4-2002

Systematic Methods for the Design of a Class of Fuzzy Logic Controllers

Saad Y. Yasin
Western Michigan University

Follow this and additional works at: http://scholarworks.wmich.edu/dissertations

Part of the Navigation, Guidance, Control and Dynamics Commons

Recommended Citation
http://scholarworks.wmich.edu/dissertations/1342

This Dissertation-Open Access is brought to you for free and open access by the Graduate College at ScholarWorks at WMU. It has been accepted for inclusion in Dissertations by an authorized administrator of ScholarWorks at WMU. For more information, please contact maira.bundza@wmich.edu.
SYSTEMATIC METHODS FOR THE DESIGN OF A CLASS OF FUZZY LOGIC CONTROLLERS

By

Saad Y. Yasin

A Dissertation
Submitted to the
Faculty of the Graduate College
in partial fulfillment of the
requirements for the
Degree of Doctor of Philosophy
Department of Mechanical and Aeronautical Engineering

Western Michigan University
Kalamazoo, Michigan
April 2002
Fuzzy logic control, a relatively new branch of control, can be used effectively whenever conventional control techniques become inapplicable or impractical. Various attempts have been made to create a generalized fuzzy control system and to formulate an analytically based fuzzy control law. In this study, two methods, the left and right parameterization method and the normalized spline-base membership function method, were utilized for formulating analytical fuzzy control laws in important practical control applications. The first model was used to design an idle speed controller, while the second was used to control an inverted control problem. The results of both showed that a fuzzy logic control system based on the developed models could be used effectively to control highly nonlinear and complex systems.

This study also investigated the application of fuzzy control in areas not fully utilizing fuzzy logic control. Three important practical applications pertaining to the automotive industries were studied. The first automotive-related application was the idle speed of spark ignition engines, using two fuzzy control methods: (1) left and right parameterization, and (2) fuzzy clustering techniques and experimental data. The simulation and experimental results showed that a conventional controller-like performance fuzzy controller could be designed based only on experimental data and
intuitive knowledge of the system.

In the second application, the automotive cruise control problem, a fuzzy control model was developed using parameters adaptive Proportional plus Integral plus Derivative (PID)-type fuzzy logic controller. Results were comparable to those using linearized conventional PID and linear quadratic regulator (LQR) controllers and, in certain cases and conditions, the developed controller outperformed the conventional PID and LQR controllers.

The third application involved the air/fuel ratio control problem, using fuzzy clustering techniques, experimental data, and a conversion algorithm, to develop a fuzzy-based control algorithm. Results were similar to those obtained by recently published conventional control based studies.

The influence of the fuzzy inference operators and parameters on performance and stability of the fuzzy logic controller was studied. Results indicated that, the selections of certain parameters or combinations of parameters, affect greatly the performance and stability of the fuzzy controller. Diagnostic guidelines used to tune or change certain factors or parameters to improve controller performance were developed based on knowledge gained from conventional control methods and knowledge gained from the experimental and the simulation results of this study.
INFORMATION TO USERS

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

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

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

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

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

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

UMI®

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
ACKNOWLEDGMENTS

This work would not have been possible without the help and support of many people. First, I would like to thank Dr. Rameshwar Sharma for his support, confidence, encouragement, advice, and valuable guidance throughout the many years of working together. I would also like to thank Dr. Subramaniam Ganesan of the Department of Computer Science and Engineering at Oakland University for his valuable guidance, stimulating thoughts and discussions, and for allowing me to use the research facilities at Oakland University. I would also like to thank Dr. Gurbux S. Alag of the Department of Electrical Engineering for his valuable and informative discussions, inputs, and feedback that made this work rich in its content and broader in its scope. I would also like to thank all members of my doctoral committee: Dr. Jerry Hamelink and Dr. Koorosh Naghshineh for their advice and guidance during my graduate studies and research. My thanks and appreciation also go to Dr. Maurice Snyder, Dr. Tim Athan, and Dr. Marcella Haghgodie at the Applied Dynamics International for their help in the experimental work conducted at their facilities. I want to express my appreciation and thanks to my wife and family, especially my parents who were very patient, understanding, and supportive. Last but not least I would like to thank Hope E. Smith who through her professionalism in the typing and editing part of this work helped greatly in producing this work in its present form.

Saad Y. Yasin

ii
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>ACKNOWLEDGMENTS</th>
<th>..................................................</th>
<th>ii</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIST OF TABLES</td>
<td>..................................................</td>
<td>ix</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>..................................................</td>
<td>x</td>
</tr>
</tbody>
</table>

## CHAPTER

### I. RESEARCH OVERVIEW ............................................. 1

- Literature Review .................................................. 12
- Dissertation Motivation .......................................... 16
- Dissertation Organization ....................................... 22

### II. BASIC MATHEMATICAL CONCEPTS OF FUZZY SETS AND NUMBERS ............................................ 25

- Introduction .......................................................... 25
- Fuzzy Sets and Crisp Sets .......................................... 26
  - Membership Function ............................................. 30
  - Fuzzy Set Operations ............................................. 33
- Linguistic Hedges and Operators ................................ 40
- Fuzzy Inference Rules ............................................. 40
- Fuzzy Logic Models ................................................ 42
- Fuzzy Knowledge Base ............................................. 43
- Fuzzy Measurements ................................................ 50
- Fuzzy Clustering ..................................................... 51
Table of Contents—Continued

CHAPTER

Extension Principle ................................................................. 54
Aggregation of Fuzzy Rules ........................................................ 57
Defuzzification Methods .............................................................. 61
Types of Fuzzy Models ............................................................... 67

Introduction .............................................................................. 67
Mamdani Fuzzy Control Model .................................................. 67
Takagi-Sugeno-Kang (TSK) Fuzzy Model ...................................... 68
Standard Additive Fuzzy Model (SAM) ...................................... 70

Fuzzy Rule-Based Expert Systems ............................................... 72

III. FUZZY SYSTEM DESIGN ...................................................... 78

Introduction .............................................................................. 78
Design of Fuzzy Systems Using Only Experimental Data .......... 78
Fuzzy Control System Using Gradient Descent Training .......... 81
Fuzzy Control Systems Design Using Recursive Least Squares .... 85

Training TSK Fuzzy Systems Using Recursive Least
Squares Methods .................................................................. 88
Design of the Fuzzy System Using Clustering ............... 91
Nearest Neighborhood Clustering Technique .................. 92

c-Means Clustering ................................................................ 94

Design of a Fuzzy Control System Using the Learning From Examples Method .............................................. 99
Table of Contents—Continued

CHAPTER

Design of a Fuzzy Logic Controller Using Conventional Control Techniques................................................................. 101

Adaptive Fuzzy Logic Control............................................................................................................................................. 102

Fuzzy Supervisory Control.................................................................................................................................................... 114

The Computational Structure of the Fuzzy Logic Controller ......................................................................................... 115

Scaling Factors.................................................................................................................................................................... 117

Denormalization................................................................................................................................................................. 118

Design of the Fuzzy Logic Controller............................................................................................................................... 119

Construction of the Knowledge Base.................................................................................................................................... 126

IV. ANALYTICAL FORMULATION OF FUZZY CONTROL LAW......................................................................................... 130

Left and Right (LR) Parameterization of the Membership Functions.............................................................................................. 134

Introduction.......................................................................................................................................................................... 134

Left and Right (LR) Parameterization............................................................................................................................... 137

Analytical Formulation of a Class of Fuzzy Logic Control Law......................................................................................... 142

Assumptions and Definitions................................................................................................................................................ 143

Analytical Membership Function Using Linear and Nonlinear Functions..................................................................................... 151

Introduction.......................................................................................................................................................................... 151

Mathematical Foundation..................................................................................................................................................... 153

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
<table>
<thead>
<tr>
<th>CHAPTER V. PRACTICAL APPLICATIONS OF FUZZY LOGIC CONTROL</th>
<th>165</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>165</td>
</tr>
<tr>
<td>A Systematic Approach for Designing a Fuzzy Logic Controller Using Analytical Method: Application to Idle Speed Engine Control</td>
<td>166</td>
</tr>
<tr>
<td>Fuzzy Control Systems</td>
<td>172</td>
</tr>
<tr>
<td>Idle Speed Control of Spark Ignition Engine</td>
<td>179</td>
</tr>
<tr>
<td>Spark Ignition Engine Control Using Fuzzy Logic: A Three Phases Control With Combined Control Variables</td>
<td>202</td>
</tr>
<tr>
<td>Basic Fuzzy Logic Controller</td>
<td>204</td>
</tr>
<tr>
<td>Universe of Discourse</td>
<td>206</td>
</tr>
<tr>
<td>Engine Control</td>
<td>208</td>
</tr>
<tr>
<td>Introduction</td>
<td>208</td>
</tr>
<tr>
<td>Problem Statement and Design Process</td>
<td>211</td>
</tr>
<tr>
<td>Results and Discussion</td>
<td>219</td>
</tr>
<tr>
<td>Fuzzy Logic Control Applications in Automotive Cruise Control and Spark Ignition Engines Air-Fuel Ratio Control</td>
<td>229</td>
</tr>
<tr>
<td>Introduction</td>
<td>230</td>
</tr>
<tr>
<td>Proposed Controller and a Review of Previous Works</td>
<td>232</td>
</tr>
<tr>
<td>Adaptive Fuzzy Logic Control</td>
<td>233</td>
</tr>
<tr>
<td>Cruise Control Problem Statement and Design</td>
<td>238</td>
</tr>
</tbody>
</table>
Table of Contents—Continued

CHAPTER

Conventional Fuzzy Logic Controller for Air/Fuel (A/F) Ratio Control ............................................................... 249

Fuzzy Logic Controller Using Normalized Spline Membership Functions for Inverted Pendulum Control Problem ................................................................. 259

The Inverted Pendulum Problem ............................................. 260

Simulation Results and Discussion .......................................... 268

VI. EXPERIMENTAL SETUP AND PROCEDURE ................................ 274

Introduction ....................................................................................... 274

Processor Selection ........................................................................... 278

Digital Signal Processors (DSPs)...................................................... 280

Microprocessor-Based Control System Design .............................. 281

DSP Processor Chosen for Simulation .................................... 284

Experimental Results and Discussion ............................................... 292

ADI Experimental Results ....................................................... 292

DSP Microcontroller and Engine Simulator Experimental Results ................................................................. 301

VII. PARAMETER SELECTIONS AND TUNING GUIDELINES FOR THE FUZZY LOGIC CONTROLLER ................................................................. 309

Introduction ....................................................................................... 309

Fuzzy Logic Controller Parameters Selection ................................ 313

Membership Function Selections ............................................. 314

Connectives and t-Norms and s-Norms ......................................... 318
### Table of Contents—Continued

**CHAPTER**

- Defuzzification Method Effects on the Controller Performance ........................................ 323
- Guidelines for Fuzzy Logic Controller Design ................................................................. 343
- Scaling Factors .................................................................................................................. 346
- Finding Membership Functions ....................................................................................... 351
- Testing and Evaluating the Fuzzy Logic Controller .......................................................... 354
- Tuning and Diagnosing the Fuzzy Logic Controller ........................................................... 356
- System Response and Stability .......................................................................................... 357
- Overshoot and Oscillatory Response .................................................................................. 358

**VIII. SUMMARY, CONCLUSION, AND SUGGESTIONS FOR FUTURE WORK** ........................................ 362

- Summary .......................................................................................................................... 362
- Discussion ........................................................................................................................ 370
- Conclusion ......................................................................................................................... 373
- Future Studies and Projects ............................................................................................... 376

**APPENDIX A** .................................................................................................................. 378

**APPENDIX B** .................................................................................................................. 392

**APPENDIX C** .................................................................................................................. 394

**BIBLIOGRAPHY** .............................................................................................................. 396

viii
LIST OF TABLES

1. Hedges and Corresponding Definition ...................................................... 41
2. Fuzzy Associate Memory (FAM) of the Mac-Whelan Template ............. 127
3. Idle Speed Fuzzy Associate Memory (FAM) Table .................................. 186
4. Experimentally Obtained Map for the Air-Mass Flow Rate as a Function of Throttle Angle and Engine Speed ............................................. 191
5. FAM Table for the Throttle as a Function of the Engine Speed Error and Its Change ....................................................................................... 214
6. Spark Advance Fuzzy Control Rules as a Function of $T_{\text{inj}}$ and $N$ .......... 216
7. FAM Table for the Error, Change of Error Inputs, and Throttle Output ......................................................................................................................... 241
8. FAM Table for the Error, Change of Error Inputs, and the Output $K_s$ .... 242
9. Fuzzy Rules Table for Manifold Pressure .............................................. 252
10. Fuzzy Control Rules for Volumetric Efficiency Control ......................... 253
11. Fuzzy Rules for the Inverted Pendulum With Four Inputs and One Output ................................................................................................................. 266
12. Spark Advance as a Function of Injection Time ($T_{\text{inj}}$) and Engine Speed ($N$) ........................................................................................................ 304
13. Throttle ($2$) as a Function of the Error ($e$) and the Error Change .......... 306
14. Common t-Norms and t-Conorms Operators ......................................... 321
15. Fuzzy Rules for the Inverted Pendulum Used for Simulation ................. 330
### LIST OF FIGURES

1. Crisp Set ................................................................................................... 26
2. Fuzzy Set on the Universe of Discourse $U$ .............................................. 28
3. S-Shape Membership Function ............................................................... 30
4. Triangular Membership Function ........................................................... 31
5. Trapezoidal Membership Function ......................................................... 32
6. Sigmoid Membership Function ............................................................... 32
7. Composition Using Sup-Min Operator .................................................... 49
8. Composition Using Sup-Product Operator .............................................. 49
9. Weighted Average Defuzzification Method ............................................. 62
10. Center of Area (COA) Defuzzification Method ....................................... 63
11. Center of Sum (COS) Defuzzification Method ........................................ 64
12. Center of Largest Area Defuzzification Method ...................................... 64
13. First/Last of Maxima Defuzzification Method ....................................... 65
14. Max-Membership Defuzzification Method ............................................. 65
15. Mean-Max or Middle of Maxima (MOM) Defuzzification Method ......... 66
16. Fuzzy System Design Using Gradient Descent Training ......................... 82
17. Conventional Direct Adaptive Control Scheme ..................................... 102
18. Model Reference Adaptive Control ......................................................... 103
19. Direct Model Adaptive Reference Fuzzy Control ................................... 104
List of Figures—Continued

20. PD-Type Direct Adaptive Reference (a) and Controller Fuzzy Supervisory Controller (b) .......................................................... 105

21. Fuzzy Logic Controller Structure Showing the Steps Involved in the Inference System ................................................................. 116

22. The Transformation From Universe of Discourse to Normalized Universe of Discourse and Its Effect on the Sliding Surface ............. 119

23. Fuzzy Logic Controller in a Control System ........................................ 135

24. Membership Function Showing Different Parameters ....................... 139

25. Universe of Discourse for Inputs (a) and Output (b), for Membership Functions .............................................................. 140

26. The Number of Rules That Can Be Fired at Any Given Inputs ............. 141

27. Typical Shape of Left and Right Parameterized Membership Functions .......................................................................................... 144

28. The Membership Function for Linguistic Value NM Defining Engine Speed Range of [0, 800] .......................................................... 161

29. Comparison Between Linear and Cubic Functional Form of Membership Function for Linguistic Value NM Defined on [0, 1500] .................................................................................. 162

30. A PI-Type Fuzzy Based Control System ............................................ 172

31. Linguistic Values Represented by Membership Functions .................. 173

32. Left and Right Parameters for Defining the Membership Function. Triangular Membership Function (Left) and General Shape (Right) ................................................................................ 174

33. The Effects of the Membership Parameters $\alpha$, $\beta$ and $\rho$ on the Shape and Width of the Membership Function ................................................. 175

34. The Parameters Shown in Equation (5.11) for Triangular Membership Function .................................................................................. 177
List of Figures—Continued

35. The Proposed Fuzzy Logic Controller Structure ........................................ 182
36. Throttle Universe of Discourse (Left). Normalized Universe of Discourses for the Inputs Error (e), Change of the Error (De), and Output (u) (Right). .................................................... 184
37. The Degree of Membership for Specific Input e for Two Adjacent Sets ........................................ 185
38. Air-Mass Flow Rate Control Surface Generated Using C-Mean Clustering .................................... 192
39. Schematic Layout of the Idle Speed Control for Spark Ignition Engine ........................................ 194
40. SIMULINK Simulation Model ................................................................. 194
41. Step Response for Idle Speed Control With Desired Idle Speed at 650 rpm .................................... 195
42. Engine Speed Error (N_{error}). Throttle Input (Top), Load Torque of 20 N.m. (Middle). Engine Speed Perturbation (Bottom). ........................................ 196
43. Engine Speed Error for Acceleration and Deceleration Cases ..................................................... 197
44. Idle Speed System Response. Load Torque (Top) and System Response (Bottom) .......................... 198
45. Idle Speed System Response for Throttle Input and Constant Load Torque .................................... 199
46. Desired and Measured Engine Speed When Load Torque Is Applied ........................................... 199
47. Random Load Disturbance Input and System Response ............................................................... 200
48. Comparison Between Linear Quadratic Gaussian (LQG) and Fuzzy Logic Controller (Solid) .......... 201
49. Proposed Engine Controller Using Two Fuzzy Controllers, One for Controlling the Throttle and the Other for Spark Advance Control. ......................................................... 204
List of Figures—Continued

50. Basic Structure of the Fuzzy Logic Controller in a Control Application ................................................................. 205

51. PI-Type Fuzzy Control Schemes Used in Practical Application .............. 207

52. Idle Speed Controller Inputs and Outputs .................................................... 210

53. Fuzzy Representation of the Throttle (Left) and Spark Advance (Right)....212

54. Normalized Universe of Discourse for Throttle (Left) and Spark Advance (Right) ................................................................. 212

55. Ignition Timing Advance Curve for Best Performance From Starting to Acceleration ......................................................... 215

56. Spark Advance Control Surface ................................................................ 216

57. The Block Diagram and Simulation Model for Spark Advance and Throttle Controls Scheme ........................ 217

58. Throttle Command for Throttle Control ..................................................... 220

59. Engine Speed Deviation From Set Idle Speed (750 rpm), While Spark Advance Is Kept Constant and No Load Torque Applied.......... 220

60. The Throttle Control With Spark Advance Kept Constant, and No Load Torque Applied .................................................... 221

61. Throttle and Spark Command With No Load Applied ......................... 222

62. Spark and Throttle Control When a Load Torque of -100 rpm Is Applied ........................................................................... 223

63. Idle Speed Error With Throttle Input, Constant Spark, and a Load Torque of -50 rpm ................................................................. 224

64. Idle Speed Change From Desired When Spark Control Is Utilized at Constant Throttle and a Load Torque of -100 rpm Is Applied......... 225

xiii
List of Figures—Continued

65. Engine Speed Error With Throttle Command, Spark Advance Set to 10° BTDC, and a Load Torque of -50 rpm Applied at $t = 8$ Seconds ...................................................................................... 226
66. Throttle Step Input Indicating Acceleration .......................................................... 226
67. Engine Speed Deviation From Set Idle Speed for Step Throttle Input and Constant Spark Advance......................................................... 227
68. The Engine Speed Error for Step Throttle Input and Spark Advance Control Command as Shown in Figure 62 .......................................................... 228
69. Typical Cruise Control Block Diagram ............................................................ 232
70. Typical Adaptive Fuzzy Logic Controller .......................................................... 234
71. Typical PD Type Fuzzy Control (a) and PI Type Fuzzy Controller (b) ................. 235
72. A PID Type Fuzzy Logic Controller .................................................................. 236
73. Adaptive PID Type Fuzzy Logic Controller ....................................................... 237
74. Normalized Universe of Discourse for the Error and Its Change Defined on the Range $[-1, 1]$.............................................................................. 239
75. Normalized Universe of Discourse for the Output On $[0, 1]$. ...................... 240
76. SIMULINK Simulation Model Used in Simulation ........................................... 243
77. Step Response of the PID Type Fuzzy Logic Controller (a) and Control Effort (b) ........................................................................................................ 245
78. Response of the PID Type Fuzzy Logic Controller When Set Speed is 45 (m/s) at $t = 5$ Seconds. ................................................................. 245
79. Comparison of the PID Type Fuzzy Logic Controller Under Different Added Resistance Percentages .......................................................... 246
List of Figures—Continued

80. Performance Comparison Between LQR/Linear PID Controllers and Proposed Controller .................................................. 247

81. The Control Efforts Corresponding to Figure 80 for the LQR/Linear PID Controllers and PID Type Fuzzy Logic Controller. .................................................. 247

82. Performance Comparison of Conventional PID, LQR, and PID Type Fuzzy Logic Controller With a Total Resistance of 15% Applied .............. 248

83. Conventional Efforts Comparison Between PID, LQR Conventional Controller, and PID Type Fuzzy Logic Controller When a Total Resistance of 15% Is Applied .................................................. 249

84. Fuzzy Generated Control Surface for Manifold Pressure .................. 251

85. Fuzzy Generated Control Surface for Volumetric Efficiency. ............. 252

86. Membership Functions for Inputs and Output of Each Controller. Manifold Pressure (Upper), Volumetric Efficiency (Lower). ............... 254

87. Throttle and Load Applied to the System at Time Indicated in the Graphs .................................................................................. 256

88. Part Throttle Air-Fuel Ratio Control Response When an Applied Load Torque as Shown in Figure 87 and Fuel Compensation Is Neglected ........................................................................ 256

89. Response for the Same Load and Throttle as Shown in Figure 87 and Fuel Compensation Is Taken Into Consideration With $f_a = f_b = 0.5$ ........................................................................ 257

90. The Response With Same Torque Load as in Figure 87, and Throttle as Shown Above; Fuel Compensation Is Neglected .................................. 257

91. Part Throttle Air-Fuel Ratio Control Response With Load Torque as in Figure 87, Throttle as Shown in Figure 90, and Fuel Compensation Factors $f_a = f_b = 0.5$ .................................................. 258

92. Inverted Pendulum System ...................................................................... 261
93. Block Diagram for the Proposed Control Schemes for Inverted Pendulum .......................................................... 262
94. Universe of Discourse and Membership Functions for the Cart Position and Velocity ........................................ 264
95. Universe of Discourse and Membership Functions for the Pendulum Angle $\theta$ ...................................................... 264
96. Universe of Discourse and Membership Functions for the Pendulum Angular Velocity .............................................. 265
97. Universe of Discourse and Membership Functions for the Inverted Pendulum Controller Output $u$ ........................................ 265
98. ANFIS Model Using TSK Fuzzy Model ............................................................................................................. 266
99. SIMULINK Simulation Diagram for ANFIS Model ............................................................................................. 269
100. SIMULINK Simulation Diagram for the Developed Model ................................................................................ 270
101. The Response for the Developed Model. Angle Response (Left), Cart Position Response (Right) ............. 271
102. The Response for the ANFIS Model. Angle Response (Left), Cart Position Response (Right) .............. 271
103. The Angle Response Comparison Between the Developed and ANFIS Models ..................................................... 272
104. SIMsystem Setup at ADI Facilities .................................................................................................................... 275
105. Diagram of Integrated Software and Hardware at ADI .................................................................................. 276
106. Diagram of AD Real Time Station (RTS) ........................................................................................................ 277
107. Block Diagram Showing How the Model Developed Is Downloaded Onto the AD RTS Using Real Time Workshop (RTW) ........................................................................................................ 278
108. Typical Use of a Microcontroller in Engine Management Control Problem .................................................. 285
List of Figures—Continued

109. Integration of the EZ-DSP Lab Board With the EZ-DSP-CPU-52 Board Showing the Analog Input (A/I), Analog Output (A/O), Digital Input (D/I), and Digital Output (D/O)................................. 289

110. Experimental Step Response for Idle Speed Control With Desired Speed Set at 650 .............................................................................................................. 294

111. Experimental Engine Speed Error for Throttle Input (Top), Load Torque (Middle).............................................................................................................. 294

112. Experimental Idle Speed Error ($N_m-N_d$) (Bottom), Load Torque (Top). 295

113. Experimental Idle Speed Control Response When a Step Load Torque Is Applied .............................................................................................................. 296

114. Comparison Between Simulation and Experimental Results for a Step Change in the Load Torque ........................................................................... 297

115. Experimental Result of Engine Speed Error (Bottom), Throttle Input (Top).............................................................................................................. 298

116. Experimental Engine Speed Error Under Throttle and Spark Control With No Load Torque Applied ........................................................................... 298

117. Experimental Spark and Throttle Control When a Load Torque of -100 rpm Is Applied ........................................................................................... 299

118. Experimental Idle Speed Error Under Throttle Control With a Load Torque of -50 rpm, and Spark Set at 10° BTDC. .................................................. 300

119. Experimental Idle Speed Error Under Spark Advance Control, Constant Throttle, and a Load Torque of -100 rpm .................................................. 301

120. Applied Load Torque of -50 rpm (top) and Engine Speed Error (bottom).............................................................................................................. 302

121. Spark Advance as a Function of Injection Time at $N = 2000$ (rpm).... 307

122. The Fuzzified Value for the Same Input With Different Membership Functions Shapes................................................................. 316

xvii

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
List of Figures—Continued

123. The Controller Output for Different Membership Functions Shapes ...... 317
124. Controller Output as a Function of t-Norms ............................................. 323
125. Effects of Different Defuzzification Methods on the Controller Output................................................................................... 327
126. Block Diagram of Proposed Simulation Model for the Inverted Pendulum With a Fuzzy Logic Controller, Which Employs the Rules in Table 15 ................................................................. 329
127. System Response for Triangular Membership Functions, Product for Connective, Product for Implication, Maximum for Aggregation, and COA Defuzzification ............................................................ 331
128. System Response for Trapezoidal Membership Function, Product for Connective, Product for Implication, Maximum for Aggregation, and COA for Defuzzification................................................................. 331
129. Response When Sigmoid Membership Functions Are Used............... 332
130. Response of the System When Different Membership Functions (Bell-Shape, Gaussian, Triangular) Are Used ................................................ 333
131. The Results of Simulation With Different Defuzzification Methods ...... 334
132. Response of the System When (MOM) Is Used......................................... 334
133. Response of the System When COS Is Used........................................... 335
134. Response of the System When the COA Method Is Used...................... 335
135. The Simulation Results for Different t-Norms Operators......................... 336
136. The Implication Operators Comparison................................................... 337
137. Comparison of the Aggregation Operator's Effect on the Fuzzy Logic Controller Performance ................................................................. 338
List of Figures—Continued

138. Combinations of Different Connectives, Implication, Aggregation, and Defuzzification Methods .......................................................... 339
139. Same Combination as in Figure 138 but With SD Defuzzification .......... 340
140. The Effect of Changing the Implication Operator While the Combination of Other Parameters Stayed the Same .......................... 340
141. Instability in Combination 4 With COA Defuzzification Method .......... 342
142. System Response for Three Combinations of Parameters .................... 342
143. Similarities Between FLC Scaling Factors and PID Parameters .......... 348
CHAPTER I

RESEARCH OVERVIEW

Many of the processes in control have complex, nonstationary, and sometimes nonlinear dynamics, which are significantly affected by unmeasured parameters and are difficult to model. Thus, precise models of process dynamics become increasingly difficult to produce as process complexity increases. This is, however, sometimes impractical in control, since precise models are required in order to produce a good controller of such systems.

