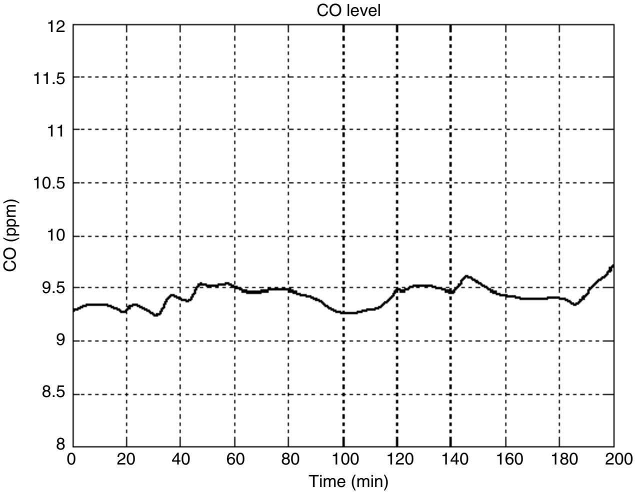


FIGURE 7.33 The CO level in the case of using the fuzzy-predictive controller. (From Bogdan, S. and Birgmajer, B., *IEEE Intl. Sympos. Industr. Electron. ISIE 2005*, 4, 151–156, 2005. With permission.)

comparing dynamics of activated jet-fans (Figure 7.32 and Figure 7.35), one can see that jet-fans are switched on and off twice more often in the case of the tabular controller, significantly increasing the stress on the supply power grid, because the fan current drain peaks when it is being switched on.

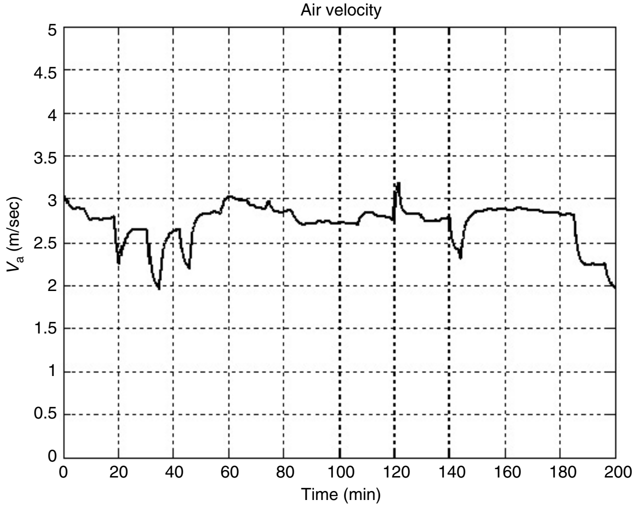


FIGURE 7.34 The air velocity in the case of using the fuzzy-predictive controller. (From Bogdan, S. and Birgmajer, B., *IEEE Intl. Sympos. Industr. Electron. ISIE 2005*, 4, 151–156, 2005. With permission.)

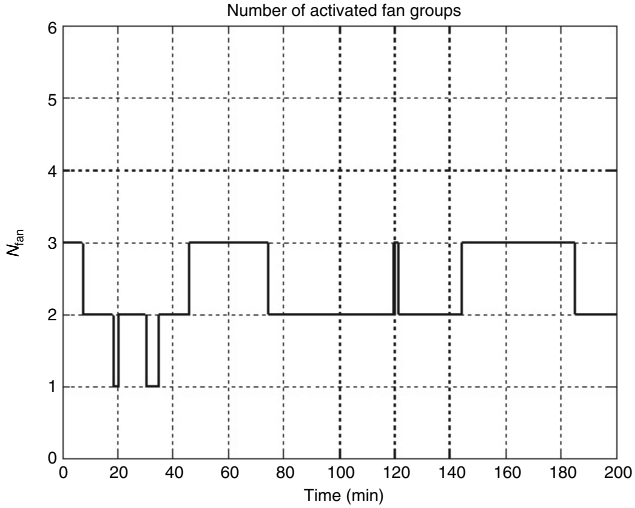


FIGURE 7.35 The number of active fan groups in the case of using the fuzzy-predictive controller. (From Bogdan, S. and Birgmajer, B., *IEEE Intl. Sympos. Industr. Electron. ISIE 2005*, 4, 151–156, 2005. With permission.)