Intelligent control attempts to solve difficult or complex problems by admitting process uncertainty or complexity and relaxing the specification on controlled response, rather than searching for exact solutions to simplified problems, as in conventional control. A good example can be seen in the attempt to design a control system to regulate the speed of a vehicle. This process is both complex to model and greatly affected by unknown factors, such as the environment in which the car moves, causing time-varying process parameters. As such, it poses a difficult case for classical control which, in this situation, attempts to produce exact solutions to problems that are incompletely specified. A typical solution to this problem is to use a linearized model and controller gain scheduling to produce a control system suitable for a small fixed region of the parameter range. However, a human operator can easily achieve the same thing after only a small amount of initial learning.
Control performance achieved by a human operator may not be as precise as that desired for a conventional controller, since the goals to be achieved have associated and often rather vague tolerances. The human operator, though, has many desirable features, such as the ability to reject significant disturbances, the ability to control nonlinear processes, the ability to sense and adapt to changes in system response caused by external environmental changes, and the ability to control different vehicles with only small amounts of self-learning. The human operator achieves good control performance as a consequence of a trade-off between precision and significance. The concept of fuzzy logic, based on the seminal work by Zadeh (1968), was introduced to model human reasoning by giving definitions to vague terms and allowing several rules in a rule-base to interact with varying degrees of membership. It is important to note that it is irrelevant whether humans store knowledge in this form; what is important is that fuzzy logic allows the creation of the rule-base.

Zadeh and Chang (1972) suggested the use of fuzzy logic in control. The most significant early applications came out of the research group headed by Mamdani during the 1970s. The work of Mamdani (1974) and Mamdani and Assilian (1975) reported the first successful attempt to use fuzzy logic to control dynamic systems. The first large-scale applications of fuzzy logic control techniques were very encouraging (e.g., see Abate and Dosio, 1990; Chiw, 1998; Vachtsevanos, 1993; Klawonn, 1995; Tong, 1978; Rutherford and Carter, 1976; Kickert and Van Nauta Lemke, 1976; Vanvuchelen and Vandewalle, 1995; Hessburg and Tomizuka, 1994).
Many more practical applications can be seen in the studies of different authors listed in the bibliography. The above mentioned applications that take advantage of the ability of fuzzy logic to implement rule-base are often produced by eliciting knowledge from an existing human operator, in which case the procedure is one of modeling rather than control, and the final closed loop response achieved is determined by the operator being modeled.

As one can see from the previous discussion, the need exists for finding another alternative to control that can utilize, in addition to the usual information required about the system to be controlled, the information and expertise of a human operator or intuitions.

Fuzzy logic is motivated by two objectives. First, it aims to alleviate difficulties in developing and analyzing complex systems encountered by conventional mathematical tools. Second, it is motivated by observing that human reasoning can utilize concepts and knowledge that do not have well-defined, sharp boundaries. The first motivation is directly related to solving real-world problems, while the second motivation is related to Artificial Intelligence (AI). The first motivation requires fuzzy logic to work in quantitative and numeric domains, while the second motivation enables fuzzy logic to have a descriptive and qualitative form, because vague concepts are often described qualitatively by words. These two motivations together not only make fuzzy logic unique and different from other methods that focus on only one of these goals, but also enable fuzzy logic to be a natural bridge between the quantitative world and the qualitative world. The unique
characteristic of fuzzy logic offers an important benefit. It not only provides a cost-effective way to model complex systems involving numeric variables, but it also offers a qualitative description of the system that is easy to comprehend.

Fuzzy controller uses a form of quantification of imprecise information (input fuzzy sets) to generate by an inference scheme, which is based on a knowledge base of control, a precise control force to be applied to the system to be controlled. The logical controller is made up of three parts and, in some cases, the three parts can be broken into four or five parts. The three parts are categorized by their functions. The first component is the fuzzification component, where the information is quantified by means of fuzzy sets. The second component is a fuzzy logic processing component, which converts the input fuzzy sets into control force fuzzy sets, through rules collected in the knowledge base, and aggregates the resulting fuzzy sets. The third component is the defuzzification component, where the output fuzzy information, which is an aggregated fuzzy set, is converted into a precise value. The advantage of this quantification is that the fuzzy sets can be represented by a unique linguistic expression, such as small, large, high, low, etc. The linguistic representation of a fuzzy set is known as a term, and a collection of such terms defines a termset, or library of fuzzy sets. The linguistic representation simplifies the specification of the control laws through the use of linguistic rules. The method of processing these rules to determine a control action is known as linguistic synthesis or inference mechanism. A fuzzy controller that implements linguistic synthesis is made up of two components: (1) a knowledge base of linguistic rules, or rule-base,
which relates a measured datum to a control force; and (2) an inference scheme, or mechanism, to process the rules. For a system of multiple controller inputs with each input corresponding to one fuzzy variable, which is defined over its own universe of discourse, a fuzzy rule \( R_i \) in the rule-base that relates the input \( v_i \) to an output \( u \) will have the form

\[
R_i: \text{If } a_1 \text{ is } A_{ij} \text{ and } \ldots \text{ and } a_k \text{ is } A_{kj} \text{ and } \ldots \text{ and } a_n \text{ is } A_{nj} \text{ Then } u \text{ is } U_m
\]  

(1.1)

where \( a_k \) is the \( k \)th input fuzzy variable, \( A_{kj} \) is the \( j \)th fuzzy set defined on the \( k \)th universe of discourse, \( u \) is the output fuzzy variable, and \( U_m \) is the \( m \)th fuzzy set on the output universe of discourse.

Fuzzy mapping is defined as the fuzzy rule that represents a fuzzy relation or implication. Fuzzy mapping can be generated using many implication operators that will be discussed in later chapters. The fuzzy mapping using, for example, the product-scheme between the input \( a = (a_1, a_2, \ldots, a_n) \) and the output \( u \) is defined as:

\[
\mu_{R_i}(a, u) = \mu_{A_{ij}}(a_1) \wedge \ldots \wedge \mu_{A_{nj}}(a_n) \mu_{U_m}(u)
\]  

(1.2)

where \( \wedge \) is the fuzzy “AND” operator. Given an arbitrary input set \( S \), an output fuzzy set for control \( U_i \) can be inferred from the \( i \)th rule by a composition scheme. Many types of composition schemes are available for use in fuzzy logic and will be discussed in depth in Chapters II and VII. For example, using Max-Product (MP) composition, the output fuzzy set for control can be expressed as:
Each rule yields one output fuzzy set \( U_i \); therefore, the resultant fuzzy output set \( U \) represents the combined effect of all the rules in the rule-base. This resultant set is obtained by combining the individual output sets with an appropriate aggregation scheme. Many aggregation schemes can be used. For example, if the sum aggregation scheme is used, then the resultant fuzzy output set is as given in Equation (1.4):

\[
\mu_U(u) = \frac{\sum_{i=1}^{R} \mu_{U_i}(u)}{\max_{u \in B} \left[ \mu_{U_1}(u) + \ldots + \mu_{U_m}(u) \right]}
\]

where \( B \) is the output universe of discourse. Aggregation operators will be further discussed in Chapters II and VII. The control force is obtained by defuzzification of the final output set \( U \). For single-input-single-output (SISO) fuzzy controller with a singleton fuzzification, Max-Product composition, product for implication, sum for aggregation, and center of area (COA) for defuzzification, the control force can be expressed as:

\[
f_u = \frac{\int_{y \in U} y \left( \sum_{i=1}^{R} [\mu_{A_i}(b) \mu_{B_i}(y)] \right) dy}{\int_{y \in U} \sum_{i=1}^{R} [\mu_{A_i}(b) \mu_{B_i}(y)] dy}
\]

where \( \mu_{A_i} \) and \( \mu_{B_i} \) are the input and output membership functions in the reference sets \( A_i \) and \( B_i \) from the \( i \)th rule of the rule-base, \( R \) is the number of rules in the rule-base, \( b \)
is the mapped image of a crisp input to the fuzzy controller, \( y \) is an element on the output universe of discourse, and \( f_y \) is the crisp output from the controller.

Though the details of the first actual methodology for implementing fuzzy control did not come until the work of Mamdani et al. was published in 1975, in some sense fuzzy control began in 1972 when Lotfi Zadeh published his brief article suggesting that fuzzy set theory could be used in control problems with great benefits. Zadeh noted that after the early 1950s control theory became increasingly obsessed with abstraction to the point that most papers published in the control theory field contained at least one mathematical proof, but little improvement was achieved in the practical application or performance of the systems developed. This opinion continues in the fuzzy set applications research community today, especially now that fuzzy control has very much proven itself. It seems that most researchers who write about fuzzy control have different opinions, depending on their particular purpose for explaining precisely in what ways fuzzy control is more practical than conventional methods. Because so much criticism of conventional control methods exists, only a few criticisms that comprise a basis of comparison between fuzzy and conventional control will be discussed.

The first problem of conventional control is that it is a field that is quite difficult to master. Even if one tries to overlook how highly abstract it becomes and focuses only on learning the practical application techniques, conventional control is still difficult to learn to use outside of the simplified problems. However, this may seem more of an advantage than disadvantage to some designers, like applied
mathematicians or highly experienced engineers. But when one considers the problem from the viewpoint of society overall, a methodology that is unnecessarily difficult is not preferred. Despite the difficult nature of conventional methods, there are considerable limitations to their applicability. If a method is based on rigorous analysis of a model of a plant or process, then obviously a condition for the use of that method is that the physics of the plant and its dynamic behavior must be understood well enough to develop the model. This is not always the case. Many of the earliest fuzzy control applications were for large-scale plants where it would have long been desirable to automate the control. But it was impossible to do so previously, because the plant dynamics were simply not understood well enough to create the type of model conventional control design requires. A third disadvantage of conventional control methods relates to the nature of the models of physical systems. Anyone who has dealt with control theory knows that control theory is not unique in its preference for linear models over nonlinear ones. Most systems-related fields have this same preference, but it should be known that there is a cost associated with using a linear model. When a system is defined on some real world object and a model is developed to represent that system, then, if insisting the model be linearized, one should know that it is an accurate approximation only if limited to a certain range within the system’s overall range of behavior. In some cases, only a certain range within the system behavior is desired. In that case then, the linearized model will work just fine. On the other hand, in some applications, the behavior over a wide range of values of the variables is desired, or, in some systems, the behavior may be
so highly nonlinear that a linear model will not fit well even to a narrow range of values. In these cases, a strictly linear model cannot work well. In some fields, piecewise linear models, i.e., models based on a collection of linear functions to represent various ranges in the system behavior, are useful, but control theory makes almost no use of these. This implies that there are two problems with conventional control’s strong preference of linear models. First, some applications are unsuitable for a linear treatment and thus force the designer to use less common and more difficult nonlinear methods. The other problem is that in some applications a linear model can be made, but it is accurate only for a limited range of conditions. This may result in controllers that function well near the center of their operating ranges but behave poorly in extreme conditions. This is in contrast to the fuzzy logic controller, which is well known for typically outperforming conventional controllers under extreme conditions.

These disadvantages of conventional control methods should not obscure the fact that conventional control design methods are very highly developed, and they work remarkably well for relatively simple systems where one needs to control only one variable and where it is possible to develop an accurate linear model of the plant. However, the assumption of a relatively simple system is a critical matter, and, even in this case, the level of expertise needed to effectively apply conventional control theory effectively is quite high. It is often possible, however, to use an off-the-shelf conventionally designed PID controller, install it, and tune it based on experience and trained intuition. Nonetheless, there is an overall sense of limitation to the types of
control problems that can be effectively attacked by someone with only a moderate degree of knowledge in conventional control theory. In contrast, with fuzzy control, there is much less of this sense of limitation even with only a moderate degree of knowledge. Very few assumptions need to be made about the nature of the system. It is possible to control several variables at once and even to control on the basis of intangible criteria. None of this requires one to reach the level of a true expert in the field. Fuzzy logic control is so readily applied to multivariate control specifications that it is easily possible to incorporate, as feedback control, many aspects of a system that might otherwise have been controlled only by sequential control. It is possible to view input variables in ways that make the distinction between the control device and expert system seems insignificant.

All of this is very favorable to fuzzy control, but it is important to remember that there are a few problems with it as well. Four such problems can be noticeable. First, the matter of fuzzy inference for approximate reasoning is far from trivial. Its mastery requires some serious study, even if one limits oneself to the classical operators and the debate over which operators are best to use. Second, typically a fuzzy control design typically involves some amount of knowledge of engineering work in order to determine the control rules and the linguistic variable definition. As in most types of artificial intelligent applications, it is a difficult problem to express knowledge that is wholly or partly intuitive. In some sense, the structure the rule-base should take is more obvious for a fuzzy control design than for an expert system design, but in other ways, the process is still quite difficult. It is also well known that
there is more communication difficulties in the case of industrial process control. The
difficulties can be partially overcome by simply observing the actions of experienced
human operators rather than asking them.

Third, unlike conventional control, fuzzy control incorporates no obvious way
of predicting how well a given control-device design will perform until it is actually
built. One can simulate, but that requires some sort of accurate model, which may
not be available or needed for fuzzy control design. It is one of the advantages of
fuzzy logic control design that the precise models are not necessary. However, the
model need not be linear, and overall there are some ways of overcoming this
difficulty. Furthermore, a little comfort exists in the idea that the literature
demonstrates the natural and mathematical based proofs of the robustness of fuzzy
control designs. Fourth, also unlike conventional control, fuzzy control theory in its
classical form does not provide a rigorous approach to tuning a controller if it is
found to be below standards in some aspect of its performance. However, there are
several approaches for dealing with this, some of which are using adaptive schemes,
neuro-fuzzy and the like. These and other related issues are discussed and expanded
upon in Chapter VII.

Despite these several challenges and potential drawbacks, there is now ample
evidence that fuzzy control design works well in practice. The challenge is not so
much in how to make it work, as in how to make it work as well as possible.
The idea of this dissertation was based on the previous research of other authors. Their research encompassed many aspects of fuzzy logic control. Due to the importance of fuzzy logic control in present and future applications, this dissertation attempts to address, in specific areas, certain aspects of fuzzy logic control, and therefore it contributes to the ongoing research in three very important areas. The first area this dissertation addresses is the ability to develop and apply analytically based fuzzy logic control law by incorporating certain assumptions and modifications and by utilizing the advantages offered by certain well known mathematical functions, such as the spline-base and normalized spline-base functions. The second important area is as important as the first, if not more so, and deals with fuzzy based control systems and their application in the automotive industry, especially in the area of idle speed control, cruise control, and air/fuel ratio control. The third area deals specifically with issues related to the fuzzy control system, such as design, simulation, diagnosis, and implementation. Before expanding any further on this dissertation's contributions to ongoing research, a brief review of previous research, which forms the basis of the work in this dissertation, is presented.

The first successful application of an automatic feedback control system used in the industrial process was developed in 1769. It was called Watt's flyball governor control system, designed by James Watt for controlling the speed of a steam engine. Conventional control techniques have been applied extensively in control
applications ever since. Conventional control methods are studied in the first course textbooks of control system theory and applications. For a simple and linearized plant model, any of the conventional control techniques can be applied with variations in their formats. Naturally, any form of new control methods is compared in certain aspects to the well-established and application-proofed conventional control methods. In some aspects, the well-experienced control designer or applied mathematician uses his/her experience and knowledge to compare and study the similarities between the new proposed technology and the well-established one.

Fuzzy logic is not unique in this regard. Since fuzzy logic is relatively new and innovative technology, it has been compared to and has benefited from conventional control methods. As is the case with model-based control methods in conventional control techniques, Babuska and Verbrugyen (1995) discussed such an approach. The approach they used is based on constructing a fuzzy model of the plant; then, by using the model, a fuzzy control law is derived. For a particular class of fuzzy models with linear rule consequent, such as the case of the Takagi-Sugeno-Kang (TSK) fuzzy model, the generalized version of pole placement or minimum variance control methods was applied in the author's model. The sliding mode control technique is an advanced conventional control method. It has been applied to linear time invariant control systems that can be represented in a state space model with relatively good results. Lin and Kung (1992) used the concept of a variable structure controller combined with a conventional fuzzy logic controller to develop the sliding mode fuzzy logic controller. Wu and Liu (1996) formulated the fuzzy control to become a
class of variable structure system controls. The sliding modes in their study are used
to determine the best values for parameters in fuzzy control rules. Ting et al. (1996)
used the concepts from the sliding mode control to construct a fuzzy control scheme.
The sliding surface in their study provided estimation to the universe of discourse on
which the fuzzy control rule is based and, modified the rules according to an
adaptation mechanism.

The design methods for conventional linear controllers are used as a starting
point for tuning a fuzzy controller in the work of Gorez et al. (1995). In their study on
the flexible robot arm, a model-based state-feedback linear controller and a fuzzy
logic controller were developed and tuned simultaneously. The results of the fuzzy
logic controller were shown to outperform conventional state-feedback linear
controller. Shenoi et al. (1995) studied the development of the fuzzy controller using
hardware. The controller developed is similar to gain scheduling conventional control
methods, except it incorporates an adaptive scheme for off- and on-line tuning.
Fuzzy clustering techniques, combined with the TSK fuzzy model, can be used to
identify the inputs and outputs of a nonlinear control system. Such is the case in a
study conducted by Zhao and Gorez (1995). The aim of their study was to use the
fuzzy clustering technique first to identify the inputs and outputs of the control
system and then to develop a model in order to design a fuzzy system consisting of
linear subsystems. These subsystems would, in turn, facilitate the design and analysis
of the overall control systems. Qiao and Mizumoto (1996), on the other hand, used
knowledge gained from the conventional linear proportional plus integral plus
derivative (PID) controller to design a PID-type fuzzy logic controller. The controller is also supported by an adaptive scheme to tune its parameters. This approach, however, is applicable only when the consequent parts of the rules are crisp.

The model developed by Qiao and Mizumoto (1996) is used in the present study in its modified form so that the fuzzy consequent parts of the rules are not limited to being crisp. The model is shown to perform better than the conventional PID controller and linear quadratic controller, as will be seen in Chapter V when the practical applications of fuzzy control systems in a different format are discussed.

Adaptive control schemes in two forms, direct and indirect, are used in fuzzy control systems to design fuzzy adaptive controllers. Moore and Harris (1992) proposed an indirect adaptive fuzzy controller. Direct and indirect adaptive control schemes are discussed in detail in Chapter III. Petrov et al. (1997) investigated the hybrid fuzzy-neural controller. The structure of the proposed controller consists of fuzzy logic controller components and two neural networks. The neural networks work as an adaptive rule estimator for online acquisition and modification of the fuzzy rules and an adaptive neural defuzzifier. Kim et al. (1995) proposed a fuzzy control system design method using genetic algorithms. The study shows how genetic algorithms can optimize the performance of the fuzzy logic controller. Knowledge gained from the conventional proportional plus derivative (PD) controller used to design a fuzzy system was presented in the work of Kawaji et al. (1991). The control law was expressed mathematically based on the similarities between the conventional PD controller and PD type fuzzy logic controller. The same approach was used by
Yager and Filev (1994) to develop the proportional plus integral (PI) type fuzzy controller and proportional plus integral plus derivative (PID) type fuzzy controller using knowledge gained from conventional PI and PID controllers. Passino and Yorkovich (1998) expressed the fuzzy control law mathematically according to the membership functions type, the defuzzification methods employed, and other simplification assumptions, such as crisp consequents of the fuzzy rules. A similar approach was also used by Wang (1997) in which the fuzzy system is expressed mathematically under certain assumptions and conditions. For each set of fuzzification, defuzzification, and other inference engine operators, a fuzzy system is expressed analytically. Langari (1992) used the characteristic functions to obtain an analytical formulation of the fuzzy control law. The control law is based on using the left and right (LR) parameterization to shape a nonlinear function that maps the process error and its rate of change or integral into the appropriate control action. Fuzzy control stability is established using this approach under certain conditions. The Langari approach is used as the basis of the analytical model developed in the present study. This will be elaborated on in depth in Chapter IV, and the design and application of the fuzzy controller based on the left and right parameterization to idle speed control problem will be discussed in Chapter V.

Dissertation Motivation

The main motivation of this dissertation is to shed light on this new but highly promising field of fuzzy logic and its use in control systems, particularly in the field
of mechanical engineering control applications. The contribution of this dissertation falls into three categories, the first of which is the development of the analytical fuzzy model. Two such models are discussed and developed. The first model is based on previous work of Langari (1992), and the second model utilizes the advantages of the mathematical foundations of the spline-base functions. In the first model, an analytical expression of the fuzzy control law is developed using the left and right (LR) parameterization to express the membership functions of the fuzzy linguistic values for the inputs and outputs of the process. The original motivation of the model developed by Langari (1992) was to study the stability of a fuzzy control law. This dissertation, however, takes the model a step forward, in that the mathematically based and derived model is modified and used to design a practical fuzzy controller and to apply it in a real-life application, such as the idle speed control problem. The Langari model is applicable to regulatory type control system, while the modified model is applicable to either tracking or regulatory types control systems.

The second form of analytical expression is based on the use of a class of spline-base and normalized spline-base membership functions. This model is developed for the purpose of expressing the membership functions in a generalized analytical form. The developed expression has the most often used membership function types in present fuzzy logic control applications, such as triangular, trapezoidal, bell-shaped, or Gaussian type membership functions as special cases. The benefits of doing this can be summarized in three areas, the first of which is the ability to generalize the expression of the membership function and therefore the
fuzzification process. Generalization of the fuzzification process will ultimately lead to the ability to design an adaptive and selective fuzzy control system. This can be achieved only if other parts of the fuzzy logic control components, such as the one mentioned earlier, are also developed in such a way that the overall control logic can be generalized under certain categories for certain classes of control problems. This means that the implication, composition, aggregation, defuzzification operators, and methods need to be formulated based on solid ground that leads to the generalizations of the fuzzy control law. Thus, this benefit is a step forward toward that goal. The second benefit of generalizing the fuzzification process is that it simplifies the design process as it exists and is used today. Thus, if one limits the selection process to the best choice of the membership functions, then incorporating an adaptive scheme into the control system design software will be much easier and faster. The third benefit is purely a mathematical one, in which a complete study and analysis of the fuzzy control system law will allow the designer to analyze the control law for stability issues or for comparison with the conventional control methods. Therefore, the designer will be able to design controllers to fit specific system requirements and performance specification.

In recent studies, the focus was on the development of a systematic way to design the fuzzy logic controller. Different studies dealt with different aspects of the design process and different components of the fuzzy inference engine. The defuzzification process has received great attention because of its importance in the inference system and in its promising prospects of yielding a better understanding and
a more simplified design process. A most recent attempt to express defuzzification methods in a generalized form was developed by Yager and Filev (1993). Through the practical use of the defuzzification method developed by Yager and Filev in this dissertation, the results indicate that the developed method is an important step forward. The simulation results shown in Chapter VII are very promising. More diverse application of the developed method, however, needs to be conducted before a final judgment is rendered on its generalization prospects. The work in the area of implication and compositions is nearing standardization, although it is not generalized, but such standardization will help greatly in that regard. The aggregation operators and method can be standardized using the same approaches, as is the case for implication operators. For example, the Simple Additive Model (SAM), which was developed by Kosko (1997), uses a sum aggregation operator. The standardization of each component of the fuzzy control logic will ultimately yield to the generalization of the design process of fuzzy logic controllers. If that happens, then most of the advantages of conventional control methods in the area of system analysis, specifications satisfaction, and the ability to achieve a robust and optimal control system will disappear, because the same things can be done using fuzzy logic control. Some studies have shown great progress in achieving that goal without the use of generalization of the control design process. The work in this dissertation in that regard is a good contribution, even though it will benefit only a class of control problems in which the control law using the generalized form can be derived and expressed analytically. However, the analytical models that are developed or
modified in this dissertation should provide insight and guidelines for developing a
more complete model that covers the majority of control problems. Different fuzzy
control models are included in this dissertation to show that, in some situations, using
them is sometimes more effective than using the usual conventional control models.

The second contribution of this dissertation is the extension of the fuzzy logic
control application into the field of the automotive industries, specifically the area of
engine management control. A most active research area in that regard is the area of
idle speed control, because of its great economical and environmental importance.
Therefore, the control of idle speed for the spark ignition engine, as it pertains to
mechanical and environmental engineering disciplines, is studied in-depth. The
control of idle speed is a very important control problem that lies within the class of
nonlinear control systems. The conventional control methods that have been used for
the idle speed control problem are based mostly on advanced control techniques, such
as adaptive control (Ault et al., 1994), sliding mode (Hendricks, 1995), conventional
event-based control (Hendricks et al., 1994), as well as a variation of fuzzy control
techniques (Klawonn et al., 1995; Vachtsevanos et al., 1993; Abate and Dosio, 1990;
Feldkamp and Puskorius, 1993; and Wills et al., 1998). The references cited above
are discussed in depth in Chapter V, where practical applications are considered.
Modifying and applying the previously developed fuzzy control model in an idle
speed control problem, which has been done in this dissertation, is an important
contribution that adds another control method to the pool of existing ones. The idle
speed control problem is also included to show the practical application of such
analytically expressed fuzzy control laws and the validation of the model when used in the application of such highly nonlinear characteristics. The developed model is shown to work well for these kinds of nonlinear control problems. The development and modification of the model is discussed in-depth in Chapter IV. In Chapter V, the developed model, as applied to the idle speed control of the spark ignition engine, is discussed in-depth, along with the simulation results obtained using the developed model. Fuzzy logic implementation of developed control models suggested in this dissertation works well and yields results that are very comparable to results obtained using conventional control techniques. In addition to applying fuzzy logic to control idle speed, other applications of fuzzy logic are also demonstrated, such as automotive cruise control problem and air/fuel ratio control problems. The results of these two applications are also very promising and compare well with the current applied control methods, as well as methods that are currently in their research stages. Also, in Chapter V, the analytical model developed to express the membership functions using generalized spline functions is applied to a benchmark nonlinear control problem exemplified by the inverted pendulum control problem.

The third contribution that this dissertation aims to achieve is to shed some light on the parameter selections problem of the fuzzy logic controller. The parameters selection of a fuzzy controller, such as the connective operators of "AND," "OR," and "ALSO," is not a standardized process and is an ad hoc selection process. Many operators have been derived; some are used extensively; others have been developed mathematically but have not been used in applications, and still
others can be developed if needed. Until the effects of these factors on the composition process are recognized, any attempt to choose one operator over another will be an application dependent and will be left to the designer's preference. The work in this dissertation goes a step further and shows the effects of choosing different operators on controller performance and stability. The effects of implication operator and aggregation operator selection, along with the effects of selecting different fuzzification and defuzzification methods on controller performance and stability, are also studied and an in-depth investigation is presented. This parametric study then concludes with guidelines and diagnostic procedures that are outlined for quick tuning and fixing of the developed controllers that initially do not perform satisfactorily and within the system specifications. The main contribution of the parametric study conducted in this dissertation is in transforming these kinds of studies from their more traditional and purely mathematical forms into real-life and real-time applications. Therefore, the effects on the performance and stability of the designed fuzzy controller will be noticed and seen directly as the controller is applied in the feedback control system.

Dissertation Organization

The dissertation is organized so that the basics of the fuzzy control logic and the most commonly used fuzzy system design methods are presented first in Chapters II and III. In Chapter II, the basics of fuzzy set theory, the basic structure of fuzzy systems and their functions, and the mathematical foundations of the control system
design and synthesis are presented. In Chapter III, different methods of designing fuzzy control systems, which concentrate on using experimental data in the design of fuzzy systems employing different techniques, are presented. Also, adaptive control schemes in two forms, direct and indirect adaptive control and supervisory control, are discussed. The aim of Chapter III is to show, in cases where expert knowledge is not enough to design a control system or is not available at all, how one can use the experimental data and knowledge gained from conventional control methods to design a fuzzy logic control system. Chapters II and III are included to provide the background and the theoretical foundation for the work conducted in later chapters.

In Chapter IV, the analytical models that are developed using the left and right parameterization (LR) and the generalized spline function are stated and discussed. Their mathematical foundations are explained, and their applications to fuzzy logic control are discussed. Chapter IV forms the foundation of two practical applications that are presented in Chapter V. Chapter V is devoted to the practical applications of fuzzy logic control as it is applied in different formats. For example, the adaptive fuzzy logic control system discussed in Chapter III is applied in a modified format to control automotive cruise control. The design of a fuzzy system using fuzzy clustering and the neuro-fuzzy methods discussed in Chapter III is applied to design an idle speed fuzzy controller and air/fuel ratio fuzzy controller using experimental data and fuzzy clustering techniques. Chapter V concludes with the use of normalized spline membership functions to develop a fuzzy control system for the inverted pendulum control problem.
In Chapter VI, the experimental setup, procedure, and results for the two fuzzy logic controllers that are developed for the idle speed control problem are presented and compared to the simulation results presented in Chapter V.

In Chapter VII, the parameter selections of the fuzzy logic controller are discussed, and different simulation results using different parameters for different fuzzy logic controller components are presented. The effects of other parameters on the stability and performance of the fuzzy logic controller, such as the overlapping ratio and scaling factors, are also discussed. The chapter will conclude with the development of a diagnostic scheme and helpful guidelines to improve the design or to fix deficiencies in the controller performance. The dissertation concludes in Chapter VIII with the discussion of the dissertation results, contributions, and suggestions for future studies and projects.
CHAPTER II

BASIC MATHEMATICAL CONCEPTS OF FUZZY SETS AND NUMBERS

Introduction

The lack of crispness is an aspect of many real world properties and one that must be taken into consideration in defining the linguistic terms used to describe these properties. Fuzzy sets have the characteristic of defining the uncertainties in real world problems, and, for that reason, fuzzy sets have a major advantage over crisp sets. Fuzzy sets allow the description of a system and its desired performance in linguistic terms rather than in terms of relationships between precise numerical values. In the past few years, fuzzy sets and fuzzy logic have been used in a wide range of problem domains. These include pattern recognition and classification, economics, operational research, decision-making, and process control. Since 1985 there has been a strong growth in their use in control, particularly in nonlinear, ill-defined, time-varying, and complex problems. The discussion throughout this dissertation is focused on the applications of fuzzy control in systems dynamics and control area of mechanical engineering. Other applications of fuzzy control are as important as the one discussed in this dissertation, but for abbreviation purpose they are not discussed in depth. The bibliography should be consulted for related topics in that regard. In the following section, an introduction to fuzzy sets is given and a comparison between fuzzy sets and crisp sets is made.
Fuzzy Sets and Crisp Sets

Crisp set is defined in Boolean algebra as the set that is defined on a given universe of discourse, with all elements in the universe of discourse having full membership in the crisp set, and elements outside the domain or universe of discourse having zero membership in the crisp set. If \( A \) is a crisp set defined on universe of discourse \( R \), then the crisp set can be defined as:

\[
X_A = \begin{cases} 
1 & x \in A \\
0 & x \notin A 
\end{cases}
\]

(2.1)

where \( X_A \) indicates the membership of element \( x \) in the crisp set \( A \). Graphically, the crisp set is depicted in Figure 1.

![Figure 1. Crisp Set.](image-url)
Fuzzy sets, on the other hand, are based on the idea that the range of the characteristic function can be extended to cover the real numbers in the interval \([0, 1]\). The membership value assigned to an element in the universe of discourse is no longer restricted to just two possibilities, 0 and 1, but can be 0, 1, or any value in between. The characteristic function or the membership function, as it is often called, takes on any value in the range of \([0, 1]\). This is clearly an example of multi-valued logic. The apparent similarity between fuzzy membership and probability can give rise to confusion. Both involve a "value" drawn from the interval \([0, 1]\), but the interpretation of this value is different in the two cases. The fuzzy membership gives a measure of judgment, whereas the probability indicates the proportion of times the result is true in the long run.

A fuzzy set is a more complex entity than a crisp set, containing more information, but its greater accuracy as a representation simplifies rules and procedures based on it. This is an example of the trade-off between doing simple things with complex entities and doing complex things with simple entities, which often arises in system design.

As with any mathematical system, the application of fuzzy sets to the real world involves judgment and interpretation. In combining fuzzy sets, for example, discrepancies can arise between the resulting fuzzy set and the intended linguistic interpretation, unless care is taken. In general, considerable care may be needed in defining the membership functions of the fuzzy sets used.
The theory of fuzzy sets deals with fuzzy sets defined in a universe of discourse. A fuzzy set can be regarded as a general concept of an ordinary set. For a given universe of discourse $U$, a fuzzy set is determined by a membership function which maps members of $U$ onto a membership range $[0, 1]$, rather than the discrete values 0 and 1, as with crisp sets. If one lets $U$ to be a collection of objects denoted by $\{u_i\} = \{u_1, u_2, \ldots, u_n\}$, $U$ is then called the universe of discourse and $u_i$ ($i=1,2,\ldots,n$) represents an element of $U$. A fuzzy set $A$ in a universe of discourse $U$ is characterized by a membership function $\mu_A$, which taken values in the interval $[0, 1]$, namely, $\mu_A : U \rightarrow [0, 1]$. This is shown in the Figure 2.

![Figure 2. Fuzzy Set on the Universe of Discourse U.](image)

A fuzzy set $A$ in $U$ is usually represented as a set of ordered pairs of elements $u$ and its grade of membership value can be represented as shown in Equation (2.2a):
\[ A = \{(u, \mu_A(u)) / u \in U\} \] (2.2a)

For discrete case the fuzzy set can be expressed as shown in Equation (2.2b):

\[ A = \sum \frac{\mu_A(u_i)}{u_i} = \frac{\mu_A(u_1)}{u_1} + \frac{\mu_A(u_2)}{u_2} + \cdots + \frac{\mu_A(u_n)}{u_n} \] (2.2b)

while for continuous universe of discourse, a fuzzy set \( A \) can be written as shown in Equation (2.2c):

\[ A = \int \frac{\mu_A(u)}{u} \] (2.2c)

In the above equations, the sum \( \sum \), the addition (+), and the integration symbol (\( \int \)) refer to set union rather than to arithmetic summation, addition, or integration. The division sign (/) is used to connect any element with its membership value and has no connection to arithmetic division. Common terms that are often used to describe a fuzzy set are as follows:

Support: The support of a fuzzy set is defined as the crisp set of all points \( u \in U \) such that \( \mu_A(u) > 0 \). The element \( u \in U \) at which \( \mu_A(u) > 0.5 \) is called the crossover point. A fuzzy singleton is a fuzzy set whose support is a single point in \( U \).

\( \beta \)-Cut: A \( \beta \)-cut set of a fuzzy set \( A \), labeled as \( A_\beta \) is the crisp set of all points \( u \in U \) such that \( \mu_A(u) \geq \beta \).

Normal fuzzy set: A normal fuzzy set is a fuzzy set \( A \) that has at least one element \( u \) in the universe of discourse \( U \), i.e., \( u \in U \), such that \( \mu_A(u) = 1 \).

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Membership Function

There are two ways to define the membership for fuzzy sets:

1. Numerical: The degree of membership function of a fuzzy set is expressed as a vector of numbers whose dimension depends on the level of discretization, i.e., the number of discrete elements in the universe of discourse.

2. Functional: The membership function of a fuzzy set is expressed analytically, which allows the membership grade for each element in the defined universe of discourse to be calculated. The most common membership functions shapes (Ross, 1995; Reznik, 1997; Yen and Langari, 1999) are as follows:

   a. S-shape membership functions, which is defined as shown in Equation (2.3) and depicted graphically as shown in Figure 3.

   \[ S(u; a, b, c) = \begin{cases} 
   0 & \text{if } u < a \\
   2 \left( \frac{u - a}{c - a} \right)^2 & \text{if } a \leq u \leq b \\
   1 - 2 \left( \frac{u - c}{c - a} \right)^2 & \text{if } b \leq u \leq c \\
   1 & \text{if } u > c 
   \end{cases} \]  

   (2.3)

   ![](image)

   Figure 3. S-shape Membership Function.
b. $\pi$-membership function (PMF), which is defined by parameters $(b, c)$ and

S-membership function, given by Equation (2.4).

$$\pi(u, b, c) = \begin{cases} S(u; c-b, \frac{c-b}{2}, c) & u \leq c \\ 1 - S(u; c, \frac{c+b}{2}, c+b) & u \geq c \end{cases}$$

(2.4)

c. Triangular membership functions, as expressed by Equation (2.5) and depicted graphically as shown in Figure 4.

$$T(u; a, b, c) = \begin{cases} 0 & u < c; u > c \\ \frac{u-a}{b-a} & a \leq u \leq b \\ \frac{c-u}{c-b} & b \leq u \leq c \end{cases}$$

(2.5)

Figure 4. Triangular Membership Function.

d. Trapezoidal membership function which is defined by Equation (2.6) and depicted graphically in Figure 5.
32

Figure 5. Trapezoidal Membership Function.

e. Sigmoid membership function, which is expressed as shown in Equation (2.7) and depicted graphically as shown in Figure 6.

\[
Sig(u; a, b) = \frac{1}{1 + e^{-\alpha(u-b)}}
\]

(2.7)

Figure 6. Sigmoid Membership Function.

f. Gauss membership function which is given by Equation (2.8):

\[
T_{pr}(u; a, b, c, d) = \begin{cases}
0 & u \leq 0; u > d \\
1 & b \leq u \leq c \\
\frac{u-a}{b-a} & a \leq u \leq b \\
\frac{d-u}{d-c} & c \leq u \leq d
\end{cases}
\]

(2.6)
where \( a \) and \( \sigma \) are the center and width of the membership function, respectively.

g. Bell curve membership function, which is given by Equation (2.9):

\[
B(u; a, b, c) = \frac{1}{1 + \left| \frac{u - a}{b} \right|^{2a}}
\]

where \( \alpha, a, \) and \( b \) are parameters that define the slope, center, and width of the membership function, respectively.

**Fuzzy Set Operations**

The use of fuzzy sets provides a basis for the systematic manipulation of vague and imprecise concepts using fuzzy set operations performed by manipulating the membership functions. The following fuzzy set operations are the most important for fuzzy control applications:

1. Equality: Two fuzzy sets \( A \) and \( B \) are equal if they are defined on the same universe of discourse \( U \) and their corresponding membership functions \( \mu_A \) and \( \mu_B \) are equal, i.e., \( \mu_A = \mu_B \) for all \( u \in U \).

2. Union (\( \cup \)): The union of two fuzzy sets \( A \) and \( B \) with membership functions \( \mu_A \) and \( \mu_B \) is the fuzzy set whose membership function is given by Equation (2.10).
\[ A \cup B = \mu_{A \cup B}(u) = \text{Max}\{\mu_A(u), \mu_B(u)\} \quad \forall u \in U \] (2.10)

Any other t-conorm operators other than the Maximum operator can be used.

3. Intersection (\(\cap\)): The intersection of A and B is a fuzzy set whose membership function is given by Equation (2.11):

\[ A \cap B = \mu_{A \cap B}(u) = \text{Min}\{\mu_A(u), \mu_B(u)\} \quad \forall u \in U \] (2.11)

Any other t-norm operators other than the Minimum operator can be used.

4. Complement: The complement of a normalized fuzzy set A with membership function is defined as the fuzzy set on the same universe of discourse with membership function given as \(v_A(u) = 1 - \mu_A(u)\) where \(v_A\) is the complement membership function.

The definitions of union and intersection of fuzzy sets defined above are not the only definitions. Many so called t-norms and s-norms or t-conorms are defined and will be explored later when the design of fuzzy logic controller and parameter selections are considered.

5. Normalizing: Normalizing is the re-scaling of the membership function so that its maximum value is 1. The normalization process can be expressed mathematically as shown in Equation (2.12):

\[ \mu_{N(A)}(u) = \frac{\mu_A(u)}{\text{Max}(\mu_A(u))} \quad \forall \quad u \in U \] (2.12)
This should not be confused with normalizing the universe of discourse, which deals with choosing a scaling factor such that the universe of discourse is transformed into the range of $[-1, 1]$ or $[0, 1]$. This will be discussed in depth when the design issues are addressed.

6. Dilation and Concentration: A fuzzy set $A$ can be dilated or concentrated by modifying its membership function. In the concentration process, the highest element’s membership function is accentuated, which can be accomplished mathematically as shown in Equation (2.13):

$$
\mu_{\text{con}(A)}(u) = \{\mu_A(u)\}^n \quad \text{where } n > 1
$$

(2.13)

To increase the importance of the lower membership elements, the membership function can be dilated, which can be accomplished using dilation. Dilation is given mathematically as shown in Equation (2.14):

$$
\mu_{\text{dil}(A)}(u) = \{\mu_A(u)\}^m \quad \text{where } m < 1
$$

(2.14)

where $n$ and $m$ in concentration and dilation are the strength to which the two operators act on the membership function, respectively.

7. Intensification: Intensification moves the normalized fuzzy set closer to being a crisp set, by enhancing the membership value of the elements with a membership function of 0.5 or above, and diminishing the membership value of
elements with less than 0.5 membership value. The mathematical expression of doing so is shown in Equation (2.15):

$$
\mu_{\text{set}, A}(u) = \begin{cases} 
2(\mu_A(u))^2 & 0 \leq \mu_A(u) \leq 0.5 \\
1 - 2(1 - \mu_A(u))^2 & 0.5 \leq \mu_A(u) \leq 1 
\end{cases} \tag{2.15}
$$

Among the t-norms and t-conorms operators that are of relative importance in fuzzy set theory, which will be discussed further in Chapter VII, are:

1. **Algebraic product**: The algebraic product of two fuzzy sets $A$ and $B$ with membership functions $\mu_A(u)$ and $\mu_B(u)$ is a fuzzy set with a membership function given as shown in Equation (2.16):

   $$
   \mu_{A \otimes B}(u) = \mu_A(u) \otimes \mu_B(u) \quad \forall u \in U \tag{2.16}
   $$

2. **Bounded sum**: The bounded sum of fuzzy set $A$ and $B$ with membership function $\mu_A(u)$ and $\mu_B(u)$, respectively, is a fuzzy set with membership function given by Equation (2.17):

   $$
   \mu_{A \oplus B}(u) = \min\{1, \mu_A(u) + \mu_B(u)\} \quad \forall u \in U \tag{2.17}
   $$

3. **Bounded product**: The bounded product of two fuzzy sets $A$ and $B$ with membership function $\mu_A(u)$ and $\mu_B(u)$ is the fuzzy set with membership function given by Equation (2.18):

   $$
   \mu_{A \odot B}(u) = \max\{0, \mu_A(u) + \mu_B(u) - 1\} \quad \forall u \in U \tag{2.18}
   $$
4. Drastic product designated by a membership function is given by Equation (2.19):

\[
\mu_{A \lor B}(u) = \begin{cases} 
\mu_A(u) & \text{if } \mu_B(u) = 1 \\
\mu_B(u) & \text{if } \mu_A(u) = 1 \\
0 & \text{otherwise}
\end{cases}
\] (2.19)

5. Probabilistic product is given by Equation (2.20):

\[
\mu_{A \otimes B}(u) = \mu_A(u) + \mu_B(u) - \mu_A(u) \mu_B(u)
\] (2.20)

Another parameterized definition that would give broader characterization of operations representing the union and intersection is the representation theorem, which states that each fuzzy set \( A \) in \( U \) can be represented as:

\[
A = \sum \alpha A_{\alpha} \quad \alpha \in [0,1]
\] (2.21)

where \( A_{\alpha} \) is an \( \alpha \)-cut of \( A \) and \( \sum \) is in the set-theoretic sense and not arithmetic sum, and the \( \alpha A_{\alpha} \) denotes the fuzzy set whose membership degrees are as shown in Equation (2.22):

\[
\mu_{\alpha A_{\alpha}}(u) = \begin{cases} 
\alpha & u \in A_{\alpha} \\
0 & \text{otherwise}
\end{cases}
\] (2.22)

The normalized distance between two fuzzy sets \( A \) and \( B \), both defined in \( U = \{u_1, u_2, \ldots, u_n\} \) with membership functions \( \mu_A(u) \) and \( \mu_B(u) \), respectively, can be
found by using two methods:

1. Normalized linear (Hamming) distance, which can be expressed mathematically as shown in Equation (2.23):

\[
L(A, B) = \frac{1}{n} \sum_{i=1}^{n} |\mu_A(u_i) - \mu_B(u_i)|
\]  
(2.23)

2. Normalized quadratic (Euclidean) distance, as expressed in Equation (2.24):

\[
Q(A, B) = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (\mu_A(u_i) - \mu_B(u_i))^2}
\]  
(2.24)

Fuzzy Relations

In the rule-base of a fuzzy control system, the rules are usually expressed in the form of If-Then statements of the sort If \( X \) Then \( Y \). A fuzzy relation \( R \) between two sets \( X = \{x\} \) and \( Y = \{y\} \) is defined as a fuzzy set in the Cartesian product \( X \times Y \) given by Equation (2.25):

\[
R = \{\frac{\mu_R(x,y)}{(x,y)}\} \quad \forall (x,y) \in X \times Y
\]  
(2.25)

where \( \mu_R(x,y): X \times Y \rightarrow [0, 1] \) is the membership function of the fuzzy relation \( R \), and gives the membership degrees to which the elements \( x \in X \) and \( y \in Y \) are in the relation.
R to each other. Fuzzy relation composition can be achieved through different methods, such as:

Max-Min composition: The Max-min composition of two fuzzy relations R in \( X \times Y \) and S in \( Y \times Z \), written as \( R \circ S \), is a fuzzy relation in \( X \times Z \) given by Equation (2.26):

\[
\mu_{R \circ S}(x, z) = \max_{y \in Y} \left[ \mu_R(x, y) \land \mu_S(y, z) \right] \quad \forall x \in X \text{ and } z \in Z
\]  

(2.26)

where the symbols \( \land \) stands for the minimum operator and \( \lor \) for the maximum operator.

In terms of \( t\)-\( s \) norms, the following generalized \( t\)-\( s \) norm composition can be stated for two fuzzy relations R and S to yield a fuzzy relation \( U \) in \( X \times Z \) as:

\[
\mu_{R \circ S}(x, z) = \max_{y \in Y} \left[ \mu_R(x, y) \cdot \mu_S(y, z) \right] \quad \forall x \in X \text{ and } z \in Z
\]  

(2.27)

Max-Product composition: The Max-product composition of two fuzzy relations R in \( X \times Y \) and S in \( Y \times Z \), written as \( R \circ S \), is a fuzzy relation in \( X \times Z \) given by Equation (2.28):

\[
\mu_{R \circ S}(x, z) = \max_{y \in Y} \left[ \mu_R(x, y) \circ \mu_S(y, z) \right] \quad \forall x \in X \text{ and } z \in Z
\]  

(2.28)

More discussion on fuzzy relations and composition and how they relate to fuzzy logic control will be presented in later sections.
For a collection of fuzzy sets \( A = \{A_i\} \) and their universes of discourse \( U = \{U_i\}, (i = 1, 2, ..., n) \), the Cartesian product of \( A_1 \times A_2 \times \ldots \times A_n \) in \( U_1 \times U_2 \times \ldots \times U_n \) is a fuzzy set \( F \) with a membership function that can be obtained using:

1. Min-operator, which is expressed mathematically as:

\[
\mu_F(u_1, u_2, ..., u_n) = \min\{\mu_{A_1}(u_1), ..., \mu_{A_n}(u_n)\}
\]  

(2.29a)

2. Product-operator, which is expressed mathematically as:

\[
\mu_F(u_1, u_2, ..., u_n) = \mu_{A_1}(u_1) \cdot \mu_{A_2}(u_2) \cdot \ldots \cdot \mu_{A_n}(u_n)
\]

(2.29b)

where \( F \) is defined as \( F = A_1 \times A_2 \times \ldots \times A_n \).

**Linguistic Hedges and Operators**

A fuzzy set can be regarded as corresponding to a linguistic value such as "small" or "high," and a linguistic variable such as "pressure" can be regarded as ranging over such linguistic value. One powerful aspect of fuzzy sets is the ability to deal with linguistic quantifiers or "hedges." Hedges such as "very", "not very", "slightly," etc., correspond to modifications in the membership function of the fuzzy set involved. Table 1 lists some of the hedges and their definitions.

**Fuzzy Inference Rules**

A fuzzy rule is often expressed in the form of "If-Then," as was discussed in the previous section. This form of expressing the rule constitutes a fuzzy relation.
A fuzzy relation, \( R \), is also called a fuzzy implication. The fuzzy relations in a fuzzy knowledge base system can be defined as a set of fuzzy implications or relations.

Two types of fuzzy inference rules, or often-called inference laws, are frequently used. There are generalized modus ponens and generalized modus tollens.

Table 1.

Hedges and Corresponding Definition

<table>
<thead>
<tr>
<th>HEDGE</th>
<th>DEFINITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very A</td>
<td>( A^2 )</td>
</tr>
<tr>
<td>More or less A</td>
<td>( \frac{1}{A^2} )</td>
</tr>
<tr>
<td>Plus A</td>
<td>( A^{1.25} )</td>
</tr>
<tr>
<td>Not A or ( \neg A )</td>
<td>( 1 - A )</td>
</tr>
</tbody>
</table>

The modus ponens is also called the direct reasoning or the law of assuming the antecedent. The modus tollens is often called the indirect reasoning or the law of contraposition. With fuzzy sets \( A, A^*, B, B^* \) and linguistic variables \( x, y \), the generalized modus ponens and modus tollens can be expressed as:

1. For generalized modus ponens

   Premise 1 \( \text{if } x \text{ is } A, \text{ then } y \text{ is } B \)

   Premise 2 \( x \text{ is } A^* \)

   Consequence \( y \text{ is } B^* \)
where $B^* = A^* \circ R$, and $R$ is the fuzzy relation from the fuzzy implication (If $A$ Then $B$), $(\circ)$ is the composition operator, and $A^*$ is a fuzzy set which might be a hedge listed in Table 1.

3. The generalized modus tollens

   Premise  \[ \text{if } x \text{ is } A, \text{ then } y \text{ is } B \]

   Premise $y$ is $B^*$

   Consequence $x$ is $A^*$

   where $A^* = R \circ B^*$

Two special cases here must be noted:

1. In generalized modus tollens, when $B^* = \neg B$ and $A^* = \neg A$, where the symbol ($\neg$) means not, then the generalized modus tollens can be thought of as a backward goal-driven inference which is commonly used in expert systems. For a generalized modus ponens and with $A^* = A$ and $B^* = B$, this case is similar to a forward data-driven inference, which is widely used in the area of fuzzy logic-based control, where the consequence of a rule is not used as an antecedent of another. In this dissertation, only the second case is assumed in all of the fuzzy logic control models developed, i.e., $A^* = A$ and $B^* = B$ are assumed.

Fuzzy Logic Models

All of the fuzzy logic control systems have a knowledge base in their inference engines that constitute the brain of the fuzzy logic controller. In this section, a brief introduction is given to the knowledge-based fuzzy control system.
Fuzzy Knowledge Base

A fuzzy knowledge base usually consists of a number of fuzzy rules. In most engineering control applications, the fuzzy rules are expressed as If-Then, i.e.,

\[ \text{If } x \text{ is } A, \text{ Then } y \text{ is } B \]

This form of expressing the rules has the following advantages:

1. The human expert’s knowledge and experience can be expressed easily.
2. It provides an easy way to construct and program the fuzzy rules.
3. It cuts the cost of the design and provides good fuzzy inference efficiency.