7.3.1.7 FBD-Based Implementation of a Fuzzy-Predictive Controller

The fuzzy-predictive controller is implemented on an industrial PC with a Windows NT-based operating system. The software structure of the control system is shown

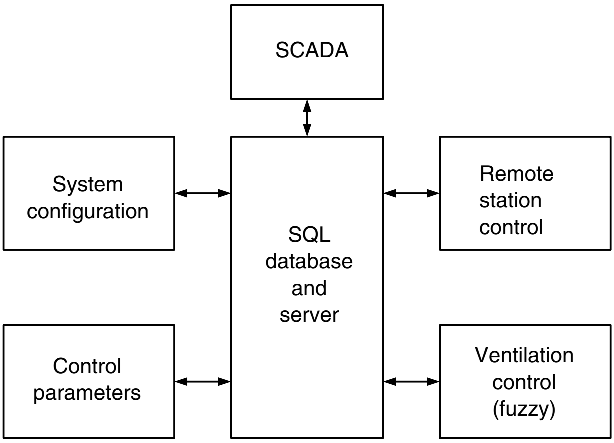


FIGURE 7.36 The road tunnel ventilation control system software.

in Figure 7.36. The system is built around a SQL database and server, SCADA for visualization and control of the system is written in Java, remote station control software is implemented in C++, while the fuzzy-predictive ventilation control is written in the Function Block Diagram (FBD) language. A PC-based executable code is generated by an FBD programming tool.

Figure 7.37 shows the implementation of the fuzzification part of the fuzzy controller, written in the FBD language. In order to simplify the control over the distribution of input membership functions, the fuzzy controller has two tunable parameters, K_{neg} and K_{pos}, for each input (see Figure 7.38). By changing only one or both parameters, the user can easily adjust the widths of membership functions at the negative and positive sides of the input universes of discourse, respectively.

7.3.2 Fuzzy Control of Anesthesia

The activities of anesthetists include numerous repeated and isolated tasks. Anesthetists have to observe and control a great number of different variables of anesthesia, and vital functions. Anesthesia means an adequate hypnosis, analgesia, and muscle relaxation for suppression of the effects of surgical manipulations. In each of these areas the main problem lies in the measurements. The degree of relaxation can be estimated by different methods such as electromyography (EMG), mechanomyography (MMG), and acceleromyography (AMG).

Quite often the meaning of anesthesia is erroneously restricted to hypnosis. Therefore, “the depth of anesthesia” should be replaced correctly by “the depth of hypnosis” (DOH). Anesthesia consists of the above named three parts. Up to now, there are only indirect parameters to quantify analgesia.

The introduction of new short-acting compounds of hypnotic drugs needs a continuous mode both for measurement of control variables and for injection

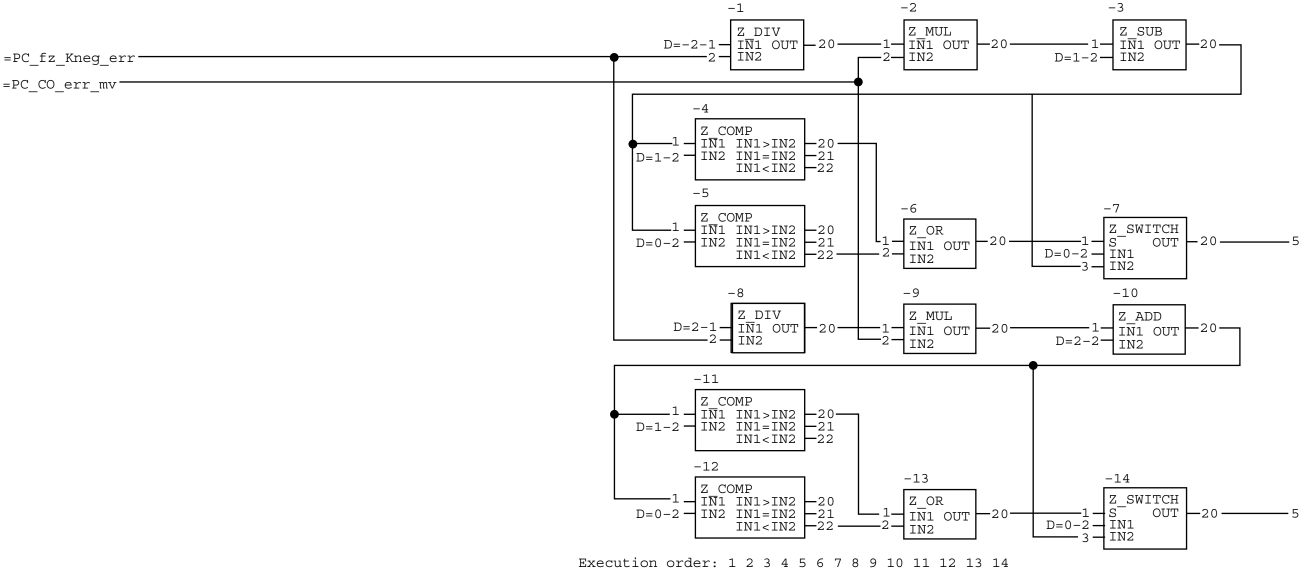


FIGURE 7.37 The fuzzification part of the fuzzy controller, written in FBD language.

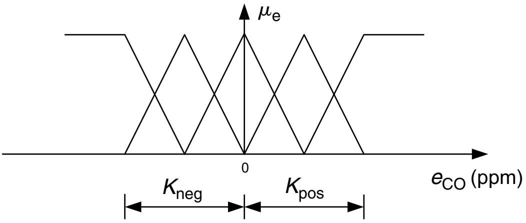


FIGURE 7.38 Adjustable coefficients, K_{neg} and K_{pos}, for changing the distribution of input.