There are several sentence connectives, such as “\text{AND},” “\text{OR},” and “\text{ALSO},” which are usually allowed. The connectives “\text{AND}” and “\text{OR}” are frequently used in the If part (antecedent part) of the fuzzy rule, while the connective “\text{ALSO}” is often used in the Then part (consequent part) of the fuzzy rules. In fuzzy logic in general, the connective “\text{AND}” is interpreted as an intersection operator, and “\text{OR}” as a union operator, while the connective “\text{ALSO}” indicates the presence of multiple outputs in the fuzzy rules. For example, a general rule can be stated as:

\[ R_k: \text{ If } u_1 \text{ is } A_{k1} \text{ OR } u_2 \text{ is } A_{k2} \text{ AND } u_3 \text{ is } A_{k3}, \text{ Then } v_1 \text{ is } B_{k1} \text{ ALSO } v_2 \text{ is } B_{k2} \]

This rule shows that there are two outputs in the consequent part of the rule. The firing strength of the rule or the degree of membership, as it is often referred to in the literature, which is calculated from the antecedent part, is applied twice in the computation of \( v_1 \) and \( v_2 \). In calculating the fire strength of the antecedent part, several operations are involved, including the intersection of \( u_2 \) and \( u_3 \) due to “\text{AND},”
and the union of $u_1$ and the computed intersection of $u_2$ and $u_3$ due to "OR." Thus, the above rule can be decomposed into two equivalent rules as:

If $u_1$ is $A_{k1}$ OR $u_2$ is $A_{k2}$ AND $u_3$ is $A_{k3}$, Then $v_1$ is $B_{k1}$

If $u_1$ is $A_{k1}$ OR $u_2$ is $A_{k2}$ AND $u_3$ is $A_{k3}$, Then $v_2$ is $B_{k2}$

which also can be decomposed into four rules:

If $u_1$ is $A_{k1}$ AND $u_3$ is $A_{k3}$, Then $v_1$ is $B_{k1}$

If $u_2$ is $A_{k2}$ AND $u_3$ is $A_{k3}$, Then $v_1$ is $B_{k1}$

If $u_1$ is $A_{k1}$ AND $u_3$ is $A_{k3}$, Then $v_2$ is $B_{k2}$

If $u_2$ is $A_{k2}$ AND $u_3$ is $A_{k3}$, Then $v_2$ is $B_{k2}$

In general, a multi-inputs multi-outputs (MIMO) control system can be decomposed into numbers of multi-inputs single output (MISO) control systems by utilizing the decomposition characteristic of the fuzzy control rule-base. For example, a 4-inputs and 2-outputs controller can be converted to a 2-inputs and 1-output controller. The only difference between the two controllers is in the number of the fuzzy control rules in the rule-base of each controller. To calculate the number of rules in the rule-base of a 2-inputs-1-output fuzzy controller resulted from the decomposition of MIMO fuzzy controller, one can use a simple mathematical formula developed in this dissertation and expressed mathematically as:

$$R_n = N_i^{N_o} - N_i$$

(2.30)

where $R_n$, $N_i$, and $N_o$ are the number of 2-inputs 1-output rules, the number of inputs, and the number of outputs, respectively.
Each rule in the fuzzy knowledge base corresponds to a fuzzy relation. For the fuzzy rule-base, each rule has a relation corresponding to rule \( R \). For the \( k \)th rule, there exists a relation denoted \( R_k \). The overall relation \( R \) of the fuzzy rule base can be obtained by performing a union operation on \( R_1, R_2, \ldots, R_n \) as 

\[ R = \bigcup_{i=1}^{n} R_i \]

Various approaches can be taken in determining the relation corresponding to a particular fuzzy rule. Fuzzy implication functions defined for a fuzzy rule in the form of If \( x \) is \( A \), Then \( y \) is \( B \), where \( A \in U \), \( x \in U \), \( B \in V \), and \( y \in V \) are:

1. Minimum rule, which can be expressed analytically as:

\[ R_c = A \times B = \int_{u \in U \times V} \mu_A(u) \land \mu_B(u) \]

(2.31)

2. Product rule, which can be expressed analytically as:

\[ R_p = A \times B = \int_{u \in U \times V} \mu_A(u) \ast \mu_B(u) \]

(2.32)

3. Min-Max rule, which can be expressed analytically as:

\[ R_m = (A \times B) \cup (\neg A \times V) = \int_{u \in U \times V} \frac{\mu_A(u) \land \mu_B(u)}{(u, v)} \lor (1 - \mu_A(u)) \]

(2.33)

4. Zadeh method, which can be expressed analytically as:

\[ R_Z = (A \times B) \Theta (\neg A \times V) = \int_{u \in U \times V} \frac{1 \land (1 - \mu_A(u) + \mu_B(u))}{(u, v)} \]

(2.34)
4. Boolean, which can be expressed analytically as:

$$R_c = \int_{\mathbb{R}^n} \frac{\mu_B(u) \lor (1 - \mu_A(u))}{(u, v)}$$  \hspace{1cm} (2.35)

where $\times$ denotes the Cartesian cross product and $(-)$ is the negation symbol.

The availability of different fuzzy implication functions provides flexibility in choosing the fuzzy implication method for a particular situation. In Chapter VII, the effects of choosing different implication methods, connectives t-norms and t-conorms, fuzzification, and defuzzification methods on the performance of the fuzzy system are investigated.

For a MISO control system with $N$ rules, the fuzzy relation for the $k$th rule $R_k$ can be expressed as: $R_k : A_k \rightarrow B_k$. The intersection of the antecedents, $A_{k1}$ and $A_{k2}$ and $\ldots \ldots \ldots \text{and } A_{ki}$ and $\ldots \ldots \text{and } A_{kn}$ is interpreted in two ways, namely, point-valued intersection or interval-valued intersection. The relation $R_k$, therefore, depends on which method is used. The point-valued intersection of the antecedents generates a point-valued fuzzy set denoted by:

$$A_k = \bigcap_{i=1}^{n} A_{ki}$$  \hspace{1cm} (2.36)

The interval-valued intersection of the antecedents generates an interval-valued fuzzy set given by:

$$A_k = [L^k(n), U^k(n)]$$  \hspace{1cm} (2.37)

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
where \( L^k(n) \) is the lower bound defined by \((A \cap B)\), and \( U^k(n) \) is the upper bound defined by \((A \cup B) \cap (A \cup B^*) \cap (A^* \cap B)\). Once \( A_k \) is found, the fuzzy relation \( R_k \) can be established by any of the methods listed earlier. In the case of the MISO system, the antecedent of the \( k \)th rule forms a fuzzy set \( A_k = A_{k1} \times A_{k2} \times \ldots \times A_{kn} \) in the product space \( U_1 \times U_2 \times \ldots \times U_n \). The consequent of the \( k \)th rule is a single control \( B_k \) in the universe of discourse \( V \). The overall fuzzy relation, \( R \), of the fuzzy rule base with \( n \) rules is given by \( R = \bigcup_{i=1}^{n} R_i \), where \( R \) may take either point-valued fuzzy implication or interval-valued fuzzy implication form.

For measured input signals to the fuzzy knowledge base expressed by If-Then rules and denoted by \( x = (A_1^*, A_2^*, \ldots, A_n^*) \) and the inferred fuzzy control from the \( i \)th rule given by \( y = B_i^* \) \( (i = 1, 2, \ldots, n) \), then using generalize modus ponens inference, the fuzzy control \( B^* e V \) can be inferred as \( B^* = (A_1^*, A_2^*, \ldots, A_n^*) \circ R \), where \((\circ)\) is the compositional operator and \( R \) is the overall fuzzy relation. Using the point-valued compositional rule of inference, the fuzzy control \( B^* \) can be expressed as:

\[
B^* = \bigcup_{k=1}^{n} A^* \circ R_k
\]

(2.38)

\[
A^* = \bigcap_{k=1}^{n} A_k^*
\]

When Sup-Min compositional operator is used, the membership value of the inferred consequence is given by Equation (2.39):

\[
\mu_{B^*} = \bigcup_{k=1}^{n} \left[ \text{Sup} \left( \bigwedge_{k=1}^{n} \mu_{A_k^*} \right) \wedge \left( \bigwedge_{i=1}^{n} \mu_{A_{ki1}} \rightarrow \mu_{B_k} \right) \right]
\]

(2.39)
Once the implication function is selected for the fuzzy relation $R$, the membership value of the inferred consequence can be determined. If the Mamdani "Min" implication function is used, then the membership value of the inferred consequence is given by Equation (2.40):

$$\mu_{\mathbf{B}^*} = \bigcup_{k=1}^{n} \left[ \bigwedge_{i=1}^{n} (\mu_{\mathbf{A}_i} \land \mu_{\mathbf{A}_k}) \right]$$  \hspace{1cm} (2.40)

The composition rule of inference involves three basic operations as follows:

1. Identify the connectives "AND" and "OR" in the antecedent of the rule base, knowing that "AND" indicates intersection while "OR" indicates union, and intersection has higher priority and should be performed first.

2. Find the firing strength $\alpha_k$ for the $k$th rule.

3. Apply the composition operator to infer the control actions in the consequent of the rule base.

Graphically, the steps can be illustrated as shown in Figures 7 and 8 for two types of compositional operators, the Sup-Min operator and Sup-Product operator for the rules:

$$R_1: \text{If } u \text{ is } A_1 \text{ and } v \text{ is } B_1 \text{ Then } z \text{ is } C_1$$

$$R_2: \text{If } u \text{ is } A_2 \text{ and } v \text{ is } B_2 \text{ Then } z \text{ is } C_2$$

where $u = \mathbf{A}^*$ and $v = \mathbf{B}^*$ and $A^*$ and $B^*$ are fuzzy inputs to the controller.
Figure 7. Composition Using Sup-Min Operator.

Figure 8. Composition Using Sup-Product Operator.
Fuzzy Measurements

A fuzzy measure indicates the degree of fuzziness of a fuzzy set, the degree or quality of imprecision or vagueness intrinsic in a property, process, or concept. The measure of the fuzziness and its characteristic behavior within the domain of the process is the semantic attribute captured by a fuzzy set. Fuzziness deals with the natural imprecision associated with everyday events. When one measures a speed against the idea of fast, then one is dealing with the imprecision concept. The fuzziness measurements can be presented by $f: p(x) \rightarrow \mathbb{R}$, where $p(x)$ denotes the set of all fuzzy subsets of $X$ and the function $f$ assigns a value $f(A)$ to each fuzzy subset $A$ of $X$ that characterizes the degree of fuzziness of $A$. The function $f$ must satisfy the following conditions in order to qualify as fuzziness measure:

1. $f(A) = 0$ if $A$ is crisp set
2. $f(A) \leq f(B)$ if $A < B$, where $\mu_A(u) \leq \mu_B(u)$ for $\mu_B(u) = 0.5$, and $\mu_A(u) \geq \mu_B(u)$ for $\mu_B(u) \geq 0.5$ for all $u \in X$.  
3. $f(A)$ attains its maximum if and only if $\mu_A(u) = 0.5$ for all $u \in X$.

The index of fuzziness is defined in terms of a metric distance between a fuzzy set $A$ and a crisp set $B$ with conditions that $\mu_C(u) = 0$ if $\mu_A(u) \leq 0.5$, and $\mu_C(u) = 1$ and $\mu_A(u) \geq 0.5$. The fuzziness measure function for the three forms of measure is given by:

1. Hamming distance which is given by:
2. Euclidean distance which is given by:

\[ f(A) = \sum |\mu_A(u) - \mu_C(u)| \]  \hspace{1cm} (2.41)

3. Minkowski distance which is given by:

\[ f(A) = \left( \sum (\mu_A(u) - \mu_C(u))^\delta \right)^{\frac{1}{\delta}} \]  \hspace{1cm} (2.43)

where \( \delta \in [0, \infty) \).

Another measure that is of importance in fuzzy mathematics is the fuzzy entropy, which is defined for a fuzzy set \( A \) by the function \( f(A) \) as shown in Equation (2.44):

\[ f(A) = -\sum \mu_A(u) \log \mu_A(u) + [1 - \mu_A(u)] \log[1 - \mu_A(u)] \]  \hspace{1cm} (2.44)

Fuzzy Clustering

Fuzzy clustering is useful in fuzzy modeling, that is, in the identification of the fuzzy rules needed to describe a "black box" system, on the basis of observed inputs and outputs data pairs. One of the most common fuzzy clustering methods is the \( c \)-means clustering. In this section, a brief introduction of fuzzy clustering is given. In Chapter III, the use of clustering in the design of a fuzzy control system is discussed. For a set of input output data given by: \( V = \{ v_1, v_2, v_3, \ldots, v_n \} \), the \( c \)-mean clustering can then be used to classify the data into \( c \)-fuzzy clusters. A membership
value that describes the degree at which each datum belongs to the specific fuzzy
cluster can be expressed by \( \mu_{ik}(v_k) \in [0, 1] \), \( 1 \leq i \leq c, \ 1 \leq k \leq N \), where \( \mu_{ik}(v_k) \) is the
degree at which the \( k \)-th datum, \( y_k \), belongs to the \( i \)-th cluster.

The fuzzy matrix for \( c \)-mean fuzzy cluster can be expressed as:

\[
\boldsymbol{\mu}_f(c) = \begin{bmatrix}
\mu_{11}(v_1) & \ldots & \mu_{1N}(v_n) \\
\vdots & \ddots & \vdots \\
\mu_{c1}(v_1) & \ldots & \mu_{cN}(v_n)
\end{bmatrix}
\]  
(2.45)

with condition that \( \sum_{i=1}^{c} \mu_{ik}(v_k) = 1 \). The performance index \( p(c) \) for the partition
matrix is expressed as:

\[
p(c) = \sum_{k=u=1}^{N} \sum_{i=1}^{c} \left( \mu_{ik} \right)^w |v_k - v_{fi}|^2
\]  
(2.46)

where \( v_{fi} \) is the center of the \( i \)-th fuzzy cluster vector given by:

\[
v_{fi} = \frac{\sum_{k=1}^{N} \left( \mu_{ik} \right)^w v_k}{\sum_{k=1}^{N} \left( \mu_{ik} \right)^w}
\]  
(2.47)

where \( w \) is a weight on the membership values. The optimal partition matrix is
obtained when \( p(c) \) is at minimum. The optimal number of clusters is determined by
its performance index \( p_{nc}(c) \), which is given by Equation (2.48):

\[
p_{nc}(c) = \sum_{k=l=1}^{N} \sum_{i=1}^{c} \left( \mu_{ik} \right)^w \left[ |v_k - v_{fi}|^2 - |v_{fi} - v_p|^2 \right]
\]  
(2.48)
where the average of all data points $v_p$ is given by Equation (2.49):

$$v_p = \frac{1}{N} \sum_{k=1}^{N} v_k$$  \hspace{1cm} (2.49)

When $p_{nc}(c)$ is minimal, the optimal number of clusters is then obtained.

The following steps are usually followed in fuzzy clustering techniques:

1. Obtain the partition matrix $\mu_f(c)$ by finding $\mu'_k$ as in Equation (2.50a):

$$\mu_f(c) = \begin{bmatrix}
\mu'_1(v_1) & \ldots & \mu'_N(v_n) \\
\vdots & \ddots & \vdots \\
\mu'_1(v_1) & \ldots & \mu'_N(v_n)
\end{bmatrix}$$  \hspace{1cm} (2.50a)

2. Set the initial cluster number $c$ to 2 at $t = 0$ and calculate the center of gravity vector for each cluster for the partition matrix using Equation (2.50b):

$$v'_f = \frac{\sum_{k=1}^{N} (\mu'_{ik}(v_k))^w v_k}{\sum_{k=1}^{N} (\mu'_{ik}(v_k))^w}$$  \hspace{1cm} (2.50b)

3. Modify each membership value according to Equations (2.50c) and (2.50d):

$$\mu'^{-1}_{ik}(v_k) = \begin{cases} 
\sum_{g=1}^{c} \left( \frac{v_k - v'_{fg}}{|v_k - v'_{fg}|^2} \right) \frac{1}{w-1} & \text{if } v_k \neq v'_f \\
0 & \text{otherwise}
\end{cases}$$  \hspace{1cm} (2.50c)
Then the partition matrix is obtained as:

\[
\mu_{ik}^{t+1}(v_k) = \begin{cases} 
1 & \text{for } i = g \\
0 & \text{for } i \neq g 
\end{cases} \quad \text{if} \quad v_k = v_f^t
\] (2.50d)

Then the partition matrix is obtained as:

\[
\mu_f^{t+1}(c) = \begin{bmatrix} 
\mu_1^{t+1}(v_1) & \ldots & \mu_{iN}^{t+1}(v_n) \\
\vdots & \ddots & \vdots \\
\mu_{c1}^{t+1}(v_1) & \ldots & \mu_{cN}^{t+1}(v_n) 
\end{bmatrix}
\] (2.50e)

4. Check for the convergence of the partition matrix using Equation (2.50f):

\[
\left| \mu_f^{t+1}(c) - \mu_f^t(c) \right| < \epsilon
\] (2.50f)

If convergence is not achieved, then the steps should be repeated.

5. If \( p_{nc}(c) \) is minimum, then clustering process is terminated; otherwise, the number of clusters should be incremented, i.e., \( c = c + 1 \), and the procedure should be repeated with performance index given by Equation (2.50g):

\[
p_{nc}(c) = \sum_{k=1}^{N} \sum_{l=1}^{c} \mu_{ik} \left[ |v_k - v_f^t| - |v_f^t - v_l| \right]^2
\] (2.50g)

Extension Principle

The process of extending fuzziness in an input set to an output set through mapping of the form \( f: A \rightarrow B \) is accomplished using the extension principle. This means that the image of fuzzy set \( A \) on universe of discourse \( X \) under mapping \( f \)
yields a fuzzy set \( B \). Mathematically, this can be expressed as \( B = f(A) \) and the membership function is given as:

\[
\mu_B(y) = \bigvee_{f(x)=y} \mu_A(x) \tag{2.51}
\]

where the mapping is of the form \( f(x) = y \).

Fuzzy vector is a vector containing fuzzy membership values. For a fuzzy set \( A \) that is defined on \( n \) elements in \( x \) \((x_1, x_2, \ldots, x_n)\), and a fuzzy set \( B \) defined on \( m \) elements in \( y \) \((y_1, y_2, \ldots, y_m)\), an array of membership functions for each of the fuzzy sets \( A \) and \( B \) can be written in fuzzy vector form as:

\[
a = \{a_1, a_2, \ldots, a_n\} = \{\mu_A(x_1), \mu_A(x_2), \ldots, \mu_A(x_n)\} \tag{2.52a}
\]

\[
b = \{b_1, b_2, \ldots, b_n\} = \{\mu_B(y_1), \mu_B(y_2), \ldots, \mu_B(y_n)\} \tag{2.52b}
\]

The image of fuzzy set \( A \) given by fuzzy set \( B \) can be found using fuzzy composition operation \( B = A \circ R \), which, when using the fuzzy vector form, can be expressed as \( b = a \circ R \), where \( R \) is an \( n \times m \) fuzzy relation matrix.

For general form and using the Cartesian product of many universes of discourse, the mapping for input sets can be defined by \( B = f(A_1, A_2, \ldots, A_m) \), where \( A_i \) \((i = 1, \ldots, n)\) are fuzzy sets defined on \( X_i \) universes of discourse. The membership function of the image \( B \) is obtained using Zadeh’s extension-principle as shown in Equation (2.53):
For a mapping to be one-to-one, the mapping function should map each
element of the universe of discourse to its image on the image universe of discourse.

For example, to map an element \( u \in U \) onto elements \( v \in V \) through a function \( f \), such
that the mapping is described by \( f: u \rightarrow v \), then a fuzzy set \( A \) on \( U \) is expressed as:

\[
A = \left\{ \frac{\mu_1}{u_1} + \frac{\mu_2}{u_2} + \cdots + \frac{\mu_n}{u_n} \right\}
\]

Using the extension principle, the fuzzy image \( f(A) = B \) is then expressed as:

\[
B = \left\{ \frac{\mu_1}{f(u_1)} + \frac{\mu_2}{f(u_2)} + \cdots + \frac{\mu_n}{f(u_n)} \right\}
\]

For cases where \( f \) maps products of elements from two universes \( U_1 \) and \( U_2 \), to
another universe \( V \), with the fuzzy set \( A \) is defined on the Cartesian space \( U_1 \times U_2 \).
then

\[
f(A) = \left( \sum \frac{\text{Min}\{\mu_1(i), \mu_j(j)\}}{f(i,j)} \right)
\]

where \( i \in U_1 \) and \( j \in U_2 \).

The stated methods are only suitable for discretized mappings. Vertex method
can be used to simplify manipulations of the extension principle for continuous-
valued fuzzy variables. The method is based on a combination of \( \beta \)-cut concept and standard interval analysis. It works as follows: Any continuous membership function can be presented by a continuous sweep of \( \beta \)-cut intervals from \( \beta = 0 \) to \( \beta = 1 \). For single-input mapping given by \( f(x) = y \) or \( B = f(A) \), and \( A \) is decomposed into a series of \( \beta \)-cut intervals \( I_\beta \), then \( B_\beta \) can be expressed as:

\[
B_\beta = f(I_\beta) = \{ \text{Min} \{f(a), f(b)\}, \text{Max} \{f(a), f(b)\} \}
\]

(2.57a)

where \( B_\beta \) is the interval representing \( B \) at particular value of \( \beta \). When the mapping is given by \( n \) inputs, i.e., \( y = f(x_1, x_2, \ldots, x_m) \), then the input space can be represented by an \( n \)-dimensional Cartesian region. Each of the input variables can be described by the interval \( I_{\beta i} \) at a specific \( \beta \)-cut, where \( a_i \leq I_{\beta i} \leq b_i, \ i = 1, 2, \ldots, n \).

In the case of the continuous \( f \) the value of the interval function for a particular \( \beta \)-cut is given by:

\[
B_\beta = f(I_{\beta 1}, I_{\beta 2}, \ldots, I_{\beta n}) = \left[ \text{Min} \left( f(c_j) \right), \text{Max} \left( f(c_j) \right) \right]\]

(2.57b)

where \( j = 1, 2, \ldots, N \), and \( c_j \) is the coordinate of the \( j \)th vertex representing the \( n \)-dimensional Cartesian region.

**Aggregation of Fuzzy Rules**

Most fuzzy rule-based systems involve more than one rule. The process of obtaining the overall consequent from individual consequents contributed by each rule is known as the aggregation process. In determining an aggregation strategy, two
extreme cases exist:

1. Conjunctive system of rules. In this system the rules are connected by “AND” connectives. In this case, the aggregated output, $y$, is found by the fuzzy intersection of all individual rule consequents, $y_i$, where $i = 1, 2, \ldots, r$ as $y = y_1 \text{ and } y_2 \text{ and } y_3 \text{ and } \ldots \text{ and } y_r$, or $y = y_1 \cap y_2 \cap y_3 \cap \ldots \cap y_r$, which is defined by the membership function

$$
\mu_y(y) = \min\{\mu_{y_1}(y), \mu_{y_2}(y), \ldots, \mu_{y_r}(y)\} \quad (2.58)
$$

2. Disjunctive system of rules. In this system the rules are connected by the “OR” connectives. The aggregated output is found by the fuzzy union of all individual rules contributions, as $y = y_1 \cup y_2 \cup y_3 \cup \ldots \cup y_r$, which is defined by the membership function as:

$$
\mu_y(y) = \max\{\mu_{y_1}(y), \mu_{y_2}(y), \ldots, \mu_{y_r}(y)\} \quad (2.59)
$$

The general rules format for two inputs one output case can be expressed as:

If $x_1$ is $A_{1k}$ and $x_2$ is $A_{2k}$ Then $y_k$ is $B_k$

where $A_{1k}$ and $A_{2k}$ are fuzzy sets representing the $k$th-antecedent pairs, and $B_k$ are the fuzzy sets representing the $k$th-consequent. The cases that can be encountered in a fuzzy system are:

1. Crisp inputs and fuzzy outputs with Max-min inference method. In this case, $x_1$ and $x_2$ are crisp values, which could be represented by delta function with the
membership for inputs \( x_1 \) and \( x_2 \) given by:

\[
\mu(x_1) = \delta(x_1 - u(i)) = \begin{cases} 
1 & x_1 = u(i) \\
0 & \text{otherwise} 
\end{cases} 
\]  
(2.61a)

\[
\mu(x_2) = \delta(x_2 - u(j)) = \begin{cases} 
1 & x_2 = u(j) \\
0 & \text{otherwise} 
\end{cases} 
\]  
(2.61b)

where \( u(i), u(j) \) denotes the input. The aggregated output for the \( k \)-rules is given by:

\[
\mu_B(k)(y) = \max_k \{\min \{\mu_{ik}(u(i)), \mu_{kj}(u(j))\}\} 
\]  
(2.62)

2. This is the same as case 1, except the Max-product inference method is used. In this case the aggregated output is expressed as:

\[
\mu_B(k)(y) = \max_k \{\mu_{ik}(u(i)) \cdot \mu_{kj}(u(j))\} 
\]  
(2.63)

3. The inputs to the system are represented by fuzzy sets and the Max-min inference method is used. Here the input \( u(i) \) and \( u(j) \) are fuzzy variables described by fuzzy membership functions. The aggregated output using a Mamdani implication is expressed as:

\[
\mu_B(k)(y) = \max_k \{\min [\max [\mu_{ik}(x) \wedge \mu(x_1)], \max [\mu_{kj}(x) \wedge \mu(x_2)]]\} 
\]  
(2.64)

4. This is the same as case 3, except the Max-product inference method is used. The aggregated output is then given by:
\[
\mu_{B_k}(y) = \text{Max}_k \{ \text{Max}_1 \{ \mu_{i_k}(x) \wedge \mu(x_1) \} \times \text{Max}_1 \{ \mu_{i_{2k}}(x) \wedge \mu(x_2) \} \} \tag{2.65}
\]

In a case where an input variable \( x \) results in a series of fuzzy output \( y_k \), then the mapping will depend on which fuzzy relation, \( R_k \), is used, for example,

\( R_1: y_1 = x \circ R_1 \), \( R_2: y_2 = x \circ R_2 \), …, \( R_n: y_n = x \circ R_n \), then a fuzzy relational equation \( R_k: y_k = x \circ R_k \), where \( y_k \) is the output of the fuzzy system contributed by the \( k \)th rule, and whose membership function \( \mu_{y_k}(y) \) can be written as single variable fuzzy relations of dimension \((l \times n \times m)\).

For a fuzzy system described by a system of conjunctive rules, one can decompose the rules into a single aggregated fuzzy relational equation for each input, \( x \), as follows:

\[
y = (x \circ R_1) \cap (x \circ R_2) \cap \ldots \cap (x \circ R_n) \tag{2.66}
\]

which also can be written as:

\[
y = x \circ R \tag{2.67a}
\]

\[
R = (R_1 \cap R_2 \cap R_3 \cap \ldots \cap R_n) \tag{2.67b}
\]

The aggregated fuzzy relation \( R \) is called the fuzzy system transfer relation for a single input, \( x \). For a system with \( n \) fuzzy inputs, \( x_i \), and a single output, \( y \), the fuzzy relational equation for disjunctive rules can be expressed as:

\[
y = x_1 \circ x_2 \circ \ldots \circ x \circ R \tag{2.68a}
\]
which also can be written as:

\[ y = x \circ R \]  

(2.68b)

where

\[ R = R_1 \cup R_2 \cup R_3 \cup \ldots \cup R_n \]  

(2.68c)

A general nonlinear system, which is composed of \( n \) inputs and \( m \) outputs, can be presented by fuzzy relational equations in the form \( R_x: \text{if } x \text{ is } A_i \text{ then } y \text{ is } B_j \). The rules could be connected logically (\( i=1,2,\ldots,r \)) by any of the "AND," "OR," or "ELSE" linguistic connectives, and the variables \( x \) and \( y \) are the input and output vectors, respectively.

Defuzzification Methods

Defuzzification is the conversion of a fuzzy quantity to a precise crisp quantity, just as fuzzification is the conversion of crisp quantity to a fuzzy one. The output of a fuzzy process can be the logical union of two or more fuzzy membership functions defined on the universe of discourse of the output variable. For example, suppose a fuzzy output is comprised of two parts: the first part is a fuzzy set represented by a Gaussian membership function and denoted by \( A_1 \), and the second is another fuzzy set represented by a triangular membership function and denoted by \( A_2 \). The union of these two membership functions, i.e., \( A = A_1 \cup A_2 \), involves the Max-
operator, which is the outer envelope of the two shapes. A general fuzzy output process can involve many output parts, and the membership function representing each part of the output can have different shapes. This can be expressed in general term as \( A_k = \bigcup_{i=1}^{k} A_i \).

The most popular defuzzification methods that are found in literature are:

1. Weighted Average (WA) Method: This method is valid for symmetrical output membership functions only. It is given by:

\[
z^* = \frac{\sum \bar{z} \mu_A(\bar{z})}{\sum \mu_A(\bar{z})}
\]

where \( \sum \) is the algebraic sum. This method is shown graphically in Figure 9.

![Figure 9. Weighted Average (WA) Defuzzification Method.](image)

The defuzzified value \( z^* \) shown in Figure 9 is given by Equation (2.70):

\[
z^* = \frac{\mu_A(a) a + \mu_A(b) b}{\mu_A(a) + \mu_A(b)} \tag{2.70}
\]
2. Centroid Method: This procedure is also called Center of Area (COA) or Center of Gravity (COG) method. It is expressed mathematically by Equation (2.71):

$$z^* = \frac{\int z \mu_A(z) dz}{\int \mu_A(z) dz}$$  \hspace{1cm} (2.71)

where $\int$ is the regular integration. The method is shown graphically in Figure 10.

![Figure 10. Center of Area (COA)/Center of Gravity (COG) Defuzzification Method.](image)

3. Center of Sums: This method is the fastest defuzzification method. It involves the algebraic sum of individual output fuzzy sets instead of their union. The drawback to this method is that the intersecting areas are added twice. This method is similar to the weighted average method, except in the center of sums (COS) method, the weights are the areas of the respective membership functions, not the weights of individual membership values as is the case in the weighted average method. It is shown graphically in Figure 11. The defuzzified value is given by:

$$z^* = \frac{\int \sum_{k=1}^{n} \mu_{A_k}(z) dz}{\int \sum_{k=1}^{n} \mu_{A_k}(z) dz}$$  \hspace{1cm} (2.72)

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
4. Center of Largest Area (CLA): If the output fuzzy set has at least two convex sub-regions, then the center of gravity can be used to obtain the defuzzified value $z^*$ as shown in Equation (2.73):

$$z^* = \frac{\int z \mu_{A_m}(z)dz}{\int \mu_{A_m}(z)dz}$$  \hspace{1cm} (2.73)

where $A_m$ is the convex subregion with the largest area as shown in Figure 12.

5. First of Maxima (FM) and Last of Maxima (LM): This method uses the overall output or union of all individual output fuzzy sets $A_k$ to find the smallest value
of the domain with maximized membership degree in $A_k$. The defuzzified value $z^*$ for
FM and LM defuzzification methods is given by Equations (2.74) and (2.75),
respectively.

\[
\begin{align*}
  z^* &= \inf_{z \in Z} \{ z \in Z / \mu_{A_k}(z) = \text{hgt}(A_k) \} \\
  \text{hgt}(A_k) &= \sup_{z \in Z} \mu_{A_k}(z) 
\end{align*}
\tag{2.74}
\]

\[
  z^* = \sup_{z \in Z} \{ z \in Z / \mu_{A_k}(z) = \text{hgt}(A_k) \} \tag{2.75}
\]

where $\text{hgt}(A_k)$ is the largest height in the union. Figure 13 depicts graphically the LM
defuzzification method.

Figure 13. Last of Maxima (LM) Defuzzification Method.

6. Max-Membership or Height Method: It is limited to the peaked output
functions as shown in Figure 14. Max-Membership (MM) defuzzification method is
given by:

\[
\mu_A(z^*) \geq \mu_A(z) \quad \forall z \in Z
\tag{2.76}
\]
7. Mean-Max Membership: This method is also called Middle of Maxima (MOM). The defuzzified value \( z^* \) shown in Figure 15, is obtained using MOM defuzzification method as given by Equation (2.77):

\[
    z^* = \frac{a + b}{2}
\]

(2.77)

Other defuzzification methods either can be constructed provided that certain criteria are satisfied. Further discussion of defuzzification methods is presented in Chapter VII when the effect of certain choices of the defuzzification method is considered in the context of parameters selection for the fuzzy logic controller. Three
more recently developed defuzzification methods that have the potential to be
generalized defuzzification methods are investigated in Chapter VII.

In the next section, the types of more frequently used fuzzy models are
discussed. Their advantages and in what situations one model can be chosen over the
other model are stated.

Types of Fuzzy Models

Introduction

There are three types of fuzzy models that are currently in use, the first of
which was introduced by Mamdani (1975). The second was developed by Takagi-
Sugeno-Kang (TSK) in 1984. The third was developed by Kosko (1996) and it is
called the standard additive model (SAM). In the following section, a brief
introduction to each of the models mentioned above is given.

Mamdani Fuzzy Control Model

The Mamdani model is a rule-based model. It is one of the most widely used
fuzzy models. It constitutes the base for the TSK and SAM models, because they are
derived from it and, in fact, the antecedent part is the same in all three models. The
general format of the Mamdani model can be expressed in linguistic rules that
describe a mapping from the input universe of discourse to the output (i.e., \( U \rightarrow V \)) for
multi-input-single-output case (MISO) as:
\[ R_j: \text{If } x_1 \text{ is } A_{j1} \text{ and } x_2 \text{ is } A_{j2} \text{ and \ldots \ and } x_r \text{ is } A_{jr} \text{ Then } y \text{ is } C_j \] (2.78)

where \((x_1, x_2, \ldots, x_r)\) are the input variables and \(y\) is the output variable. The linguistic values \(A_{jr}\) and \(C_j\) represent the fuzzy sets for the inputs and output variables, respectively. The output of the Mamdani fuzzy model for the rule \(R_j\), given a set of input such as \(A_{j1}^*, A_{j2}^*, \ldots, A_{jr}^*\) of the form \(x_1 = A_{j1}^*, x_2 = A_{j2}^*, \ldots, x_r = A_{jr}^*\), is given by:

\[
\mu_{C_j}(y) = (\mu_{j1} \land \mu_{j2} \land \mu_{j3} \land \ldots \land \mu_{jr}) \land \mu_{C_j}(y) 
\] (2.79)

where \(\mu\) is the firing strength of the degree of membership function of the rule \(R_j\), and \(\mu_{jr}\) is the degree of membership between \(x_r\) and \(R_j\)'s condition about \(x_r\), i.e.,

\[
\mu_{jr} = \sup_{x_r} (\mu_{A_{jr}}(x_r) \land \mu_{A_{jr}}(x_r))
\] (2.80)

The Mamdani model aggregated output of all of the rules combined using the maximum operator is given by:

\[
\mu_C(y) = \max(\mu_{c_1}(y), \mu_{c_2}(y), \ldots, \mu_{c_r}(y))
\] (2.81)

where \(C\) is the aggregated output fuzzy set that needs to be defuzzified to get the crisp fuzzy logic controller output.

**Takagi-Sugeno-Kang (TSK) Fuzzy Model**

Takagi and Sugeno proposed this model in 1981 in the hope of reducing the
number of rules required by the Mamdani model. The main difference between this
model and Mamdani model is in the consequent part of the If-Then rules. Instead of
having fuzzy sets in the consequent part of the Mamdani model, the TSK model
expresses the consequent part as a linear function of the linguistic variables. The
general format of TSK model is given by:

\[ \text{If } x_1 \text{ is } A_{i1} \text{ and } x_2 \text{ is } A_{i2} \text{ and ... and } x_r \text{ is } A_{ir} \text{ Then } y = f(x_1, x_2, x_3, ..., x_r) = b_{o0} + b_{i1}x_1 + ... + b_{ir}x_r \]  

(2.82)

where \( f \) is a linear model and \( b_{ij} (j = 0, 1, 2, ..., r) \) are constants. For a two-input-
one-output case, the TSK model can be expressed as:

\[ \text{If } x_1 \text{ is } A_1 \text{ and } x_2 \text{ is } A_2 \text{ Then } y = b_o + b_1x_1 - b_2x_2 \]  

(2.83)

The output of a TSK fuzzy model, presuming that the Center Of Area (COA)
defuzzification method is used, is given by:

\[ y = \frac{\sum_{i=1}^{l} \mu_i (b_{i0} + b_{i1}x_1 + ... + b_{ir}x_r)}{\sum_{i=1}^{l} \mu_i} \]  

(2.84)

where \( \mu_i \) is the firing strength of the matching degree of rule \( R_i \), and it is computed
using either the minimum operator, product operator, or any other \( t \)-norms operator.

In the case of the Minimum operator, the firing strength is given by:

\[ \mu_i = \text{Min}(\mu_{A_{i1}}(a_1), \mu_{A_{i2}}(a_2), ..., \mu_{A_{ir}}(a_r)) \]  

(2.85)
where $a_1, a_2, \ldots, a_r$ are the crisp inputs to $x_i (i = 1, 2, \ldots, r)$, respectively. The generalized form of Equation (2.85) in terms of $t$-norms operators is expressed as:

$$
\mu_i = \mu_{A_{i1}}(a_1) \mu_{A_{i2}}(a_2) \cdots \mu_{A_{ir}}(a_r)
$$

where $t$ represents any of the $t$-norms operators.

**Standard Additive Fuzzy Model (SAM)**

The standard additive fuzzy model (SAM) is similar to the Mamdani fuzzy model. Kosko (1996) originally introduced the SAM model. It is a relatively new model and was derived from the Mamdani model with some modifications. The differences between the two models can be summarized as follows:

1. The SAM model accepts only crisp inputs to the fuzzy system similar to the TSK model, which is also, assumes crisp inputs to the system. The Mamdani model, on the other hand, accepts both crisp and fuzzy inputs to the fuzzy system.

2. The composition operator in SAM is assumed to be the product operator, while the Mamdani model works with any $t$-norms composition operator.

3. In the SAM model, the aggregation of the rules uses the addition operator, while the Mamdani model can accept the sum operator as well as other operators.

4. The defuzzification method used in SAM is assumed to be the COA method, while Mamdani defuzzification methods can be any defuzzification methods that were discussed in the previous section.

The output of the SAM fuzzy model for a general rule such as:
R_j: If \( x_1 \) is \( A_{j1} \) and \( x_2 \) is \( A_{j2} \) and \ldots and \( x_r \) is \( A_{jr} \) Then \( y \) is \( C_j \)
is given by:

\[
y = \frac{\sum_{i=1}^{n} [\mu_{A_{j1}}(a_1) \times \mu_{A_{j2}}(a_2) \times \ldots \times \mu_{A_{jr}}(a_r)] \times M_i \times G_i}{\sum_{i=1}^{n} [\mu_{A_{j1}}(a_1) \times \mu_{A_{j2}}(a_2) \times \ldots \times \mu_{A_{jr}}(a_r)] \times M_i}
\]

(2.87)

where \( M_i \) is the area under the \( i \)th rule and it is given by:

\[
M_i = \int \mu_{C_j}(y)dy \quad \text{continuous case}
\]

(2.88)

\[
M_i = \sum_{i=1}^{n} \mu_{C_j}(y) \quad \text{discrete case}
\]

and \( G_i \) is the COA of the \( C_i \). \( G_i \) for a continuous case is expressed as:

\[
G_i = \frac{\int y \times \mu_{C_j}(y)dy}{\int \mu_{C_j}(y)dy}
\]

(2.89)

and for a discrete case as:

\[
G_i = \frac{\sum_{i=1}^{n} \mu_{C_j}(y) \times y}{\sum_{i=1}^{n} \mu_{C_j}(y)}
\]

(2.90)

The three variations of fuzzy logic control models can be used for different applications, provided that the restrictions on each model are taking into consideration. The rule-base in each of the above models must satisfy certain
requirements for the fuzzy model to function properly. The rule-base consists of the whole set of fuzzy If-Then rules. The relationship among these rules and the rules as a whole must satisfy certain requirements. First, they must be complete, i.e., there exists at least one rule in the rule-base such that the membership function of an element in the universe of discourse does not equal 0. For example, for $x \in U$ with a membership $\mu_p(x_i)$, then $\mu_p(x_i) \neq 0$ for $j = 1, 2, \ldots, n$. The second requirement is that the set of If-Then rules must be consistent, meaning that there are no rules with the same If-part and different Then-part. The third requirement in a set of If-Then rules is that the set must be continuous, which means the nonexistence of neighboring rules whose Then-part fuzzy sets have empty intersections. This means that the input-output behavior of the fuzzy system should be smooth.

**Fuzzy Rule-Based Expert Systems**

Physical systems for single input-single output (SISO), in most cases, can be expressed mathematically as:

$$\frac{d^n y}{dt^n} = G(t, y(t), y(t), \ldots, \frac{d^{n-1} y(t)}{dt^{n-1}}, u(t))$$

\[ (2.91) \]

where $u(t)$ and $y(t)$ are the input and output functions, respectively. A simple transformation in the form of $\frac{d^{n-1} y}{dt^{n-1}} = x_n$ gives a system of $n$ first order differential equations given by $\frac{dx_{n-1}}{dt} = x_n$, i.e., $\dot{x}_1 = x_2, \dot{x}_2 = x_3$, etc. This transformation is
known in control as state space transformation and for

\[
\tilde{X}(t) = \left[ x_1(t), x_2(t), \ldots, x_n(t) \right]^T
\]  

(2.92)

\[
F(t, \tilde{X}(t), u(t)) = \left[ X_2, X_3, \ldots, X_n, G(t, X_1, \ldots, X_n, u) \right]^T
\]  

(2.93)

where \( \tilde{X}(t) \) is the system state vector expressed as a function of time. The state space presentation for the model in Equation (2.91) can then be written as:

\[
\frac{d\tilde{X}(t)}{dt} = F(t, \tilde{X}(t), u(t))
\]  

\[
y(t) = h\tilde{X}(t)
\]  

(2.94)

where \( h \) is a row matrix of the form \( h = [1, 0, 0, \ldots, 0] \). For multi-inputs and multi-outputs (MIMO), the general format of the state space representation is given by:

\[
\frac{d\tilde{X}(t)}{dt} = F(t, \tilde{X}(t), \tilde{u}(t))
\]  

\[
\tilde{y}(t) = W(t, \tilde{X}(t), \tilde{u}(t))
\]  

(2.95)

where \( \tilde{u}(t) \) and \( \tilde{y}(t) \) are vectors of the inputs and outputs, respectively. For a time-invariant system (TIS), Equation (2.95) can be expressed as:

\[
\frac{d\tilde{X}(t)}{dt} = F(\tilde{X}(t), u(t))
\]  

\[
\tilde{y}(t) = W(\tilde{X}(t), \tilde{u}(t))
\]  

(2.96)
If the system is linear time-invariant systems (LTIS), Equation (2.95) can be expressed as in Equation (2.97):

\[
\begin{align*}
\frac{d\tilde{x}(t)}{dt} &= A\tilde{x}(t) + B\tilde{u}(t) \\
\tilde{y}(t) &= C\tilde{x}(t) + D\tilde{u}(t)
\end{align*}
\] (2.97)

where \( A, B, C, \) and \( D \) are the system constant matrices that are found from the state space transformation.

A feedback control system is defined as:

\[
u(t) = W(t, \tilde{x}(t), \tilde{r}(t))
\] (2.98)

where \( u(t) \) is the input to the plant, \( r(t) \) is the reference input, \( \tilde{x}(t) \) is the state space vector, and \( W \) is the feedback control that stabilizes the system. The function \( W \) is defined as a nonlinear hypersurface in an \( n \)-dimensional space, which is also called control or decision surface. The control surface describes the dynamics of the controller. Fuzzy logic rule-based expert system uses a collection of fuzzy conditional statements derived from expert knowledge to approximate and contract the control surface. Fuzzy logic, in contrast to conventional control techniques, is a model-free system and can be considered as a universal nonlinear function approximators.

The conventional and fuzzy logic control systems have similar design stages when designing a complex physical system. The basic steps are:

1. Decomposition and decentralization of large-scale systems into a collection of decoupled subsystems must take place.
2. The plant nonlinear dynamics are linearized about a set of operating points.

3. The control variables, output, and state space variables are defined.

4. A state-feedback controller in the form of Proportional (P), Proportional plus Integral (PI), Proportional plus Derivative (PD), Proportional plus Integral plus Derivative (PID), Linear Quadratic Regulator (LQR) or other techniques is designed for each decoupled system.

5. Using expert available knowledge in the form of input-output observations data, linguistic, intuition, and analytical expression, the controller performance should be made as close as possible to the optimal performance.

6. Tuning and adjusting the controller’s parameters in order to compensate for the effects of uncertainties and variations due to unmodeled dynamics are accomplished using a supervisory controller.

Fuzzy logic controllers provide a means of converting a linguistic control strategy derived from expert knowledge into an automatic control strategy. The approach provides an effective means of capturing the approximate fuzzy nature of the way real world problems are normally described.

A typical fuzzy logic controller consists of the following: (a) a fuzzification unit, which maps a crisp fuzzy controller input into the fuzzy linguistic values using fuzzy reasoning mechanism; (b) a knowledge base, which consists of a collection of the expert control rule or knowledge needed to achieve the control goal; (c) a fuzzy reasoning mechanism or fuzzy inference engine, which performs fuzzy logic operations to infer the control action for the given fuzzy inputs; and (d) a
defuzzification unit, which converts the inferred fuzzy control action into the required crisp control value.

Fuzzy-logic-based process control can be best applied to systems that: (a) are heavily dependent on human or expert knowledge, intuition, and experience, and when the application of conventional control algorithms are either difficult or impossible; (b) use conventional control methods but are in need of human or expert involvement for adjustment and fine tuning; (c) are very sensitive and very complex; (d) are operated in open loop or manually; (e) are nonlinear and time-varying; and (f) require rapid prototyping and design, while conventional control methods are model-based, and a mathematical model is essential for the design of the control algorithms. Fuzzy logic, on the other hand, incorporates the expert's knowledge of the application domain explicitly and arrives at a decision along lines that simulate human reasoning. Fuzzy logic works in a broadly similar way to classical systems in that it employs sensed inputs as a basis for determining control outputs. The only difference is in the way it arrives at that determination. A fuzzy-logic-based approach can be used in combination with a classical approach. This will be explained in later sections when adaptive and supervisory control systems are discussed. The benefits of using fuzzy logic controllers include, but are not limited to, the following: (a) reliability and efficiency; (b) the ability to solve unsolvable control problems that use conventional controllers; (c) a more effective design work force (less time spent on design process); (d) consistent and improved quality of operation; (e) fast prototyping and
economical in terms of time and money; and (f) incorporation of human knowledge into the design process and control algorithms, as depicted in real-life situations.

This chapter serves as an introductory chapter to fuzzy mathematics and gives an idea of how these fuzzy systems are used in control. The fuzzy sets and their relation to membership functions is introduced. The inference engine of fuzzy systems that consists of fuzzification, rule-base, implication, aggregation, and defuzzification is also presented. The inference engine constitutes the brain of the fuzzy logic controller and gives an insight into how fuzzy system functions in real applications. Three types of fuzzy models, namely, Mamdani, TSK, and SAM are also discussed. These models, in one way or another can be found in any fuzzy control system. In Chapter III, the design of fuzzy systems using different approaches is discussed. Depending on what is available for the design engineer, Chapter III gives a good idea of how to go about designing fuzzy systems using experimental data, fuzzy clustering, intuitive knowledge, and knowledge gained from conventional control, such as PID and adaptive control. The purpose of this chapter and Chapter III is to present, in as complete a way as possible, an objective, theoretical, and practical foundation of fuzzy logic and its application to control problems.
CHAPTER III

FUZZY SYSTEM DESIGN

Introduction

Fuzzy systems are used to formulate human knowledge about a particular engineering problem. Two kinds of knowledge can be encountered in application situations: knowledge that can be expressed in words, and knowledge that cannot be expressed in words. If the knowledge can be expressed in words, then it can be expressed in terms of fuzzy If-Then rules and can be put into fuzzy systems. In the case where knowledge cannot be explicitly expressed in words, a set of input-output data pairs can be collected, and a fuzzy system can be constructed from these data pairs. In this section, some of the methods used to design fuzzy systems using experimental data are explored, because in many practical situations, only input and output data pairs are provided.

Design of Fuzzy Systems Using Only Experimental Data

In certain situations, when little knowledge is available to design a control system, then designing a fuzzy control system can be a difficult task. Sometimes, however, there exist experimental data that describe the relationship between a set of inputs and outputs. If such data exist, then designing a fuzzy control system using the
experimental data becomes an easy task. The following section gives an outline that can be used in the case where only experimental data are available for the designer.

For given input-output data pairs of the form \((u_o^p, v_o^p), p = 1, 2, ..., N\), where \(u_o^p \in U\) and \(v_o^p \in V\), a fuzzy control system can be designed from these data pairs. The objective then becomes finding a fuzzy system \(F(x)\) based on \(N\) input-output data pairs. A five-step scheme to design the fuzzy system using the experimental data can be stated as follows:

1. Define fuzzy sets to cover the input and output spaces. For each data pair \([\alpha_i, \beta_i], i = 1, 2, ..., n\), one can define \(N\) fuzzy sets \(A_y (j = 1, 2, ..., N_j)\), which means that for \(u_i \in [\alpha_i, \beta_i]\), there exists \(A_y\) such that \(\mu_{A_y} (u_i) \neq 0\). \(\mu_{A_y} (u_i)\) can be any membership functions shape such as triangular, trapezoidal, etc. A similar thing must be done for \(B_j\); then define \(N_k\) fuzzy sets for \(B_j, (j = 1, 2, ..., N_k)\), which are complete in \([\alpha_k, \beta_k]\). Also, \(\mu_{B_j} (v)\) can be chosen to be any membership function, such as triangular or Gaussian.

2. Generate one rule from one input-output data pair. For each input-output pair \((u_o^p, u_{o1}^p, ..., u_{on}^p, v_o^p)\), find the membership value of \(u_o^p (i = 1, 2, ..., n)\) in fuzzy sets \(A_y (j = 1, 2, ..., N)\) and the membership values of \(v_o^p\) in fuzzy sets \(B_l (l = 1, 2, ..., N_k)\).

Thus, the following must be computed:

\[
\mu_{A_y} (v_o^p) \quad \text{for} \quad j = 1, 2, 3, ..., N_i \quad \text{and} \quad i = 1, 2, 3, ..., n \quad (3.1)
\]

\[
\mu_{B_l} (v_o^p) \quad \text{for} \quad l = 1, 2, 3, ..., N_k \quad (3.2)
\]
For each input variable $u_i (i = 1, ..., n)$, the fuzzy set in which $u_{oi}^p$ has the maximum membership value must be determined, such that $\mu_{A_y^*} (u_{oi}^p) \geq \mu_{A_y} (u_{oi}^p)$ for all $j=1,2,....,N_i$. Similarly, $B_l^*$ must be determined, such that $\mu_{B_l^*} (v_o^p) \geq \mu_{B_l} (v_o^p)$ for all $l=1,2,3,.......,N_k$. Finally, a fuzzy If-Then rule can be obtained as:

* If $u_1$ is $A_1^*$ and $u_2$ is $A_2^*$ and.....and $u_n$ is $A_n^*$, Then $v$ is $B_l^*$. 

3. Assign a degree for each rule generated in Step 2. Since the number of input-output data pairs is proportional to number of rules generated, the probability of creating conflicting rules is very high. To avoid creating conflicting rules, a degree must be assigned to each rule generated and only the rule with the highest degree is kept. This can reduce the number of rules generated and solve the conflicting problem. The degree of a rule is defined as follows:

a. For each rule generated from the input-output pair $(u_o^p, v_o^p)$, the degree of the rule is given by:

$$\mu_{R_l} = \prod_{i=1}^{n} \mu_{A_y^*} (u_{oi}^p) \mu_{B_l^*} (v_o^p)$$ (3.3)

b. If the input-output pairs have different reliability with the reliability number $\eta_p$, where $\eta_p \in [0,1]$, then the degree of the rule generated is given by:

$$\mu_{R_l} = \prod_{i=1}^{n} \eta_p \mu_{A_y^*} (u_{oi}^p) \mu_{B_l^*} (v_o^p)$$ (3.4)

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
\( \eta_p \) can be chosen according to data, if the number of data pairs is small, and the data are not contaminated with noise, and if there is no way to determine \( \eta_p \). Setting \( \eta_p = 1 \) is a good starting assumption.

4. Create the fuzzy rule base. The fuzzy rule base consists of (a) the non-conflicting rules generated in Step 2; (b) the rule from a conflicting group that has the maximum degree; conflicting rules are those rules that have the same IF parts; and (c) linguistic rules from human experts.

5. Construct the fuzzy system based on the fuzzy rule base. Any fuzzy system can be chosen in this step provided that fuzzification, defuzzification, and the inference engine are chosen accordingly. For example, the singleton fuzzifier, COG defuzzifier, and product inference engine can be used to construct the fuzzy system.