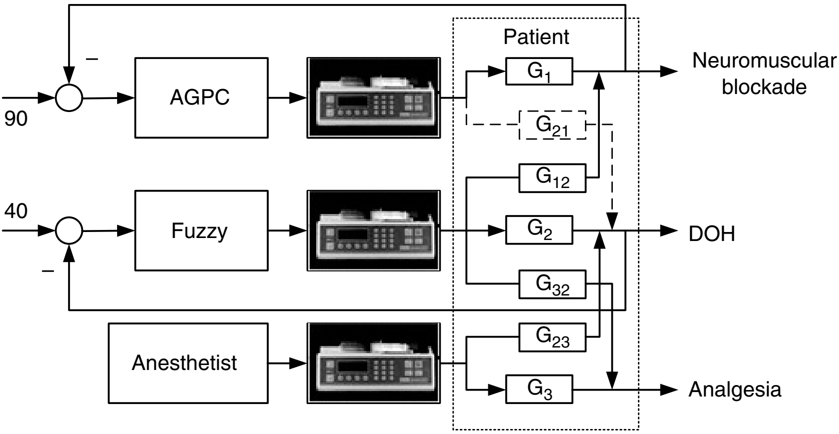


FIGURE 7.39 Overall control structure with AGPC NMB control end set point (e.g., 90% NMB), open-loop control of analgesia and closed-loop fuzzy DOH control.

of drugs into the patient. Figure 7.39 shows the overall control structure for the noted three parts of anesthesia [64]. One can see that the whole control system consists of the control loop for neuromuscular blockade (NMB) controlled via adaptive generalized predictive controller (AGPC), open-loop control of analgesia, and closed-loop DOH control via fuzzy controller. Automatic control of anesthesia supports the anesthetist in his activities. The closed-loop NMB and DOH control prevents awareness and drug overdoses.

The anesthetist uses different parameters to estimate the depth of hypnosis. Some of them such as tearing and sweating are not measurable. However, automatic DOH control needs measurable outputs. A direct measurement of the hypnosis is not yet available. DOH is expected to be reflected in the electroencephalogram (EEG). Different algorithms are known for the estimation of residuals as indicator for the depth of hypnosis from the raw EEG. The main disadvantage of the EEG measurement is its variation with different anesthetic agents. One group of measurement methods is based on the power spectrum. The complexity of the raw EEG decreases with an increasing DOH. The spectral edge frequency 95%

(SEF 95) determines a frequency limit, such that 95% of the signal power corresponds to frequencies up to this limit. The median frequency corresponds to 50% of the power spectrum. The edge frequencies decrease with the increase in DOH. The correlation between the spectral edge frequencies and the DOH is not clearly defined. The use of the SEF as a valid measurement for the depth of hypnosis is contentious [58].

The bispectral (BIS) index has become very popular during the last years and has been validated in a large number of studies. The BIS index combines the power spectrum and bispectrum with a burst suppression analysis. The BIS describes a complex EEG pattern by a simple variable. A good correlation of the BIS with the plasma concentration of Sevoflurane was described in Reference 58. Different versions of so-called BIS-monitors vary a lot in measurement and cannot be compared. Another measurement procedure is the measurement of the EEG response to stimulation. The evoked potentials reflect the subjective clinical signs that anesthetists use. They are indicators of the response of the central nervous system (CNS). Auditory evoked potentials (AEP) are used in different applications [59]. In the presented control solution, the BIS-monitor Aspect-XP from Aspect Medical Systems was used for the recording of EEG and DOH estimation. The daily clinical use of such equipment is simpler than the use of the AEP.

The controlled plant consists of three parts: a drug application unit, the patient, and a measurement unit. In the presented control solution, the liquid drug *propofol* and the BIS-monitor as a measurement device were used. For the application of liquid drugs, there exist two systems, infusion pumps and infusion pumps combined with a drug distribution model. The latter are called target controlled infusion systems (TCI), which can appear in open- and closed-loop configurations [59]. The TCI system calculates the necessary infusion value with regard to a target blood concentration. The described controller estimates an infusion value in mg/kg/h. This value is mapped onto the infusion rate. The syringe pumps (Graseby 3400) as actuators make errors for low infusion rates, so a correction function for the infusion value was used.

A universal DOH controller has to manage different types of patients. Patients react in a different way to the injected infusion of *propofol* and therefore it is difficult to make a universal model of the DOH. Another reason for difficulties with modeling lies in the nonlinear operating method of the BIS-monitor. A mixture of statistic and lookup table calculations ascertains the output value — BIS-index of the BIS-monitor. Additional sources of ambiguity are still undefined or unquantified interactions between hypnotic drugs, or the DOH and analgetic drugs, or the depth of analgesia. Therefore, the design of model-based, robust and predictive controllers is not a promising way to go. On the other hand, fuzzy controllers, known for their stable and robust performance, can be developed without numerical models, thereby opening the possibility of implementing expert knowledge from anesthetists.

Figure 7.40 shows the structure of a developed Fuzzy PD + I controller. A prefilter determines the dynamically filtered and normalized control error and change of the control error. The fuzzy system has two outputs. One output corresponds

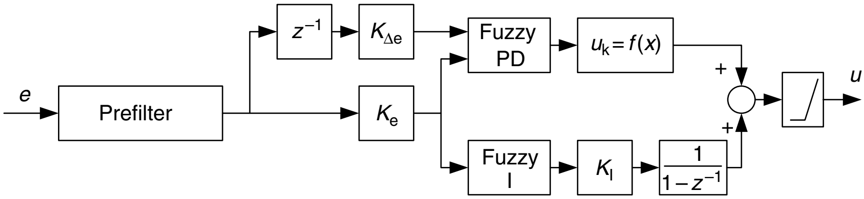


FIGURE 7.40 Structure of the fuzzy PD + I controller with a prefilter and a compensation [64].

TABLE 7.11
Fuzzy Rule-Table of the PD-Type Controller

	NLE	NME	NSE	ZE	PSE	PME	PLE
NLDE	VS	VS	VS	VS	S	M	SB
NMDE	VS	VS	VS	S	M	SB	B
NSDE	VS	VS	S	M	M	B	VB
ZDE	VS	VS	S	M	SB	B	VB
PSDE	VS	VS	M	SB	SB	VB	VB
PMDE	VS	VS	M	SB	B	VB	VB
PLDE	VS	S	SB	B	B	VB	VB

to a PD-type fuzzy controller output, while the other is summed up to generate an integral controller output. These two parts are superposed to the control signal [60]. To avoid stability problems with highly sensitive patients, the control signal is limited by a lower bound of 2 mg/kg/h. Table 7.11 shows the fuzzy rule-table and Figure 7.41 shows the control surface of the PD-type fuzzy DOH controller.

The designed fuzzy DOH controller is tested by simulation before it is used in a real situation in the operating theatre. The known pharmacokinetic behavior is described in a linear three-compartment model [61], as shown in Figure 7.42. Using the notations x_i for the drug concentration in the ith compartment at time t, $\dot{x}_i$ for the rate of change and u for the given drug input, the pharmacokinetic model is described by

$$\dot{x}_1 = k_{21}x_2 + k_{13}x_3 - (k_{10} + k_{12} + k_{13})x_1 + u \tag{7.17}$$

$$\dot{x}_2 = k_{12}x_1 - k_{21}x_2 \tag{7.18}$$

$$\dot{x}_3 = k_{13}x_1 - k_{31}x_3 \tag{7.19}$$

Experimental work has shown that a so-called effect compartment should be added to the pharmacokinetic equations [62]

$$\dot{x}_E = k_{1E}x_1 - k_{E0}x_E \tag{7.20}$$

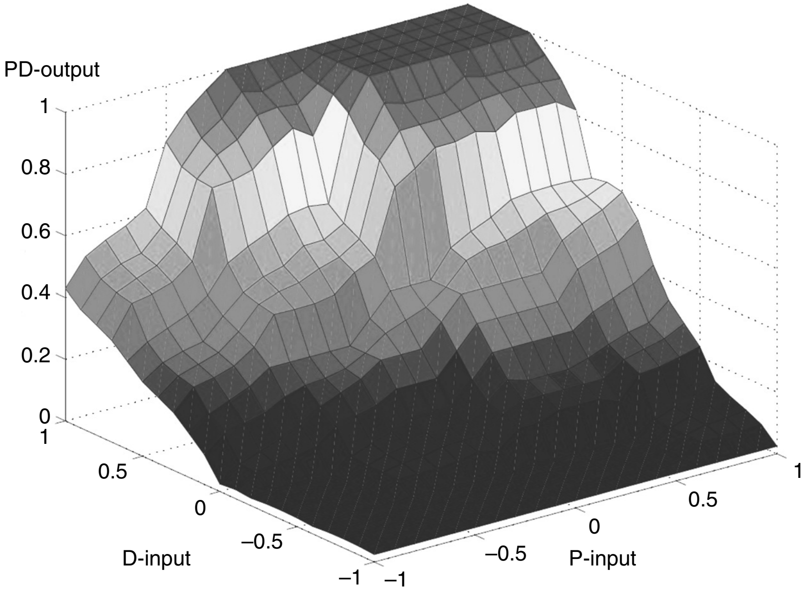


FIGURE 7.41 Fuzzy control surface of the PD-type controller.