The shortcoming of this method, however, is that the fuzzy rule base generated by this method may not be complete, i.e., some rules that are needed will not be generated, so that in the rule base there will be rules that cause no control actions. The rules that are missing can then be found by interpolating from the existing rules. The other method of designing fuzzy system from experimental data is in the utilization of training algorithms. These methods are discussed next.

**Fuzzy Control System Using Gradient Descent Training**

In this approach, the structure of the fuzzy system is specified first and some parameters in the structure are free to change. Then these free parameters are
determined according to the input-output data pairs. The steps involved in this approach as listed in Wang (1997) can be summarized as follows:

1. Specify the structure of the fuzzy system to be designed by choosing the inference engine operators and the defuzzifier operator. For the product inference engine, singleton fuzzifier, center average defuzzifier, and Gaussian membership function, the fuzzy system can be expressed by:

\[
\begin{align*}
    f(u) &= \frac{\sum_{i=1}^{N} b_i \left\{ \prod_{j=1}^{N} \exp \left( \frac{-(u_j - u_j')^2}{(\sigma_j')^2} \right) \right\}}{\sum_{i=1}^{N} \left\{ \prod_{j=1}^{N} \exp \left( \frac{-(u_j - u_j')^2}{(\sigma_j')^2} \right) \right\}} \\
    &\quad \text{(3.5)}
\end{align*}
\]

where $N$ is fixed, and $b_i, u_j', \text{ and } \sigma_j'$ are free parameters. Once the free parameters are specified, the fuzzy system is obtained. This method is shown graphically in Figure 16.

![Fuzzy System Design Using Gradient Descent Training](image)

Figure 16. Fuzzy System Design Using Gradient Descent Training.
As shown in Figure 16, the input $u$ is passed through a product Gaussian operator to find the parameters $Z_i$, which can be calculated by:

$$Z_i = \prod_{j=1}^{n} \left( \frac{|u_j - u_j'|}{\sigma_j^i} \right)^2$$  \hspace{1cm} (3.6)

The values of $Z_i$ found by using Equation (3.6) are then passed through a summation operator and a weighted summation operator to obtain $m$ and $n$ which are given by:

$$m = \sum_{i=1}^{N} Z_i \hspace{1cm} (3.7a)$$

$$n = \sum_{i=1}^{N} b_i Z_i \hspace{1cm} (3.7b)$$

and, finally, the output of the fuzzy system is computed as:

$$f(u) = \frac{n}{m} \hspace{1cm} (3.7c)$$

Gradient descent is then used to minimize the error, which is given by:

$$e_p = \frac{1}{2} [f(u_o^p) - v_o^p]^2 \hspace{1cm} (3.7d)$$

so the task becomes the determination of the parameters $b_i$, $u_j'$, and $\sigma_j'$ such that the error $e_p$ is minimized. Using the gradient decent method, the parameters can be specified as shown in Equations (3.8), (3.9), and (3.10).
where $i = 1, 2, ..., N$, $t = 0, 1, 2, ...$ and $\rho$ is a constant step size. The design steps can then be summarized as:

1. Choose the fuzzy system and determine the number of rules $N$.

2. Specify the initial parameters $b_i(0), u'_j(0), \text{ and } \sigma'_j(0)$.

3. Present $u^p$ for a given input-output pair $(u^p, v^p)$, where $p = 1, 2, ...$ and at the $k$th stage of training $k = 0, 1, 2, ..., \text{ to the input layer of the fuzzy system and compute the output of the system as depicted in Figure 16. That is, compute } Z_i, m, n, \text{ and } f(u)$.

4. Use the training algorithm for $b_i(t + 1), u'_j(t + 1), \text{ and } \sigma'_j(t + 1)$ to update the parameters.

5. Repeat step 3 by setting $t = t + 1$ until the error $e_p$ or until $k$ is reached.

6. Repeat the process by going to Step 3 and taking another data pair, i.e., $p = p + 1$.

7. If desirable and feasible, the parameter $p$ is set equal to 1, i.e., ($p = 1$) and steps 3 to 6 are repeated until the fuzzy system design is satisfactory. This step is
good only in off-line situations.

In this approach, one should keep in mind that the initial parameters choice is very important to the success of the algorithm. The parameters that need to be found are the parameter of the centers of the fuzzy sets in the Then parts of the rules \( b_i \) and the centers and width of Gaussian fuzzy sets in the If pairs of the rule \( u'_j, \sigma'_j \), respectively, so that the fuzzy system can recover the fuzzy If-Then rules.

Fuzzy Control Systems Design Using Recursive Least Squares

Another approach that can be used for training fuzzy systems is accomplished by using Recursive Least Squares (RLS). The objective here is to develop a training algorithm that minimizes the summation of the matching errors for all the input-output pairs up to \( k \), that is, the objective is to design a fuzzy system \( f(u) \) such that Equation (3.11) is minimized. The recursive least squares method can achieve that, and in doing so, the fuzzy system designed \( f_k(u) \) can be expressed as a function of \( f_{k-1}(u) \).

\[
J_k = \sum_{j=1}^{k} [f(u'_j) - v'_j]^2
\]  
(3.11)

The steps involved in this scheme as given by Wang (1997) are:

1. For each \([\alpha_i, \beta_i] \), \( i = 1, \ldots, n \), define \( N_i \) fuzzy sets \( A_i^\beta \), \( (f_i = 1, 2, \ldots, N_i) \), which are complete in \([\alpha_i, \beta_i] \). \( A_i^\beta \) can be chosen as a triangular membership function, trapezoid membership function, etc.
2. Construct the fuzzy system If-Then rules. The number of rules \( N_r \) is given by Equation (3.12a) and the rules are as expressed in Equation (3.12b):

\[
N_r = \prod_{i=1}^{n} N_i \tag{3.12a}
\]

If \( u_i \) is \( A_i^{f_1} \) and \( \ldots \ldots \) and \( u_n \) is \( A_n^{f_n} \) then \( v \) is \( B_{f_1-f_n} \)

\[
\tag{3.12b}
\]

where \( B_{f_1-f_n} \) is any fuzzy set with center at \( b_{f_1-f_n} \) which is free to change.

The resulting fuzzy system using the product inference engine, singleton fuzzifier, and center defuzzifier can be expressed mathematically as shown in Equation (3.13):

\[
f(u) = \frac{\sum_{j_1=1}^{N_1} \ldots \sum_{j_n=1}^{N_n} b_{j_1 \ldots j_n} [\prod_{i=1}^{n} \mu_{A_i}^{f_i}(u_i)]}{\sum_{j_1=1}^{N_1} \ldots \sum_{j_n=1}^{N_n} [\prod_{i=1}^{n} \mu_{A_i^{f_i}}(u_i)]} \tag{3.13}
\]

where \( b_{j_1 \ldots j_n} \) are free parameters to be designed, and \( A_i^{f_i} \) are designed in Step 1.

3. Collect the parameters \( b_{j_1 \ldots j_n} \) into the \( \prod_{i=1}^{n} N_i \) dimensional vector

\[
\mathcal{G} = \left\{ b_{1,1}, \ldots, b_{N_1,1}, \ldots, b_{N_1,2,1}, \ldots, b_{N_1,2,N_n}, \ldots, b_{N_1,2,\ldots N_n}, \ldots, b_{N_1,2,\ldots N_n} \right\}^T
\tag{3.14}
\]

and express \( f(u) = b^T(u) \mathcal{G} \), where \( b(u) \) is given by Equation (3.15):
\[ b_{m_1 \ldots m_n}(u) = \frac{\prod_{i=1}^{n} \mu_{i,m_i}(u_i)}{\sum_{m_1} \left( \sum_{m_2} \left( \prod_{i=1}^{n} \mu_{i,m_i}(u_i) \right) \right)} \]  

(3.15)

4. The initial parameters can be chosen as follows:

a. If there are linguistic rules from human experts whose If parts agree with the If parts of the rule express as:

If \( u_i \) is \( A_{i,m_i}^{l_i} \) and ... and \( u_n \) is \( A_{n,m_n}^{m_n} \) Then \( v \) is \( B_{m_1 \ldots m_n} \)

then \( b_{m_1 \ldots m_n}(0) \) can be chosen to be the center of the Then parts of the fuzzy sets in these linguistic rules.

b. Choose \( b(0) = 0 \) if part (a) is not satisfied.

5. For \( K = 1, 2, \ldots \), compute the parameter \( b \) using the recursive least squares algorithm given by:

\[
\begin{align*}
\theta(k) &= \theta(k-1) + \gamma(k)[v_k b^T(u_k)\theta(k-1)] \\
\gamma(k) &= B(k-1)b(u_k)B(k-1)b(u_k)b^T(u_k)B(k-1)b(u_k) + 1]^{-1} \\
B(k) &= B(k-1) - B(k-1)b(u_k)B(k-1)b(u_k)b^T(u_k)B(k-1)b(u_k) + 1]^{-1}b^T(u_k)B(k-1)
\end{align*}
\]  

(3.16)

taking into account that \( B(0) = \sigma I \), where \( \sigma \) is a large constant and I is identity matrix.

The designed fuzzy system \( f(u) \) is then obtained with the parameters \( b_{m_1 \ldots m_n} \) equal to the corresponding elements in \( \theta(k) \).
Training TSK Fuzzy Systems Using Recursive Least Squares Methods

The fuzzy model that is best suited for use in RLS training is the TSK because of its functional capabilities. The training algorithms of the recursive least squares method are often used in training conventional, neural, or fuzzy-based control systems. The training of the TSK fuzzy control model using the RLS method can be proceeded Passino et al. (1998) as follows.

The TSK fuzzy system model as stated in Chapter II is given by:

\[ f(u) = \sum_{i=1}^{N} g_i(u) \mu_i(u) \]

where

\[ g_i(u) = a_{i0} + a_{i1} u_1 + \ldots + a_{in} u_n \]

The function \( f(u) \) can also be expressed as:

\[ f(u) = \frac{\sum_{i=1}^{N} a_{i0} \mu_i(u)}{\sum_{i=1}^{N} \mu_i(u)} + \frac{\sum_{i=1}^{N} a_{i1} u_1 \mu_i(u)}{\sum_{i=1}^{N} \mu_i(u)} + \ldots + \frac{\sum_{i=1}^{N} a_{in} u_n \mu_i(u)}{\sum_{i=1}^{N} \mu_i(u)} \]

Equation (3.19) can also be expressed as:

\[ f(u / \lambda) = \lambda^T \psi(u) \]
where

\[ \psi(u) = [\psi_1(u), \psi_2(u), \ldots, \psi_N(u), u, \psi_1(u), \ldots, u_N \psi_N(u), \ldots, u_n \psi_n(u), \ldots, u_n \psi_n(u)]^T \]  \tag{3.21} 

and

\[ \psi_i = \frac{\mu_i(u)}{\sum_{i=1}^{N} \mu_i(u)} \]  \tag{3.22} 

\[ \lambda = [a_{10}, a_{20}, \ldots, a_{N0}, a_{11}, a_{21}, \ldots, a_{N1}, \ldots, a_{1n}, a_{2n}, \ldots, a_{Nn}]^T \]  \tag{3.23} 

This can be done by picking in advance the \( \mu_i(u) \) and hence \( \psi_i(u) \) vector. The \( i \)th input and output data pair is denoted by \( (u', v') \) where \( u' \in U \) and \( v' \in V \) and \( v' = f(u') \)

and \( u' = \{u'_1, u'_2, u'_3, \ldots, u'_n\} \) represent the input vector for the \( i \)th data pair. The input-output data pairs and training data set can be represented by:

\[ G = \{(u^1, v^1), \ldots, (u^m, v^m)\} \]  \tag{3.24} 

where \( m \) is the number of input-output data pairs contained in \( G \). Also, the output data vector \( (V) \) and the \( m \times n \) matrix consisting of the \( u' \) data stack \( (\Theta) \) can be expressed as:

\[ V = [v^\top, v^2, \ldots, v^m] \] 

\[ \Theta = \begin{bmatrix} \psi_1^\top \\ \psi_2^\top \\ \vdots \\ \psi_n^\top \end{bmatrix} \]  \tag{3.25} 

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
The two parameters \( \lambda \) and \( p(k) \) gives the recursive least squares as:

\[
\begin{align*}
\lambda(k) &= p(k)u^k(v^k - (u^k)^T \lambda(k - 1) + \lambda(k - 1) \\
p(k) &= p(k - 1) - p(k - 1)u^k(I + (u^k)^T p(k - 1)u^k)^{-1}(u^k)^T p(k - 1)
\end{align*}
\]  (3.26)

where

\[
\begin{align*}
\lambda(k) &= (\Theta^T(k)\Theta(k))^{-1}\Theta^T(k)\nu(k) \\
p(k) &= (\Theta^T(k)\Theta(k))^{-1} = \sum_{i=1}^{k} u^i(u^i)^T)^{-1}
\end{align*}
\]  (3.26a, 3.26b)

The resulting \( f(u/\lambda) \) function in case a Gaussian membership is used for the training is as shown in Equation (3.27):

\[
\begin{align*}
f(u/\lambda) &= \frac{\sum_{i=1}^{N} b_j \prod_{j=1}^{n} \exp\left(-\frac{1}{2} \left(\frac{u_i - c_j^i}{\sigma_j^i}\right)^2\right)}{\sum_{i=1}^{N} \prod_{j=1}^{n} \exp\left(-\frac{1}{2} \left(\frac{u_j - c_j^i}{\sigma_j^i}\right)^2\right)}
\end{align*}
\]  (3.27)

where \( b_i \) is the point in the output space at which the output membership function for the \( i \)th rule achieves a maximum, \( c_j^i \) is the point in the \( j \)th input universe of discourse where the membership function for the \( i \)th rule achieves a maximum, and \( \sigma_j^i \) is the relative width of the membership function for the \( j \)th input and the \( i \)th rule.
Design of the Fuzzy System Using Clustering

Fuzzy clustering is best suited when the system can be described only by a set of input/output data. It has been used very often in neuro-fuzzy systems to train a fuzzy control system. In this method, the input-output data pairs are clustered into groups and each group is then represented by one rule, i.e., the number of clusters equal to the number of rules. An optimal fuzzy system that can match all the input-output pairs to arbitrary accuracy is essential in fuzzy clustering. An optimal fuzzy system for input/output pairs of length \( N \) can be represented using Gaussian membership functions by:

\[
f(u) = \frac{\sum_{i=1}^{N} b_i \exp \left( -\frac{|u - u_i^o|}{\sigma} \right)^2}{\sum_{i=1}^{N} \exp \left( -\frac{|u - u_i^o|}{\sigma} \right)^2}
\]  

(3.28)

In this optimal fuzzy system membership function is expressed by:

\[
\mu_{A_y}(u_i) = \exp \left( -\frac{|u_i - u_i^o|}{\sigma} \right)^2
\]  

(3.29)

where \( b_i \) is the center of \( B_i \). Product inference engine, singleton fuzzifier, center average defuzzifier are assumed in this system.
If the number of input-output pairs is large, the optimal fuzzy system defined above becomes impractical. Other clustering techniques can be used to group the input/output pairs so that a group can be represented by one rule. Two most often used clustering techniques are the nearest neighborhood clustering and c-mean clustering. In the following sections, a discussion of the two clustering methods is presented.

Nearest Neighborhood Clustering Technique

Nearest neighborhood clustering algorithm is one of many clustering techniques (Wang, 1997; Passino and Yurkovich, 1998). It is simple and can be conducted using the following steps:

1. Establish the cluster center $u^1_c$ at $u^1_o$ for the first input/output pair $(u^1_o, v^1_o)$.

2. Set $A^1(1) = v^1_o$ and $B^1(1) = 1$

3. Select the cluster radius $r$

4. For the $k$th input-output pair $(u^k_o, v^k_o)$, $k = 2, 3, ..., N$, calculate the cluster centers $u^1_c, u^2_c, ..., u^N_c$, and compute the distance of $u^k_o$ of these $N$ cluster centers. The distance is given by:

$$u^k_o = \left| u^k_o - u^l_o \right|, \quad l = 1, 2, ..., m$$ (3.30a)

5. Let the smallest distances be $\left| u^k_o - u^k_c \right|$ and use Equations (3.30) and (3.31) according to cases 1 and 2.
Case 1: If \( |u_o^k - u_c^k| > r \) Then

\[
u_c^{N+1} = u_o^k \tag{3.30b}\]

\[A^{N+1}(k) = v_o^k \text{ and } B^{N+1}(k) = 1 \tag{3.30c}\]

\[A^l(k-1) = A^l(k-1) \text{ and } B^l(k) = B^l(k-1) \text{ for all } l = 1, 2, \ldots, N \tag{3.30d}\]

Case 2: If \( |u_o^k - u_c^k| \leq r \) Then

\[
A^k(k) = A^k(k-1) + v_o^k \\
B^k(k) = B^k(k-1) + 1 \\
A^l(k) = A^l(k-1) \\
B^l(k) = B^l(k-1)
\]

for all \( l = 1, 2, \ldots, N \), and \( l \neq k \).

6. If \( u_o^k \) does not establish a new cluster, then the designed fuzzy system based on the \( k \)th input/output pairs \((u_o^j, v_o^j)\), \( j = 1, 2, \ldots, k \) is given by:

\[
f_k(u) = \frac{\sum_{i=1}^{N} A^i(k) \exp(-\mu(u_c))}{\sum_{i=1}^{N} B^i(k) \exp(-\mu(u_c))} \tag{3.32}\]

If \( u_o^k \) does establish a new cluster, then \( f_k(u) \) is given by:

\[
f_k(u) = \frac{\sum_{i=1}^{N+1} A^i(k) \exp(-\mu(u_c))}{\sum_{i=1}^{N+1} B^i(k) \exp(-\mu(u_c))} \tag{3.33}\]

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
where

\[ \mu(u_c) = \left( \frac{|u - u_c|}{\sigma} \right)^2 \]  \hspace{1cm} (3.34)

7. The process is by repeat by letting \( k = k + 1 \).

The cluster radius \( r \) determines the complexity of the designed fuzzy system. For smaller \( r \), more clusters result and, therefore, a more sophisticated fuzzy system. If \( r \) is large, then the designed fuzzy system is simpler but less sophisticated. A good value for \( r \) can be found by trial and error.

**c-Means Clustering**

Fuzzy c-means clustering is an iterative algorithm that can be used to find the membership grades \( \mu_j \) and cluster centers \( u_c \) to minimize the objective function

\[ J = \sum_{i=1}^{N} \sum_{j=1}^{R} (\mu_j)^m |u_i - u_c|^2 \]  \hspace{1cm} (3.35)

where \( m \) is a design parameter greater than 1 \((m > 1)\). \( N \) is the number of input/output data pairs in the training data set. \( R \) is the number of clusters or number of rules, and \( u_i \) for \( c = 1, ..., N \) is the input portion of the input/output training data pairs. Where \( u_c = \{u_{c1}, u_{c2}, ..., u_{cn}\} \) are the cluster centers and \( \mu_j \) for \( i = 1, ..., N \) and \( j = 1, ..., R \) is the grade membership of \( u_i \) in the \( j \)th cluster. Fuzzy clustering is used to form the
antecedent part of the *If-Then* rules. A c-means clustering combined with pre-
defuzzification, i.e., least squares training for consequent parameters, can be used to
construct multi-input multi-output (MIMO) fuzzy systems. A special case of MIMO
is multi-input-single-output (MISO) is illustrated here, along with the TSK fuzzy
system model that is expressed as stated in the previous section. The general
expression of the TSK model is *If* $H^j$ *Then* $g_j(u)$, where $H^j$ is an input fuzzy
linguistic variable set, $\mu_{H^j}$ is the membership function associated with $H^j$ that
represents the premise certainty for rule $j$. The fuzzy system can be represented by:

$$f(u) = \frac{\sum_{j=1}^{R} g_j(u)\mu_{H^j}(u)}{\sum_{j=1}^{R} \mu_{H^j}(u)}$$  \hspace{1cm} (3.36)

The steps that are involved in this method can be summarized as:

1. Choose the fuzziness factor $m > 1$. This determines the amount of overlap
   of the clusters. When $m$ is much larger than 1 ($m >> 1$), then points with less
   membership in the $j$th cluster have less influence on the determination of the new
   cluster center.

2. Specify the number of clusters $R$. The number of clusters equals the
   number of the rules in the rule-base and must be less or equal to the number of data
   pairs in the data pairs set ($R \leq N$).
3. Specify the error tolerance $\varepsilon > 0$, which is the amount of error allowed in calculating the cluster centers.

4. Initialize the cluster centers $u^j_c$ via a random number generated so that each component of $u^0_c$ is no larger than the largest corresponding component of the input portion of the data.

5. Compute the new cluster centers $u^j_c$ based on the previous cluster centers so that the objective function is minimized, according to Equations (3.37) and (3.38).

$$u^j_{\text{new}} = \frac{\sum_{i=1}^{N} u^j (\mu_{ij}^{\text{new}})^n}{\sum_{i=1}^{N} (\mu_{ij}^{\text{new}})^n}$$  

(3.37)

$$\mu_{ij}^{\text{new}} = \left( \sum_{k=1}^{R} \left( \frac{u^j - u_{i\text{old}}}{u^j - u_{k\text{old}}} \right)^{2 \cdot m-1} \right)^{-1}$$

(3.38)

for $i = 1, ..., N$, $j = 1, 2, ..., R$ and $\sum_{j=1}^{R} \mu_{ij}^{\text{new}} = 1$

If $|u^j - u_{i\text{old}}|^2 = 0$ for some $j = 1, 2, ..., R$, then let $\mu_{ij}$ for all $i$ be any nonnegative numbers such that $\sum_{j=1}^{R} \mu_{ij}^{\text{new}} = 1$ and $\mu_{ij} = 0$.

If $|u^j - u_{i\text{old}}|^2 \neq 0$, then $\mu_{ij}^{\text{new}}$ is found using Equation (3.38).
6. Calculate the distance between the current cluster centers and the previous
cluster center, i.e.,

$$d_c = |u_{\text{new}}^j - u_{\text{old}}^j|$$

(3.39)

If $d_c < \epsilon$ for all $j = 1, 2, ..., R$, then the new cluster centers accurately represent the
input data and the fuzzy clustering process is terminated. Otherwise, iterate using the
equations above until $d_c < \epsilon$.

7. Use the final value of $u_j, j = 1, ..., R$ to specify the premise membership
function for the $r$th rule which can be expressed as:

$$\mu_{H_j}(u) = \frac{1}{\sum_{k=1}^{R} \left( \frac{2}{|u - u_j^k|} \right)^{-1}}$$

(3.40)

where $u_j$ is the cluster centers from the last iteration.

8. Specify the consequent of the rules using least squares as follows:

For each cluster center $u_j$ minimize the cost function, which is given by

Equation (3.41):

$$J_j = \sum_{i=1}^{N}(\mu_j)^2(v_i - (\bar{u}^i)^T a_j)^2$$

(3.41)

where

$$\bar{u}^i = \{1, (u^i)^T\}^T, j = 1, 2, ..., R,$$
and \( v^i \) is the output portion of the \( i \)th data pair. The parameters \( a_j \) are found using the expression in Equation (3.43):

\[
a_j = (\bar{u}^T D_j \bar{u})^{-1} \bar{u}^T D_j v
\]  

(3.43)

where

\[
\bar{u} = \begin{pmatrix}
1 & \cdots & 1 \\
\vdots & \ddots & \vdots \\
u^n & \cdots & u^n
\end{pmatrix}^T
\]

\[
v = [v^1, v^2, \ldots, v^N]^T
\]

(3.45)

\[
D_j^2 = (\text{diag}([\mu_1, \mu_2, \ldots, \mu_N]))^2
\]

9. Choose the parameters \( a_j \bar{u}^i \) for \( j = 1, 2, \ldots, N \) such that the \( J_j \) minimized and use those parameters for \( g_j (\bar{u}) \) in TSK model.

This concludes the control system design using clustering techniques. A brief introduction to clustering in general was discussed in Chapter II. The aim of this chapter was to show how these techniques can be used to design a control system based on predefined implication, fuzzification, and defuzzification methods. The fuzzy clustering technique is used in Chapter V to design a fuzzy control system for idle speed using experimental data.
Design of a Fuzzy Control System Using the Learning from Examples Method

The learning from example technique generates a rule-base for a fuzzy system by using numerical data from a physical system and linguistic information from human experts. A multi-input-single-output (MISO) case is illustrated here. The case illustrated is based on the assumption that the fuzzification is singleton, the premise and implication operator is the minimum operator, and the defuzzification method used is COA. Keeping in mind that any other choices of fuzzification, defuzzification, and implication operators can be used.

The following guidelines in constructing a fuzzy system by this method (learning from example method) are helpful.

1. The membership function’s shape should be chosen first for each of the inputs and outputs.

2. The universe of discourse for the inputs and outputs are then defined according to the expected range of variation in the input and output variables. Two input and one output case can be defined by \( U = [u_{-}, u_{+}] \), \( i = 1,2 \), for the two inputs, and \( V = [v_{-}, v_{+}] \) for the output.

3. The number of the membership functions on each universe of discourse should be defined. The number of the membership functions affects the accuracy of the designed system. Fewer membership functions result in lower accuracy, while a large number can lead to an explosion of rules and results in a complex system.

4. The associated membership functions \( \mu_{u_{i}}(u_{i}) \) and \( \mu_{v_{j}}(v) \) should be defined.
on the input and output universe of discourses, respectively.

5. The system is then trained using the data pairs in the data set \((u', v') \in G\) to form the premise of the rules, and using the output of the data pairs \(v'\) to form the consequents. The rules can be expressed as:

\[ R_1: \text{If } u_1 \text{ is } u_1^j \text{ and } u_2 \text{ is } u_2^k \text{ then } v \text{ is } V^l \]

6. The associated degree for the \(i\)th rule is calculated using any of the \(\ell\)-norms operators by:

\[
\mu_{R_i} = \mu_{u_1} (u_1) \mu_{u_2} (u_2) \mu_{v_i} (v)
\]

(3.46)

7. Another pair of training data should be trained i.e.

\[
d^m = ([u_1^m, u_2^m]^T, v^m)
\]

(3.47)

so that it yields a fuzzy rule such as:

\[ R_m: \text{If } u_1 \text{ is } u_1^j \text{ and } u_2 \text{ is } u_2^k \text{ then } v \text{ is } V^l \]

8. The following can be used to add or reject a rule:

a. If \(\mu_{R_m} > \mu_{R_i}\) for all \(i \neq m\) such that the rules \(R_i\) are already in the rule-base and \(\mu_{R_i}\) is evaluated for \(d^m\) and the premises for \(R_i\) and \(R_m\) are the same, then rule \(R_m\) replaces rule \(R_i\) in the existing rule base.

b. If \(\mu_{R_m} \leq \mu_{R_i}\) for some \(i, i \neq m\) and the premises for \(R_i\) and \(R_m\) are the same, then rule \(R_m\) is rejected.

c. If \(R_m\) does not have the same premise as any other rule already existing in the rule-base, then \(R_m\) is added to the rule-base.
9. The process should be repeated considering all data pairs \( i = 1, 2, 3, \ldots, N \) until it is completed.