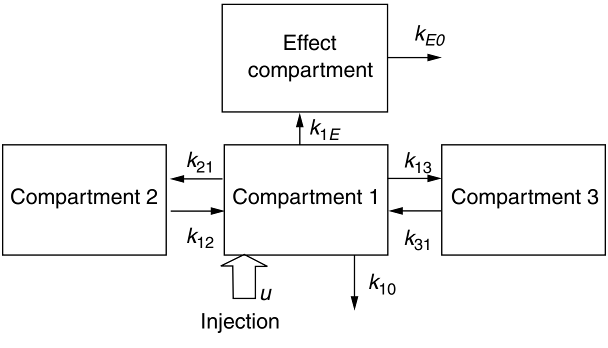


FIGURE 7.42 Compartment model for simulation.

where k_{10}, k_{E0} denote the elimination rate constants and $k_{12},\ k_{21},\ k_{13}$, and k_{31}, the transfer constants between compartments 1 and 2 or compartments 1 and 3, respectively.

By combining Equation (7.20) with Equations (7.17) to (7.19) and applying the Laplace transform we get the transfer function

$$\frac{X_E(s)}{U(s)} = \frac{K(1 + T_{v1}s)(1 + T_{v2}s)\mathrm{e}^{-\tau s}}{(1 + T_1 s)(1 + T_2 s)(1 + T_3 s)(1 + T_4 s)} \tag{7.21}$$

TABLE 7.12
Statistics of DOH Control

Parameter	Mean value ± SD	Min	Max
tcontrol [min]	81.1 ± 41.6	12	170
Mean DOH [BIS-Index]	39.8 ± 1.1	38.2	44.2
RMSD [BIS-Index]	4.9 ± 2.2	2	11.3

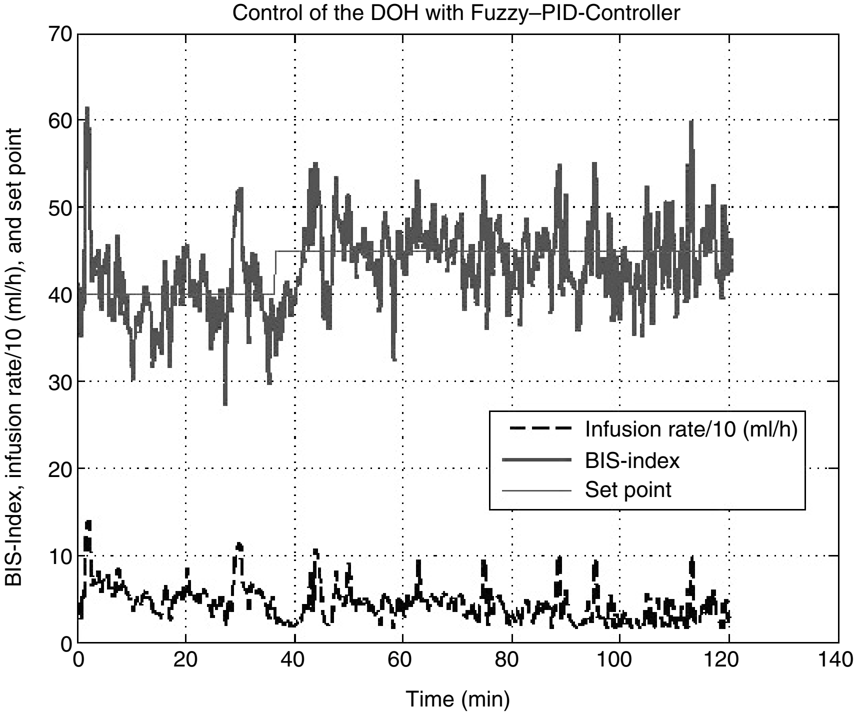


FIGURE 7.43 DOH fuzzy control in practice.

where an additional dead time τ has been introduced to model the transport delay of the drug.

To complete a *propofol* nonlinear model, we added a Hill-equation to describe the relationship between the concentration in the blood and the measured BIS value [61]:

$$E_{\text{eff}} = \frac{E_{\max} X_{\text{e}}^{\alpha}}{X_E^{\alpha} + X_E^{\alpha}(50)} \tag{7.22}$$

where $X_E(50)$ is the drug concentration at a 50% effect of the final value $E_{\max}$, X_E the drug concentration and α the Hill coefficient.

After collection of some identification data sets, parameter optimization methods were used to build a simulation model with the described structure. The following simulation parameters were calculated:

$$\begin{array}{rlrlrl} K &= 0.0036 & T_{v1} &= 6.41\ \text{min} & T_{v2} &= 6.55\ \text{min} \\ T_1 &= 0.46\ \text{min} & T_2 &= 13.31\ \text{min} & T_3 &= 11.83\ \text{min} \\ T_4 &= 3.36\ \text{min} & X_E(50) &= 0.17\ \mu\text{g/mL} & \alpha &= 3.98 \\ \tau &= \frac{1}{3}\ \text{min} & & & & \end{array}$$

For the DOH control, 25 patients were successfully controlled. The set point for the DOH was BIS-index 40. As can be seen from Table 7.12, a mean BIS-index of 39.8 ± 1.1 was reached. In this study, 25 patients with general and accident operation were involved. In average, the controller worked 81 ± 41.6 min. Due to the nonadaptive robust control strategy, the root mean square deviation (RMSD) (with 4.9 ± 2.2) was high.

The transient response of fuzzy DOH control appears in Figure 7.43. In the experiment, the set point was adjusted to BIS-index 40, where a BIS-index between 30 and 50 was tolerated. According to the intraoperative patient behavior, the anesthetist changed the set point in this example to BIS-index 45. This became necessary because the blood pressure and the hearth frequency decreased as the consequence of the *propofol* infusion.

References

1. Mamdani, E.H., "Applications of fuzzy algorithms for control of simple dynamic plant," *Proceedings IEE*, 121, 1585–1588, 1974.
2. Assilian, S. and Mamdani, E.H., "An experiment in linguistic synthesis with a fuzzy logic controller," *International Journal of Man–Machine Studies*, 7, 1–13, 1974.
3. Sugeno, M., "Application of fuzzy set and logic to control," *Measurement and Control*, 18, 150–160, 1975.
4. Kickert, W.J.M. and Van Nauta Lemke, H.R., "Application of a fuzzy controller in a warm water plant," *Automatica*, 12, 301–308, 1976.
5. Tong, R.M., "A control engineering review of fuzzy systems," *Automatica*, 13, 559–569, 1977.
6. Gupta, M.M. and Nikiforuk, P.N., "Feedback control of industrial processes via fuzzy set theory: some remarks," *Proceedings of the Joint Automatic Control Conference*, Pittsburgh, pp. 37–52, 1978.
7. Larsen, P.M., "Industrial applications of fuzzy logic control," *International Journal of Man–Machine Studies*, 12, 3–10, 1980.
8. Pappis, C.P. et al., "A fuzzy logic controller for a traffic junction," *IEEE Transactions on Systems, Man and Cybernetics*, 7, 707–712, 1977.

9. Yasunobu, S. et al., "Fuzzy control for automatic train operation system," *Proceedings of the 4th IFAC/IFIP/IFRS International Conference on Transportation Systems*, pp. 39–45, 1983.
10. Etschmaier, M.M., "Fuzzy controls for maintenance scheduling in transportation systems," *Automatica*, 16, 255–264, 1980.
11. Rovind, G., "Synthesis of fuzzy controllers for process plants," *Proceedings of the IEEE International Conference on Cybernetics and Society*, Tokyo, November 1978.
12. Tong, R.M. et al., "Fuzzy control of the activated sludge wastewater treatment process," *Automatica*, 16, 659–701, 1980.
13. Holmblod, L.P. and Ostergaard, J.J., "Control of a cement kiln by fuzzy logic," in M.M. Gupta and E. Sanchez (Eds), *Fuzzy Information and Decision Process*, North-Holland Company, Amsterdam, pp. 389–399, 1982.
14. King, R.E., "Fuzzy logic control of a cement kiln precalciner flash furnace," *Proceedings of the IEEE Conference on Applications of Adaptive and Multivariable Control*, Hull (UK), 1982.
15. Sugeno, M. and Takagi, T., "A new approach to design of fuzzy controller," in P.P. Wang and S.K. Cheng (Eds), *Advances in Fuzzy Sets, Possibility Theory, and Applications*, Plenum Press, New York, pp. 325–334, 1983.
16. Uragani, M., Mizumoto, M., and Tanaka, K., "Fuzzy robot controls," *Journal of Cybernetics*, 6, 39–64, 1976.
17. Scharf, E.M. and Mandic, N.J., "The application of a fuzzy controller to the control of a multi-degree-of-freedom robot Arm," in M. Sugeno (Ed.), *Industrial Applications of Fuzzy Control*, Elsevier Science Publishers B.V., North-Holland, Amsterdam, pp. 41–61, 1985.
18. Larkin, L.I., "A fuzzy logic controller for aircraft flight control," in M. Sugeno (Ed.), *Industrial Applications of Fuzzy Control*, Elsevier Science Publishers B.V., North-Holland, Amsterdam, pp. 87–103, 1985.
19. Yamaguchi, T., Goto, K., Takagi, T., Doya, K., and Mita, T., "Intelligent control of a flying vehicle using fuzzy associative memory system," *Proceedings of the IEEE International Conference on Fuzzy Systems*, pp. 1139–1149, 1992.
20. Sugeno, M., Murofushi, T., Nishino, J., and Miwa, H., "Helicopter flight control based on fuzzy logic," *Fuzzy Engineering toward Human Friendly Systems: Proceedings of the International Fuzzy Engineering Symposium*, Vol. 2, pp. 1120–1121, 1991.
21. Sugeno, M. (Ed.), *Industrial Applications of Fuzzy Control*, Elsevier Science Publishers B.V., North-Holland, Amsterdam, pp. 231–239, 1985.
22. Yen, J., Langari, R., and Zadeh, L.A. (Eds), *Industrial Applications of Fuzzy Logic and Intelligent Systems*, IEEE Press, Washington, 1995.
23. Bogdan, S. and Birgmajer, B., "Fuzzy-predictive control of a road tunnel ventilation system," *CD-ROM Proceedings of the IEEE International Symposium on Industrial Electronics*, Dubrovnik, Croatia, 2005.
24. Linkens, D.A., Abbod, M.F., Backory, J.K., and Shieh, J.S., "Closed-loop control of anaethesia using fuzzy logic," in P.S. Szczepaniak, P.J.G. Lisboa, and J. Kacprzyk (Eds), *Fuzzy Systems in Medicine*, Studies in Fuzziness and Soft Computing, Physica-Verlag, Heidelberg, New York, pp. 467–483, 2000.
25. Yamakawa, T. and Miki, T., "The current mode fuzzy logic integrated circuits fabricated by the standard CMOS process," *IEEE Transactions on Computers*, C-35, 161–167, 1986.