This concludes the methods of designing a fuzzy system that are of interest in this dissertation. The following section shows how to construct a fuzzy system utilizing conventional control knowledge and techniques.

Design of a Fuzzy Logic Controller Using Conventional Control Techniques

In this section, we present different types of fuzzy logic control that can be designed using knowledge acquired from conventional control techniques for nonlinear and linear, and time variant and time invariant systems. Methods such as adaptive control, supervisory control, sliding mode control, and PID control are some of many conventional control methods that have been utilized by fuzzy systems' designers and engineers. Other methods that utilize neural network and genetic algorithms are beyond the scope of this dissertation and they are not considered. The reader is referred to the Appendix for valuable references on using neural network and genetic algorithms in fuzzy control scheme.

Knowledge acquired from conventional control methods has been utilized and adapted to work for fuzzy logic control system, in the same way as conventional control. For example, in traditional adaptive control, the different approaches used in control systems that are modeled based are applicable to fuzzy logic control. The next section shows how an adaptive fuzzy control system can be designed in a way similar to traditional adaptive control.
Adaptive Fuzzy Logic Control

Fuzzy controller design based on heuristic information from human experts has found success in many industrial applications. Moreover, constructing a fuzzy controller from experiential data in the form of input/output data pairs as discussed earlier is finding use in present applications.

There are certain problems that are encountered in practical control problems, such as the difficulty of choosing the controller parameters, like the membership functions and the fuzzy inference system, so that specific performance criteria are met. Another problem is the difficulty in performing the initial synthesis of the fuzzy controller if the fuzzy controller was constructed for a nominal plant and there is noise or different types of disturbances exist that were not anticipated in the initial design process.

There are two general approaches to conventional and fuzzy adaptive control, the first of which is called directive adaptive control, shown in Figure 17.

![Figure 17. Conventional Direct Adaptive Control Scheme.](image-url)
In this approach, the adaptation mechanism observes the signals from the control system and adapts the parameters of the controller to maintain performance under all conditions, such as disturbance and/or plant changes. If a reference model is used such that the performance of the controller is judged based on the reference model performance, the model is called model reference adaptive control (MRAC). The second approach to adaptive control is called indirect adaptive control, shown in Figure 18. As can be seen in Figure 18, this method is suited for an on-line system identification to estimate the parameters of the plant, and from the estimated parameters, the controller is designed or redesigned if required. In this method, the actual plant parameters are assumed to be very close or identical to the estimated parameters at all times.

Figure 18. Model Reference Adaptive Control.
**Direct Model Reference Adaptive Controller**

A block diagram for the direct model reference adaptive or learning controller is shown in Figure 19.

![Block Diagram of Direct Model Reference Adaptive Controller](image)

Figure 19. Direct Model Adaptive Reference Fuzzy Control.

The block diagram shown in Figure 19 consists of four parts, which are the fuzzy controller, the reference model, the learning mechanism, and the plant. For a PD type fuzzy logic controller, where the error and its rate are taken to be the inputs to the fuzzy controller, the block diagram can be decomposed as shown in Figure 20a. A supervisory fuzzy controller is shown in Figure 20b and will be discussed in depth in a later section.

Assuming discrete time system, the fuzzy model reference adaptive controller uses the learning mechanism to observe numerical data from a fuzzy control system.
Figure 20. A PD-type Direct Adaptive Reference Controller (a) and A Fuzzy Supervisory Controller (b).

\((r(nT), y(nT)),\) where \(T\) is a sampling period. The current performance and the adjustment of the fuzzy controller to meet desired performance criteria is done using the numerical input data. The performance objectives are specified in the reference model. In this procedure, the plant has an input \(u(nT)\) and output \(y(nT)\). A prefilter
can be placed between the reference input and the summation. The prefilter makes
sure that smooth and reasonable request is made of the fuzzy controller. The choice
of the input to the fuzzy controller and the processing of \( r(nT) \) and \( y(nT) \) should be
properly made; otherwise, the performance and stability of the fuzzy controller can be
compromised.

The rule-base for the PD-adaptive reference model fuzzy controller has rules
of the form

\[
\text{If error is } A_j \text{ and error rate is } B_l, \text{ Then } u \text{ is } U_m
\]

where \( A_j, B_l \) and \( U_m \) denotes the \( j \)th, \( l \)th, and \( m \)th linguistic value associated with
error, rate of error, and the controller output.

The number of the rules depends mainly on the number of membership
functions on each input universe of discourse. If the number of the membership
function on the error universe of discourse is \( n \) and its rate is \( m \), then the rule-base
will have total of \((nm)\) rules. The input membership functions are defined to
characterize the premises of the rules that define the various situations in which rules
should be applied. To initialize the rule base, the membership functions for the input
are left constant and not tuned by the adaptation scheme. The membership functions
on the output universe of discourse are assumed to be unknown and will be
automatically synthesized or tuned by the fuzzy model reference learning controller.

The output membership functions' initial values should be chosen for each
one. Triangular, trapezoidal, Gaussian, or any other type of membership function can
be chosen to represent the output membership functions.

The conjunction in the premise of the rules, implication, and the composition operator need to be specified. Most often used operators are the minimum or product.

The defuzzification method needs to be specified so that the specification of the fuzzy controller can be complete. Most often-used defuzzification methods are the center of gravity (COG) method, also called the center of the area (COA) method.

The reference model is sometimes hard to know because the performance level expected is unknown. The reference model, however, should be chosen so that a desired specified performance is satisfied. The reference model may be discrete or continuous, linear or nonlinear, and time-invariant or time varying. A given reference model characterizes the design criteria such as rise-time and overshoot. The input to the reference model is the reference input \( r(nT) \); the desired performance of the controlled process is met if the learning mechanism forces \( y_e(nT) \) to remain very small for all time, no matter what the reference input is and for any plant parameters variations.

The learning mechanism tunes the rule-base of the direct fuzzy controller so that the closed loop system behaves like the reference model. The rule-base modifications are achieved using the observed data of the controlled process, reference model, and the fuzzy controller. The learning mechanism consists of two parts: a fuzzy inverse model and knowledge-based modifier. The fuzzy inverse model performs the function of mapping the deviation between the process output and the reference model to changes in the process inputs \( p(nT) \) that are necessary to force...
$y_e(nT)$ to zero. The knowledge-base modifier performs the function of modifying the fuzzy controller's rule-base to effect the needed changes in the process inputs.

To change the process inputs $p(nT)$, the inverse model of the plant uses a fuzzy system to map $y_e(nT)$ and functions of $y_e(nT)$ such as:

$$y_e(nT) = \frac{1}{T}(y_e(nT) - y_e(nT - T))$$  \hspace{1cm} (3.48)

The fuzzy system then tries to reduce the error $y_e(nT)$ by appropriate changes in $p(nT)$. The fuzzy inverse model contains rules of the form

If $y_e$ is $Y_{e l}$ and $y_e$ is $Y_{cl}$ Then $p$ is $P_m$

where $Y_{e l}, Y_{cl}$ denote linguistic values for $y_e$ and $y_e$, respectively, and $P_m$ denotes the linguistic value associated with the $m$th output fuzzy set. The same procedures can be used in choosing the fuzzy system inference, fuzzification, and defuzzification as was done in the direct fuzzy controller case.

The knowledge-base modifier changes the rule-base of the fuzzy controller given the information about the necessary changes in the input, $p(nT)$, so that the error becomes zero.

For a value of $y(nT)$ that is different from $y_e(nT)$ and assuming that the plant input can effect the plant output in one step, then by modifying the fuzzy controller's knowledge base, the fuzzy controller can be forced to produce a desired output $u(nT - T) + p(nT)$, which is the input at time $(nT - T)$ to make $y_e(nT)$ smaller. If the next time similar values of the error and change of error are obtained, then the input
to the plant will be one that will reduce the error between the reference model and plant output.

For symmetric output membership functions and for a fuzzy controller with $b_m$ denoting the center of the membership function associated with $u_m$, knowledge-base modification is performed by shifting the center $b_m$ of the membership functions of the output linguistic value $u_m$ associated with the fuzzy controller rules that contributed to the previous control action $u(nT-T)$. This is can be done in two steps:

1. Find all the fuzzy controller active rules that have premise certainty greater than zero, i.e., $\mu_i(e(nT-T), \dot{e}(nT-T)) > 0$.

2. For the $m$th output membership function at time $nT$, define the center to be $b_m(nT)$, and for all active rules, calculate the value of $b_m(nT)$ using Equation (3.49).

$$b_m(nT) = b_m(nT-T) - p(nT)$$

so that the output membership function centers are modified.

Guidelines for the Fuzzy Inverse Model Design

Since choosing the inverse fuzzy model in most cases is application dependent, i.e., the fuzzy inverse model is derived from the direct fuzzy controller, guidelines for the inverse model can be constructed taking this fact into consideration. The first guideline is based on the fact that the fuzzy inverse model often takes on a form quite similar to a direct fuzzy controller. For example, the rule-base often has some typically symmetrical properties. The second guideline is based
on the fact that the inverse fuzzy model is defined so that the response of the plant follows very closely the output of the reference model. In this case, the fuzzy inverse model turns off the adaptation if at some time \( t \), \( y_e = (y - y_m) \approx 0 \). This is known as the satisfaction principle, which means that the inverse model is satisfied with the response as long as it is close to the reference model. Another way to achieve the result thing is to require the fuzzy inverse model to stop functioning if certain conditions are satisfied. A condition of the form \( p(nT) < \varepsilon \) Then \( p(nT) = 0 \) can serve this purpose. The value of \( \varepsilon \) can be chosen so that \( \varepsilon > 0 \) and \( \varepsilon \) is a very small positive number. For cases in which the size of the inputs to the fuzzy inverse model is directly proportional to the size of its output, the above rule ensures that when the inputs to the fuzzy inverse model are in the region of zero, its output will be modified to zero. If the error between the plant output and the reference input grows, then the learning mechanism will turn back on and work to reduce the error. For tuning of fuzzy inverse model parameters such as scaling factors, the following can be helpful guidelines:

1. The scaling factors of the model input \( y_e(nT) \) denoted by \( K_e \) is chosen so that the input will not saturate the input membership function near the end points.

2. The scaling factor for the inverse model output \( p(nT) \) denoted by \( K_p \) is chosen to be the same as the scaling factor of the direct fuzzy controller.

3. The scaling factor for the inverse model input change \( K_{e_i} \) is initially chosen to be zero or very close to zero.
4. A step reference input $r(nT)$ is applied with same magnitude as that of normal operation.

5. Three cases can be observed in the plant and reference model responses:
   a. If there is unacceptable oscillation in the plant output response about the reference model response, then $K_e$ should be increased.
   b. If the plant output is unable to follow the reference model response, then $K_e$ should be decreased and Step 4 should be repeated.
   c. If the plant response is acceptable with respect to the reference model response, then the controller design is completed.

**Indirect Adaptive Fuzzy Control**

In this kind of adaptive fuzzy control, on-line identification techniques are necessary to estimate the parameters of a model of the plant. The estimated model of the plant is then used to design the fuzzy controller using the identified parameters. The indirect adaptive fuzzy controller can be considered as an independent in its design. Two most frequently used methods for indirect adaptive fuzzy control are the gradient and recursive least squares methods. It should be noted that the recursive least squares method allows only for tuning parameters that enter the model in a linear fashion, such as the output membership function centers, while the gradient method allows for tuning parameters that enter in a nonlinear fashion, such as the membership function widths. The gradient method, if used, can tune the fuzzy system to act like the plant more than the recursive least squares method can.
**Indirect Adaptive Fuzzy Control Using the TSK Fuzzy Model**

The indirect adaptive fuzzy control can be developed using the TSK fuzzy model. If other fuzzy models are desired then the TSK model is initially used and a conversion algorithm can be written and incorporated in the control scheme. In this section, however, the TSK is utilized to design indirect adaptive fuzzy controller.

The plant can be represented mathematically by:

\[
\begin{align*}
    \dot{x}(t) &= f(x(t), u(t)) \\
    y(t) &= g(x(t))
\end{align*}
\tag{3.50}
\]

where \(x\) is the state, \(u\) is the input to the plant, and \(y\) is the output of the plant.

The on-line identification method can be used to adjust the parameters of the TSK identifier model so that the model behavior matches that of the plant.

The identifier model can be specified by \(R\) rules of the form

\[
\text{If } y(k) \text{ is } A_{i1} \text{ and } \ldots \text{ and } y(k-n-1) \text{ is } A_{in} \text{ Then } y_m(k+1) = a_{i1} y(k) + \ldots + a_{in} y(k-n-1) - B_i u(k) - \ldots - B_m u(k-m-1)
\]

where \(u(k)\) and \(y(k)\) are the plant input and output, respectively; \(A_{ij}\) is the linguistic value, \(B_{ip}, i = 1, \ldots, R, j = 1, 2, \ldots, n,\) and \(p = 1, \ldots, m,\) are the parameters of the consequents; and \(y_m(k+1)\) is the identifier model output considering only rule \(i.\) The identifier model output using the center-average defuzzification method can be expressed as shown in Equation (3.51):
\[ y_m(k + 1) = \frac{\sum_{i=1}^{R} y_{mi}(k + 1) \mu_i}{\sum_{i=1}^{R} \mu_i} \]  

(3.51)

where \( \mu_i \) is the premise certainty for rule \( i \).

Another way to express the identifier model output can take the form as shown in Equation (3.52):

\[ y_{mi}(k + 1) = \theta^T \psi \]

\[ \psi_i = \frac{\mu_i}{\sum_{i=1}^{R} \mu_i} \]

\[ \psi = \{ y(k) \psi_1, y(k) \psi_2, \ldots, y(k) \psi_R, \ldots, u(k - m + 1) \psi_1, \ldots, u(k - m + 1) \psi_R \}^T \]

\[ \theta = \{ \alpha_{i1}, \alpha_{i2}, \ldots, \alpha_{R1}, \ldots, \beta_{im}, \beta_{2m}, \ldots, \beta_{Rm} \}^T \]

An on-line method such as recursive least squares could adjust the \( \alpha_{ij} \) and \( \beta_{ip} \) parameters since they are linear, while the gradient method could adjust \( \alpha_{ij} \) and \( \beta_{ip} \) parameters and the premise parameters, such as the input membership function center and width if Gaussian input membership functions are used.

The controller using the TSK fuzzy model can be expressed as:

\[ \text{if } y(k) \text{ is } A_{ij} \text{ and } \ldots \text{and } y(k-n-1) \text{ is } A_{ni} \text{ Then } u_i(k) \text{ is } g_i(*) \]  

(3.53)

where \( g_i(*) \) is a linear function of its arguments that can depend on past plant inputs and outputs and the reference input, and \( u_i(k) \) is the controller output considering only
rule \( i \). For example, a proportional controller with gains \( k_{i0} \) and \( k_{i1} \) can be used to express the function \( g_i(*) \) and can be stated as:

\[
g_i(R(k), y(k)) = k_{i0} r(k) - k_{i1} y(k)
\] (3.54)

The function \( g_i(*) \) can be chosen to suit the application for which it is designed.

**Fuzzy Supervisory Control**

PID controllers control many processes in the industry. Despite their advantages, conventional PID controllers suffer from the fact that the controller must be re-tuned when the operating conditions change. This disadvantage can be reduced by using a fuzzy supervisor for adjusting the parameters of the low-level controller as can be seen in Figure 20..

A set of rules can be obtained from data or experts to adjust the gains \( K_p \) and \( K_D \) of a PD controller. For example, based on current set-point \( r \), the rules can be expressed as:

- If \( r \) is Low Then \( K_p \) is High and \( K_D \) is Low
- If \( r \) is High Then \( K_p \) is Low and \( K_D \) is Medium

The gains \( K_p, K_D \) can be expressed on their universe of discourse by the linguistic values Low, High, Medium, etc.

The inference mechanism is the same as with direct fuzzy controllers. The rules give a linguistic description of the tuning strategy of an expert. The low-level
controller can be a conventional PID of a fuzzy PID-type controller. The fuzzy supervisory PID can be extended with a fuzzy performance evaluator, as shown in Figure 20. Instead of adjusting the parameters according to the process variables, this structure uses a fuzzy performance evaluation of the response of the low-level controller and adjusts the parameters accordingly, as can be seen in the work of Hans et al. (1991). The performance evaluator uses information about the system response indicators, such as rise time, overshoot, steady state error, etc. The rules of performance evaluation can take the following format:

If Overshoot is Small and Rise Time is Medium and the ITAE is Small Then Performance is Satisfactory.

The supervisory fuzzy logic controller in traditionally conventional controlled applications is gaining popularity and its use has proven to be a valuable addition to existing control systems. This is a brief introduction to the use of fuzzy logic control in a supervisory function.

The Computational Structure of the Fuzzy Logic Controller

A control law represented in the form of a fuzzy logic controller (FLC) depends on the measurements of signals and is thus static control law. This means that the fuzzy rule-based representation of FLC does not include any dynamics, which makes an FLC a static transfer element, like a state controller. It is a nonlinear static transfer element due to the computational steps involved that have nonlinear characteristics. The following sections present a discussion of the design of the fuzzy logic controller in a special case such as the case of regulatory type fuzzy control that
can be used in control systems where conventional control techniques do not perform as desired. Controllers such as proportional, proportional plus derivative (PD), proportional plus integral (PI), and proportional plus derivative plus integral (PID) depends on their inputs on the deviations from a set reference input. The case considered here is when the controller inputs are the error and its derivative or integral. This case is encountered often in the industrial process and automotive industry. Cases such as idle speed control, cruise control, and engine management system control are a few of the many applications in which this kind of controller can be utilized.

The computational structure of FLC consists of a number of computational steps. There are five blocks corresponding to five computational steps, as discussed before and depicted in Figure 21. These steps are: (1) input scaling factors or normalization of the input universe of discourse, (2) fuzzification of the controller

![Figure 21. Fuzzy Logic Controller Structure Showing the Steps Involved in the Inference System.](image)
input variables, (3) inference or rule-firing, (4) defuzzification of the controller-output variables, and (5) Denormalization of the output scaling factor.

Fuzzification, inference engine, and defuzzification are discussed in Chapter II. In the following section, scaling factors and the denormalization factor are discussed.

Scaling Factors

In the state space control system, the state variables can be designated by $x_i$ $(i=1,2,\ldots,n)$. For regulatory type control systems, the states can be represented by the error and its derivatives, i.e., $(e, \dot{e}, \ddot{e}, \ldots, e^n)$. The states usually appear in the $If$-part of the fuzzy rules on an FLC, and therefore they constitute the inputs of the fuzzy logic controller. The controller output variables $(u_1, u_2, u_3, \ldots, u_m)$ appear in the $Then$-part of the rules of an FLC. For a multi-input-single-output (MISO) FLC, the computational steps are as follows. The membership functions that define the fuzzy values of the controller inputs and outputs are defined off-line on a normalized universe of discourse. Thus, the actual values of the controller inputs and controller output are mapped onto the same predetermined normalized universe of discourse. The mapping process is called the normalization. The scaling factors are used for such mapping for the inputs of the controller. Input scaling factors are then the multiplication of a physical, crisp controller input with the scaling factor so that it is mapped onto the normalized universe of discourse. The output scaling is the multiplication of a normalized or scaled controller output with a denormalization
factor so that it is mapped back onto the physical universe of discourse of the controller outputs. The advantage of using this is that fuzzification, rule-firing, and defuzzification can be designed independently of the actual physical universe of discourse of the controller inputs and outputs. For example, if the state vector \((e) = (e, \dot{e})\), where the error is defined as \((e = y_1 - y_d)\), a normalized error can be represented by \(e_N\), and it is given as \(e_N = k_e e\), where \(k_e\) is the scaling factor. The normalized universe of discourse for the error \((e)\) can be represented by \(E_N = [-a, a]\). The same thing is true for the error change. In this case, the scaling factor is denoted by \(k_e\) and the normalized universe of discourse is \(D_E = [-b, b]\). In the context of a phase plane representation of the dynamic behavior of the controller inputs, the scaling factor affects the angle of the sliding line that divides the phase plane into two semi-planes Yager and Filev (1994), as can be seen in Figure 22. This is important when considering the sliding mode fuzzy logic controller. The sliding surface can be affected with the scaling factors, which facilitate the design of the sliding mode fuzzy logic controller.

**Denormalization**

The defuzzified normalized controller output is denormalized using the inverse of the normalization factor \(k_u\). For normalized controller output \(u_N\), the denormalized output \(u\) is obtained from Equation (3.55):
The choice of the scaling factors determines the stability of the controller. If the TSK fuzzy model is used, then scaling factors are not needed.

\[
\begin{align*}
    u_N &= k_u u \\
    u &= k_u^{-1} u_N
\end{align*}
\] (3.55)

Figure 22. The Transformation from Universe of Discourse to Normalized Universe Of Discourse and Its Effect on the Sliding Surface.

Design of the Fuzzy Logic Controller

Two important problems are encountered in the design of the fuzzy logic controller (FLC). The first is the determination of the linguistic values, the membership functions of the associated reference fuzzy sets. The second problem is in the formation of the rule-base of the FLC. Reference fuzzy sets are considered as
parameters. The rule-base of FLC expresses the connections between the input and output variables. The design of the FLC is a unified approach to determine both the parameters, such as linguistic values and reference fuzzy sets, and the structure of the rule-base. Generally, the universes of discourse of input and output variables of the FLC are restricted to intervals that are related to the maximal and minimal possible values of the respective variable, that is, to the operating range of the input and output variables.

In the case of a Mamdani-type PI-like FLC, the universe of discourse of the error \( e(t) = r(t) - y(t) \) is defined by the maximal and minimal values of the variables \( r(t) \) and \( y(t) \). The error interval is defined on the range \([e_{\text{min}}, e_{\text{max}}]\), where

\[
e_{\text{max}}(t) = r_{\text{max}}(t) - y_{\text{min}}(t) \quad (3.56a)
\]

\[
e_{\text{min}}(t) = r_{\text{min}}(t) - y_{\text{max}}(t) \quad (3.56b)
\]

The change of error, denoted by \( \Delta e(t) = e(t) - e(t-1) \) has operating ranges \([\Delta e_{\text{min}}(t), \Delta e_{\text{max}}(t)]\) and \([\Delta u_{\text{min}}, \Delta u_{\text{max}}]\), where

\[
\Delta e_{\text{max}}(t) = e_{\text{max}}(t) - e_{\text{min}}(t) \quad (3.57a)
\]

\[
\Delta e_{\text{min}}(t) = e_{\text{min}}(t) - e_{\text{max}}(t) \quad (3.57b)
\]

\[
\Delta u_{\text{max}}(t) = u_{\text{max}}(t) - u_{\text{min}}(t) \quad (3.57c)
\]
The operating ranges can be adjusted by taking into account the dynamics of the controlled system and the sampling interval. It is often the case that normalized universes of discourse are used for the input/output variables of the FLC, as discussed in the previous section. The normalized universes of discourse are well-defined domains; the fuzzy values of the input/output variables are fuzzy subsets of these domains. In general, the normalized universes can be identical to the real operating ranges of the variables, but, in most cases, they can coincide with the closed interval [-1, 1].

For the cases where the error $e(t)$, change of error $\Delta e(t)$, and change of the controller output $\Delta u(t)$ are defined on the operating ranges $[-x_e, x_e]$, $[-x_{de}, x_{de}]$, and $[-x_{du}, x_{du}]$, respectively, the normalized and denormalized universe of discourses are defined by:

$$U_1 = [-x_e^*, x_e^*], U_2 = [-x_{de}^*, x_{de}^*], V = [-x_{du}^*, x_{du}^*],$$

where

$$x_e^* = x_e k_e \text{ and } x_{de}^* = x_{de} k_e$$

$k_e$ and $k_{de}$ are the scaling factors that transform the universe of discourse range into [-1, 1] or [0, 1] interval.

The normalized universes are then defined for normalized error, error change as:

$$e_N(t) = k_e e(t)$$ (3.58a)
\[ \Delta e_N(t) = k_e \Delta e(t) \]  

(3.58b)

The reasoning mechanism then applies to these normalized values \( e_N(t) \) and \( \Delta e_N(t) \).

The defuzzified value \( \Delta u^*(t) \), obtained by application of the FLC algorithm, belongs to the normalized universe \( V = [-x_{\text{e}}, x_{\text{e}}^*] \) and it is related to the actual change of control variable \( \Delta u(t) \) from the operating range \( [-x_u, x_u] \) through the scaling factor \( k_{du} \):

\[ \Delta u(k) = k_{du} \Delta u^*(t) \]  

(3.59)

The scaling factor \( k_{du} \) is the denormalization factor. It brings the output of FLC from the normalized universe \( V \) to its actual operating range \( [-x_u, x_u] \).

A similar approach for PD-type fuzzy logic controllers can be constructed with \( e_N(t) = k_e e(t), \Delta e_N(t) = k_e \Delta e(t) \), and \( u(t) = k_u u^*(k) \), which are the same as in PI-type fuzzy controllers stated above except for minor changes in the scaling factor for the controller output \( k_u \) which is different from \( k_{du} \) for PI-type FLC.

If the universe of discourse is defined on non-symmetric interval \( [a, b] \), then for normalization purposes, the non-symmetric interval can be transformed to a symmetric universe \( [-c, c] \), using a linear transformation of the form:

\[ x^* = k_1 x + k_2 \]  

(3.60)

where \( x \in [a, b], x^* \in [-c, c] \).
Applying the boundary conditions, $k_1$ and $k_2$ can be determined using Equation (3.61):

$$
k_1 = \frac{2c}{b-a} \quad k_2 = \frac{a+b}{b-a} c
$$

(3.61)

This transformation is applied only when the interval universes are almost symmetric and there is $g \in [a, b]$ such that $g < 0$; otherwise, the transformation does not work.