26. *Fuzzy Microcontrollers User's Guide*, NeuraLogix, Inc., Sanford, FL, 1991.
27. Dualibe, C., Jespers, P., and Verleysen, M., "On designing mixed-signal programmable fuzzy logic controllers as embedded subsystems in standard CMOS technologies," *Proceedings of the 14th Symposium on Integrated Circuits and System Design*, Pirenopolis (Brazil), pp. 194–200, 2001.
28. Yamakawa, T., "A fuzzy inference engine in nonlinear analog mode and its application to a fuzzy logic control," *IEEE Transactions on Neural Networks*, 4, 496–522, 1993.
29. Miki, T., Matsumoto, H., Ohto, K., and Yamakawa, T., "Silicon implementation for a novel high-speed fuzzy inference engine: mega-flips analog fuzzy processor," *Journal of Intelligent and Fuzzy Systems*, 1, 27–42, 1993.
30. Rodríguez-Vázquez, A., Vidal, F., Linares, B., and Delgado, M., "Analog CMOS design of singleton fuzzy controllers," *3rd International Conference on Industrial Fuzzy Control Intelligent Systems*, Houston, Texas, December 1993.
31. Lemaitre, L., Patyra, M.J., and Mlynek, D. "Analysis and design of CMOS fuzzy logic controller in current mode," *IEEE Journal of Solid-State Circuits*, 29, 317–322, 1994.
32. Manaresi, N., Rovatti, R., Franchi, E., Guerrieri, R., and Baccarani, G., "A silicon compiler of analog fuzzy controllers: from behavioral specifications to layout," *IEEE Transactions on Fuzzy Systems*, 4, 418–428, 1996.
33. Guo, S., Peters, L., and Surmann, H., "Design and application of an analog fuzzy logic controller," *IEEE Transaction on Fuzzy Systems*, 4, 429–438, 1996.
34. Huertas, J., Sanchez-Solano, S., Baturone, I., and Barriga, A., "Integrated circuit implementation of fuzzy controllers," *IEEE Journal of Solid State Circuits*, 31, 1051–1058, 1996.
35. Liu, D., Huang, Y., and Wu, Y., "Modular current-mode defuzzification circuit for fuzzy logic controllers," *Electronics Letters*, 30, 1287–1288, 1994.
36. Marshall, G. and Collins, S., "Fuzzy logic architecture using subthreshold analogue floating-gate devices," *IEEE Transactions on Fuzzy Systems*, 5, 32–43, 1997.
37. Franchi, E., Manaresi, N., Rovatti, R., Bellini, A., and Baccarani, G., "Analog synthesis of nonlinear functions based on fuzzy logic," *IEEE Journal of Solid State Circuits*, 33, 885–895, 1998.
38. Landolt, O., "Low power analog fuzzy rule implementation based on a linear transistor network," *Proceedings of the 5th International Conference on Microelectronics for Neural Networks and Fuzzy Systems, MicroNeuro'96*, pp. 86–93, 1996.
39. Rodriguez-Vazquez, A., Navas, R., Delgado-Restituto, M., and Vidal-Verdu, F., "A modular programmable CMOS analog fuzzy controller chip," *IEEE Transactions on Circuits and Systems — II: Analog and Digital Signal Processing*, 46, 251–265, 1999.
40. Omron Corporation, FP-3000 Digital Fuzzy Processor User's Manual, 1991.
41. Pagni, A., Poluzzi, R., and Rizotto, G., "Automatic synthesis, analysis and implementation of fuzzy controller," *Proceedings of the IEEE International Conference on Fuzzy Systems*, pp. 105–110, 1992.
42. Högener, J., "Fuzzy PLC: a connection with a future," *Proceedings of the 1st European Congress on Fuzzy and Intelligent Technologies, EUFIT'93*, pp. 688–691, 1993.

43. Yamakawa, T., “A fuzzy programmable logic array (fuzzy PLA),” *Proceedings of the IEEE International Conference on Fuzzy Systems*, San Diego, CA, USA, pp. 459–465, 1992.
44. McKenna, M. and Wilamovski, B.M., “Implementing a fuzzy system on a field programmable gate array,” *Proceedings of the International Joint Conference on Neural Networks IEEE IJCNN*, pp. 189–194, 2001.
45. Yamazaki, T. and Sugeno, M., “A microprocessor based fuzzy controller for industrial purposes,” in M. Sugeno (Ed.), *Industrial Applications of Fuzzy Control*, Elsevier Science Publishers B.V., North-Holland, Amsterdam, pp. 231–239, 1985.
46. Bogdan, S., Kovačić, Z., Lončar, D., and Lukačević, D., “Fuzzy gain scheduling control of the condensate level,” *CD-ROM Proceedings of the 9th Mediterranean Conference on Control and Automation Control, FM1-A*, Dubrovnik, 2001.
47. Ordys, A.W., Pike, A.W., Johnson, M.A., Katebi, R.M., and Grimble, M.J., *Modelling and Simulation of Power Generation Plants*, Springer-Verlag, London, 1994.
48. Lu, C.X., Bell, R.D., and Rees, N.W., “Scheduling control of deaerator plant,” *Proceedings of the IFAC Control of Power Systems and Power Plants*, Beijing, pp. 183–189, 1997.
49. Bell, R.D. and Astrom, K.J., “A fourth order non-linear model for drum-boiler dynamics,” *Proceedings of the IFAC 13th World Congres*s, San Francisco, pp. 31–36, 1996.
50. Lewis, R.W., *Programming industrial control systems using IEC 61131–3*, IEE, UK, 1998.
51. Modic, J., “Air velocity and concentration of noxious substances in a naturally ventilated tunnel,” *Tunnelling and Underground Space Technology*, 18, 405–410, 2003.
52. Bettelini, M., Brandt, R., and Riess, I., “Progress in tunnel ventilation — the Mont-Blanc tunnel,” *Proceedings of the AITES-ITA 2001 World Tunnel Congress*, Milano, 2001.
53. Karakas, E., “The control of highway tunnel ventilation using fuzzy logic,” *Engineering Application of Artificial Intelligence*, 16, 717–721, 2003.
54. Hoogendoorn, S., Hoogendoorn-Lanser, S., and Schuurman, H., “Fuzzy perspectives in traffic engineering,” *Research Report, TRAIL Research School, Delft, Report on behalf of Dutch Ministry of Transport*, 1998.
55. Mizuno, A. et al., “Evaluation of the performance of control of the road tunnel ventilation,” *Proceedings of the 8th ISAVVT*, Liverpool, pp. 903–917, 1994.
56. Jang, H.M. and Chen, F., “On the determination of the aerodynamic coefficients of highway tunnels,” *Journal of Wind Engineering and Industrial Aerodynamics*, 90, 869–896, 2002.
57. Bring, A., Malmstrom, T., and Boman, C.A., “Simulation and measurement of road tunnel ventilation,” *Tunneling and Underground Space Technology*, 12, 417–424, 1997.
58. Widman, G., Schreiber, T., Rehberg, B., Hoeft, A., and Elger, C.E., “Quantification of depth of anaesthesia by nonlinear time series analysis of brain electrical activity,” *Physics Review E*, 62, 4898–4903, 2000.
59. Kenny, G.N. and Mantzaridis, H., “Closed-loop control of propofol anesthesia,” *British Journal of Anesthesia*, 83, 223–228, 1999.

60. Simanski, O., Kähler, R., Pohl, B., Hofmockel, R., and Lampe, B., "Control in general anaesthesia — a contribution," in K.P. Adlassnig (Ed.), *Proceedings Eunite Workshop on Intelligent Systems in Patient Care*, Vienna, pp. 169–175, 2001.
61. Simanski, O., "Development of a System for Measurement and Control of the Neuromuscular Blockade and the Depth of Hypnosis," PhD thesis, Institute of Automation, University of Rostock, Germany, 2002.
62. Sheiner, L.B., Stanski, D.R., Vozeh, S., Miller, R.D., and Ham, J., "Simultaneous modelling of pharmacokinetics and pharmacodynamics: application to d-tubocurarine," *Clinical Pharmacology and Therapeutics*, 25, 358–362, 1979.
63. Bogdan, S. and Birgmajer, B., "Fuzzy predictive control at a road tunnel ventilation systems", *Proceedings of the IEEE International Symposium on Industrial Electronics ISIE 2005,* 4, 151–156, Dubrovnik, 2005
64. Simanski, O., Kähler, R., Pohl, B., Hoinfmockel, R., Friedrich, R., and Lampe, B.P., "Measurement and control of neuromuscular block and depth of hyphosis" *European Journal of Control* (submitted for publication).