Input and output variables of the FLC are usually quantified into sets defined by linguistic values such as "positive small," "negative large," and so forth. Linguistic terms and these associated fuzzy sets form a fuzzy partitioning of the normalized universes of discourse. For example, the input variable error (linguistic value) determines a fuzzy partitioning of the universe $[-1, 1]$. The numbers of linguistic values for each variable or for each input determines the cardinality of the fuzzy partitioning of that universe of discourse. Cardinalities of the term sets associated with the inputs of the FLC define the maximal number of rules contained in its rule-base. The maximum number of rules for a FLC with $n$-input and $m$ fuzzy linguistic values for each input is given by:

$$
N_R = m^n
$$

(3.62)

where $N_R$ is the total possible number of fuzzy If-Then rules in the rule-base.
If the number of linguistic variables defined for each input of the FLC is different from one input to another, then one can use Equation (3.63) developed in this dissertation to calculate the total number of rules in the rule-base:

\[ N_R = \prod_{i=1}^{n} m_i^{p_i} \quad (3.63) \]

where

- \( N_R = \) total number of rules
- \( n = \) number of inputs
- \( m_i = \) number of linguistic values for input \( i \)
- \( p_i = \) number of inputs that have the same number of linguistic values \( m_i \)

For example, suppose that FLC has four inputs \((n = 4)\) with input 1 \((n_1)\) has seven linguistic variables \((m_1 = 7)\). Input 2 \((n_2)\) has five linguistic values \((m_2 = 5)\). Input 3 and input 4 \((n_3, n_4 = 3)\) each have nine linguistic values \((m_3 = m_4 = 9)\). Then, by using equation (3.63) the total number of rules in the rule-base \((N_R)\) is calculated as:

\[ N_R = \prod_{i=1}^{n} m_i^{p_i} = m_{1}^{p_1} m_{2}^{p_2} m_{3}^{p_3} = 7^1 \cdot 5^1 \cdot 9^2 = 2835 \text{ rules} \]

Membership functions of the inputs and outputs are usually defined numerically by vectors of dimensions determined by the cardinality of the discretized universe of discourse. Equidistant or nonequidistant discretization of the universe of discourse can be applied in order to find a reasonable compromise between the quality of approximation and the storage requirements. Also, the speed of calculation
Membership functions can be expressed in a functional form, as was discussed in Chapter II, i.e., one can express the linguistic value positive big (PB) as a continuous function that satisfies the membership function requirements and covers the intended range for linguistic value PB. The functional membership functions are convenient way of presenting the input's linguistic values. They simplify the calculation of the fuzzification process, i.e., matching and finding the degree of membership of the rules. So if given crisp or fuzzy inputs, and the membership functions for different linguistic values are expressed as $\mu_j(x^*)$, $\mu_k(x^*_k)$, where $\mu_j(x)$ is expressed analytically then the degree of membership for the rule can be found using product operator as shown in Equation (3.64):

$$\alpha_i = \mu_j(x^*_j) \cdot \mu_k(x^*_k)$$ (3.64)

Other $t$-norms operator can be used equally. The membership functions of the output should be in vector form, due to the fact that finding the output of each of the rules, aggregating the individual rule outputs, and the defuzzification process, apply pointwise to each element of the output universe of discourse.

The process of formation of the input/output fuzzy sets associated with linguistic values can be simplified if normalized universes with the same membership functions are considered. If functional-type membership functions are used, the normalization then involves the changes in their parameters, not the changes in
Construction of the Knowledge Base

The construction of the rule-base is the most difficult aspect of the fuzzy logic controller design. Two approaches that are discussed here can be used. The first approach is based on intuition, knowledge, and experience. Different sources of knowledge, resulting in formulation of the rule-bases, can be conducted. One source is the knowledge based on the experience of an operator controlling a given process. Usually it is difficult to extract control skills from the operator in a form that can be useful for construction of the rule-base of the FLC. Most FLCs combine an approach based on the operator’s experience and knowledge with a good understanding of systems and control theory.

The second approach of the construction of the rule-base of the FLC is based on use of the general concept that is formulated into a knowledge rule-base for specific types of fuzzy logic controllers and can be used as a template for that type. For example, if a PI-type fuzzy controller is used, then the error \(e\), change of error \(\Delta e\), and change of the control action \(\Delta u\) can be formulated into generalized fuzzy rules of the form: If the error and change of the error are zero, then the controller output should stay the same. If the error is approaching zero at a rate that does not warrant changes in the control action, then the control action should also stay the same. If the error is not approaching zero, and either going further away or oscillating around a specific value, then the control action \(\Delta u(t)\) is not zero and depends on the sign and
magnitude of $e(t)$ and $\Delta e(t)$. A well-known template-rule-base is the Macvicar-Whelan rule-base. The fuzzy associate memory (FAM) table of this template is shown in Table 2. The basic steps that are recommended in the design process of this type of FLC are as follows:

**Step 1:** Determine the input and output variables of the fuzzy logic controller.

Appropriate selection of the input and output variables determine the type of FLC.

**Step 2:** Define the parameters of FLC, i.e., define the scaling factors and the normalized universes of discourse if possible, and membership functions of the input/output linguistic values.

Table 2

Fuzzy Associate Memory (FAM) of the Mac-Whelan Template

<table>
<thead>
<tr>
<th>$\begin{array}{l} \downarrow e \ \downarrow De \end{array}$</th>
<th>NB</th>
<th>NM</th>
<th>NS</th>
<th>Z</th>
<th>PS</th>
<th>PM</th>
<th>PB</th>
</tr>
</thead>
<tbody>
<tr>
<td>NB</td>
<td>NB</td>
<td>NB</td>
<td>NB</td>
<td>NB</td>
<td>NM</td>
<td>NS</td>
<td>Z</td>
</tr>
<tr>
<td>NM</td>
<td>NB</td>
<td>NB</td>
<td>NM</td>
<td>NM</td>
<td>NS</td>
<td>Z</td>
<td>PS</td>
</tr>
<tr>
<td>NS</td>
<td>NB</td>
<td>NM</td>
<td>NS</td>
<td>NS</td>
<td>Z</td>
<td>PS</td>
<td>PM</td>
</tr>
<tr>
<td>Z</td>
<td>NM</td>
<td>NS</td>
<td>Z</td>
<td>PS</td>
<td>PM</td>
<td>PM</td>
<td>PB</td>
</tr>
<tr>
<td>PS</td>
<td>NM</td>
<td>NS</td>
<td>Z</td>
<td>PS</td>
<td>PM</td>
<td>PB</td>
<td>PB</td>
</tr>
<tr>
<td>PM</td>
<td>NS</td>
<td>Z</td>
<td>PS</td>
<td>PM</td>
<td>PM</td>
<td>PB</td>
<td>PB</td>
</tr>
<tr>
<td>PB</td>
<td>Z</td>
<td>PS</td>
<td>PM</td>
<td>PB</td>
<td>PB</td>
<td>PB</td>
<td>PB</td>
</tr>
</tbody>
</table>

**Step 3:** Determine the rule-base of the FLC. A good starting point, if knowledge is not available, is to use some rule-base template or to derive the rule intuitively or by using other methods that were discussed in the previous sections.

Table 2 is a rule table and can be used as a design template for a PD, PI, PID type fuzzy logic controller, More discussion on this can be found in Chapters IV and VII.
Step 4: Simulate and test the performance of the designed FLC. In the case that the performance of the designed FLC is not satisfactory, tuning the fuzzy logic control is then required. Tuning the FLC is a more difficult and sophisticated procedure than with a conventional controller. The reason for this is that the FLC is an extremely flexible system, whose behavior is determined by a large number of parameters defining the membership functions and its inference mechanism. The scaling factors are the main parameters used for tuning the FLC. This is because changing the scaling factor changes the normalized universes of discourse, the domains of the membership functions of the fuzzy values of input/output variables of the FLC. The scaling factors have a dramatic effect on the performance of the controlled object; the scaling of the error and the error change is effectively a transformation of the input universes of discourse to a wider or narrower range.

Guidelines developed by Mamdani for tuning scaling factors for regulatory type fuzzy logic controllers of the types discussed in this section are listed as:

1. Higher values of error scaling factor result in good response of the system, that is, low steady state error and low rise time with large value of overshoot and, therefore, a decreases in the stability of the system.

2. Low values of the error-scaling factor result in poor response.

3. The system's fast response and stability are bounded by the high values of the scaling factor for the error and the change of error and the low values of the change of error, i.e.,
\[ k_{e_{\text{min}}} \leq k_e \leq k_{e_{\text{max}}} \]
\[ k_{\dot{e}_{\text{min}}} \leq k_{\dot{e}} \leq k_{\dot{e}_{\text{max}}} \]  

(3.65)

where \( k_e \) and \( k_{\dot{e}} \) are the scaling factor for the error and the error change, respectively.

4. Low value of the control actions change scaling factors which in turns increases the rise time and the integral square error.

Tuning scaling factors should be done within the limitations that are imposed on the system input/output variables. Compromise and adaptive schemes for scaling factors tuning techniques can be utilized. One way to do this can be in the implementation of nonlinear scaling factors. This can be done through employing the adaptive tuning techniques.

This concludes the background introduction that are relevant and of interest in this dissertation. The materials presented in this chapter cover variety of subjects in which various methods and techniques of designing a fuzzy control system were discussed. The design of the fuzzy controller using different techniques and utilizing the experimental data was discussed in length. The use of knowledge gained from conventional control techniques such as adaptive, PD, PI, PID, supervisory, and others control methods were discussed. The different fuzzification and defuzzification methods as well as the different operators of fuzzy inference mechanism were discussed. Techniques to design a fuzzy controller using different fuzzy clustering methods were described and outlined.
CHAPTER IV

ANALYTICAL FORMULATION OF FUZZY CONTROL LAW

Fuzzy logic control system can be expressed analytically if certain requirements and specifications are met. The analytical expression of the fuzzy logic control law has advantages and limitations. The advantages are that given knowledge about a plant and its design specifications, one can systematically design a fuzzy control law that, in taking the knowledge about the process and its performance specifications into account, can lead to an analytically expressed fuzzy control law. The resulting control law then can be analyzed, if desired, for optimality, stability, and reliability. As was mentioned earlier, the stability proof of the fuzzy control has not yet been established on a solid ground, but some work has been conducted on this issue of fuzzy control design. Wang et al. (1996) developed a design methodology for the stabilization of a class of nonlinear systems using a TSK fuzzy model and parallel distribution compensation. Using this approach, a nonlinear plant is then presented by a TSK type fuzzy model, which can be presented by linear models using local dynamics in different state space regions. Jansen et al. (1995) gave an overview of the existing methods for evaluating and testing the stability of fuzzy control systems. Langari (1991) studied the asymptotic stability of the fuzzy closed loop dynamic system using the parameterization of the membership functions. Buckley (1994) argued that if a system is stable by any definition of stability, then many ways to
design an elementary fuzzy controller for that system exist. The stability of the fuzzy control system will not be considered in this work. For readers who are interested in the stability of fuzzy logic control system, the above-cited references and other references on the stability of a fuzzy control system can be found in the bibliography.

What distinguishes fuzzy logic control from conventional control is that fuzzy control can be designed without a need to mathematically model the plant. This, however, should not be confused by the mathematical need for fuzzy logic control. Fuzzy logic control is based on the fuzzy set theory. In Chapter II, some of the basic mathematical foundations of fuzzy logic were discussed.

The work presented in this chapter presents the theory and the mathematical foundation of two methods of expressing the fuzzy control law in an analytical form. In doing so, a major step toward the generalization of the design process of the fuzzy logic controller can be achieved. This study is just a step, and only a step, toward achieving that objective and should not be considered more than that. Many other factors need to be considered before the generalization of the fuzzy logic control design can be achieved. The work presented in other parts of this dissertation will undoubtedly contribute to understanding the need and the benefits of the generalization of fuzzy controller design process.

In this chapter, the theoretical foundation of the two developed analytical models is presented. The two models developed in this chapter will be used in Chapter V to design two fuzzy logic controllers.
The first model is the parameterization model. In this model, the membership function is represented by left and right parametric functions. Langari (1991) initially developed this model to study the stability of the control system and to show that fuzzy control law can be equivalent to nonlinear control law. In this work, the parameterization of the fuzzy membership function is accomplished by relaxing some of the Langari model restrictions and assumptions. One of the restrictions imposed on the model developed by Langari is that the fuzzy logic controller can be either a PI or a PD type fuzzy logic controller, i.e., a regulatory-type controller. This restricts the application of the Langari model to regulatory-type control systems. The modified model is applicable to other control systems, such as a tracking-type control system. A good example might be in idle speed control, where certain measurements of some variables can be taken that will, in turn, affect other variables in ways that cannot be measured by the error and change or integral of the error. This will be explained later when the idle speed controller is developed.

The other model that is developed here is derived from the characteristics of a class of piecewise continuous functions to represent the membership function in an analytical form. The normalized spline and the normalized basis spline are chosen to develop the functional format of the membership function. Since spline functions signify a class of piecewise polynomial functions passing through a set of points and satisfying continuity properties, this gives them the ability to be generalized as a class of polynomials. Under the spline membership function, the triangular membership function, trapezoidal membership function, and bell-shaped membership function, or
any other membership function types or shapes can be considered as special cases of spline. Both of the developed models of representing the membership function can be used to design an analytical fuzzy control system.

One advantage of doing so is the ability to express the fuzzy control algorithms in such a way that the usual design analysis and specifications can be conducted or incorporated in the design process. Also, the stability of the fuzzy control system can be verified, and a step toward designing an optimal fuzzy control system based on conventional knowledge and mathematical tools can be achieved.

Another important advantage is that when a fuzzy analytical model is obtained from expressing the fuzzification and defuzzification methods in an analytical form, then any of the three fuzzy models (Mamdani, TSK, and SAM) discussed in Chapter II can be used, regardless of their restrictions and limitations.

In addition, automating the design process of the fuzzy logic control can also be achieved through the utilization of this ability of expressing the fuzzy control law analytically or by creating more generalized fuzzy inference engine operators. This can be achieved through well-designed and developed software that is based on adaptive parts in which the designed controller achieves the required or sought specifications.

The two developed models will be utilized in Chapter V to develop a fuzzy logic controller for two important practical applications that are selected because of their great research contributions in two active research fields.
In the next sections, the theory and derivation of the fuzzy control law based on the left and right \((LR)\) parameterization will be discussed. While the derivation and the development of the fuzzy control model utilizing the normalized spline-base membership functions will be discussed in later sections.

**Left and Right \((LR)\) Parameterization of the Membership Functions**

**Introduction**

The personates of the fuzzy controller are based on human intuition and experience, and, it can be built without a mathematical model of the process to be controlled. A fuzzy logic controller is also considered to be a robust controller. However, it has not been proven that fuzzy control systems have these properties in themselves, and there is no systematic procedure for the design of the fuzzy control system. The determination of control rules, membership function, and parameters is a very important and essential step in the design of a fuzzy logic controller, so that the closed-loop system is stable. These requirements and other aspects of the fuzzy logic control system design are difficult to systematically formulate, because a fuzzy logic controller is a nonlinear control system. However, many attempts have been made toward establishing standard control methods for fuzzy logic control systems. The knowledge acquired from conventional control theory, in some cases, was applied toward that goal. Kawaji (1991) utilized knowledge gained from conventional proportional plus derivative controllers to design fuzzy control systems by finding the
membership parameters through approximations between the two corresponding control algorithms. Related research in this area can be seen in Qiao and Mizumoto (1996), who proposed a new fuzzy controller structure called the PID type fuzzy logic controller, which is derived from the characteristics of conventional PID controller. Passino and Yurkovich (1998) developed a mathematical expression of a fuzzy control law by simplifying assumptions of the crisp input and crisp output membership functions. In addition, both the TSK model and SAM model have attempted to compact the fuzzy control law by reducing some mathematical complexity of the conventional Mamdani approach. In fact, both of these models are employed in some way or another in the many attempts to express the fuzzy control system in an analytical form.

A typical fuzzy linguistic control system is shown in Figure 23. The FLC box represents a typical fuzzy controller that was shown in Chapter III.

![Figure 23. Fuzzy Logic Controller in a Control System.](image)

The rule base is used to store knowledge of operation of the process in the form of If-Then conditional rules. A typical rule-base rule can be expressed as:

$$\text{If } e(t) \text{ is } A \text{ and } \dot{e}(t) \text{ is } B, \text{ then } u(t) \text{ is } C$$  \hspace{1cm} (4.1a)
where

\[ e(t) = r(t) - y(t) \]  \hspace{1cm} (4.1b)

In the case where the reference input does not vary with time, i.e., regulator controller with fixed reference value, the change of the error can be expressed as:

\[ \frac{de}{dt} = \dot{e} = -\frac{dy}{dt} \] \hspace{1cm} (4.1c)

The output \( u(t) \) in Equation (4.1a) is the instantaneous value of the controller output, i.e., the process input and the alphabetical symbols \( A, B, \) and \( C \) are generic for linguistic terms such as \( high, low, small, large, \) etc. The fuzzy controller inputs are defined as the linguistic variables; for each such input, a universe of discourse is composed of many subsets of the linguistic terms. For example, if an input of a fuzzy logic controller consists of just the error between the measured process output and the desired output (i.e., regulator type controller), then the error universe of discourse consists of a range that encompasses the extremes of the error values in the negative and positive sense. The range or the domain is then divided into as many fuzzy linguistic terms as needed, each of which constitute a fuzzy number or fuzzy membership function.

To quantify the description of fuzzy subsets, one can use parameterization. In the next section, such an approach is discussed as a first step toward developing the analytical model for the fuzzy control law.
Left and Right (LR) Parameterization

A membership function can be constructed by dividing it into two halves. The first half is called the left half, and it covers elements on the left side of the membership centroid point. The other half covers the elements to the right of the centroid. The left and right sides of the membership functions are represented by a parametric function called the left and right (LR) parameterization functions. Using left and right parameterization (LR), a fuzzy subset \( A \) defined on a universe of discourse \( U \) can be described by LR as follows:

\[
\mu_A(u) = \begin{cases} 
L(u^*) & \text{for } u \leq u_0 \\
R(\bar{u}) & \text{for } u > u_0 
\end{cases}
\] 

where \( u^* \) and \( \bar{u} \) are given by:

\[
\begin{align}
\mu^* &= \frac{u_0 - u}{\alpha} \\
\bar{u} &= \frac{u - u_0}{\beta}
\end{align}
\]

For \( \forall u \in U \), \( u_0 \) is the center of the fuzzy set \( A \), \( \mu_A(u_0) = \text{Max}(u_A(u)) \) for \( \forall u \in A \), and \( \alpha \) and \( \beta \) are parameters that define the fuzziness and the shape of the left and right parametric functions. For a symmetric membership function the values of \( \alpha \) and \( \beta \) are equal and the left and right parametric functions are mirrored, i.e.,
\[ L(u^*) = -R(\overline{u}) . \] The \( LR \) parameterization takes any of the forms shown in Equation (4.3):

\[
R(x) = L(x) = \begin{cases} 
\exp(-|x|^p) \\
\max(0, 1-|x|^p) \\
\frac{1}{1+|x|^p}
\end{cases} \quad p > 0
\] (4.3)

The values that are chosen for the parameters \( \alpha \) and \( \beta \) define the strength and weakness of the meaning of the fuzzy subset \( A \). Reducing the values of the parameters give the linguistic value \( (A) \) a more precise meaning, while increasing them broadens the interpretation of a given term. These factors can be thought of as scaling factors for the universe of discourse of the fuzzy set \( A \). For a choice of 
\[ L(x) = \max(0, 1-|x|^p) \] where \( p = 1 \), the fuzzy set \( A \) can be represented by triangular membership function. The slopes of the respective sides of the membership function are represented by \( \alpha \) and \( \beta \), as shown in Figure 24. The general forms for expressing this in \( LR \) parameterization are:

\[ L(u^*) = \max(0, 1-u^*) \] (4.4a)

\[ R(\overline{u}) = \max(0, 1-\overline{u}) \] (4.4b)

For a given input or output universe of discourse, the collection of the fuzzy subsets that divide it can be denoted by \( A = \{A_j\} \), as shown in Figure 25 (a), \( B = \{B_i\} \).
for the second input, as shown in Figure 25 (b), and $C = \{C_j\}$ for the output, as shown in Figure 26. In order to express the control law analytically, some conditions are imposed on the individual element of the collection, such as:

$$\sum_{j} \mu_{A_j}(u) = 1$$  \hspace{1cm} (4.5)

This condition is an essential one that needs to be imposed on all types of fuzzy logic control systems. It implies that the fuzzy classification is compatible. In terms of the membership value, this means that for a specific input, the total sum of the degree of membership value from all contributing membership functions must add up to 1. The number of rules that can be active at any given time depends greatly on how the rules are expressed and the number of inputs. For example, if the rules are expressed as:

If $x_{i1}(t)$ is $A_j$ and $x_{i2}$ is $B_l$ Then $u(t)$ is $C_{jl}$

then at least one rule and at most four rules are fired at one time. The rules that can be fired at any given time, as shown in Figure 26, are:

$R_{jl}$: If $x_{i1}$ is $A_j$ and $x_{i2}$ is $B_l$, Then $u$ is $C_{jl}$
\[ R_{j-1,i}: \text{If } x_{11} \text{ is } A_{j-1} \text{ and } x_{12} \text{ is } B_i, \text{ Then } u \text{ is } C_{j-1,l} \]

\[ R_{j,l-1}: \text{If } x_{11} \text{ is } A_j \text{ and } x_{12} \text{ is } B_{l-1}, \text{ Then } u \text{ is } C_{j,l-1} \]

\[ R_{j-1,l-1}: \text{If } x_{11} \text{ is } A_{j-1} \text{ and } x_{12} \text{ is } B_{l-1}, \text{ Then } u \text{ is } C_{j-1,l-1} \]

The degree of certainty or the truth-value of Rule 1 can be found using any of the t-norms operators. If the Minimum operator is used then the truth-value of Rule 1 is expressed as:

\[ \mu = \min(\mu_{A_j}(x_{11}), \mu_{B_l}(x_{12})) \]  

(4.6)

While, if the Product operator is used then the truth-value of Rule 1 is expressed as:

\[ \mu = \mu_{A_j}(x_{11}) \cdot \mu_{B_l}(x_{12}) \]  

(4.7)

---

**Figure 25.** Universe of Discourses for Inputs (A), and Output (B), for Δ-Membership Functions.
Figure 26. The Number of Rules That Can Be Fired at Any Given Inputs.

Note that the degree of certainty can also be found by any of the t-norm operators discussed in Chapters II and VII. The outcome of the fuzzy inference engine can be found using Equation (4.8):

$$\mu_c(u) = \max_{\mu_{A_j}} \{ \min(\min(\mu_{A_j}(x_{i1}), \mu_B(x_{i2}), \mu_{C_{j+1}}(u)) \}$$  \hspace{1cm} (4.8)

The Center of Area (COA) also called the centroid or the weighted average defuzzification method can be used to find the crisp output. The COA method in this case is given by:

$$u(t) = \frac{\int u\mu_c(u)}{\int \mu_c(u)}$$  \hspace{1cm} (4.8)

where $C$ is the outcome fuzzy set of the fuzzy inference.

If $C_{j}$ or $C_{j+1}$ have a linear profile as discussed earlier, then the defuzzified value $U_j$ or $U_{j-1}$ is given by:
\[ U_j = \frac{1}{3} (\beta_j - \alpha_j) + \mu_{o_j} \]  

(4.9)

where \( \alpha_j \) and \( \beta_j \) are the inverse of the slope as before, and \( \mu_{o_j} \) is the center value of \( U_j \) for any \( j \)th. For other than linear profile cases of the fuzzy subsets of the universe of discourse, the following is an in-depth review of how to design the analytical control model. It should be noted that the slope of the \( LR \) parameterization functions determines whether the membership function has a linear or nonlinear profile.

Analytical Formulation of a Class of Fuzzy Logic Control Law

The rule-base in the fuzzy logic controller stores the knowledge of operation of the plant to be controlled. The rule is expressed in the If-Then format. If the error is defined as the deviation of the plant output from the desired output, and the rate of the error is taken to indicate the rate at which error is changing with time from its previous value, then using these two inputs as the fuzzy controller inputs for a regulatory type fuzzy controller, the rule can be expressed as:

\[ \text{If } e(t) \text{ is } A_j \text{ and } \dot{e}(t) \text{ is } B_i \text{ Then } u \text{ is } C_{ji} \]

Other forms of expressing a fuzzy rule like the rule above can be stated. For example, one might measure the engine speed and the manifold pressure in designing an idle speed controller to express the rule as:

\[ \text{If } N \text{ is } A_j \text{ and } P_m \text{ is } B_i \text{ Then } \theta \text{ is } C_{ji} \]
Using the left and right parameterization for a general class of fuzzy sets and assuming the product operator for the conjunction and COA for defuzzification, the control law \( u(t) \) can be expressed as:

\[
    u(t) = \frac{\sum_{j} \mu_{jl} U_{jl}}{\sum_{j} \mu_{jl}}
\]

(4.10)

where \( j \) and \( i \) range over the indices of all applicable rules. The \( \mu_{jl} \) is the degree of certainty of the corresponding rule. The degree of certainty, or the truth-value can be found by any of the t-norms operators. \( U_{jl} \) is the center of the area of the output fuzzy subset \( C_{jl} \). The objective then becomes designing a fuzzy control law

\[
    u(t) = FLC(e, \dot{e}) \text{ or, in the modified form, } u(t) = FLC(\bar{x}(t)) ,
\]

where \( \bar{x}(t) \) represents the state variables vector.

**Assumptions and Definitions**

If one denotes the universe of discourse for the first input \( x \) by \( X \), for the second input \( y \) by \( Y \), and for the plant input or the controller output by \( U \), then a collection \( A = \{ A_j \} \) and a collection \( B = \{ B_i \} \), partition \( X \) and \( Y \), respectively. For a fuzzy subset \( A_j, B_i, \) and \( C_{jl} \), a membership function can be constructed like the one shown in Figure 27. Figure 27 shows a membership function of the fuzzy linguistic value \( A_j \) centered at \( x_j \in X \), and characterized by the left and right parametric function that is given by:
Similarly, the linguistic value $B_t$ that is centered at $y_t$, which represents a second input can be expressed using the left and right parameterization as:

$$\mu_{B_t}(Y) = \begin{cases} 
L_t\left(\frac{Y - y_t}{\alpha_t}\right) & \text{if } Y \leq y_t \\
R_t\left(\frac{Y - y_t}{\beta_t}\right) & \text{if } Y > y_t
\end{cases} \quad (4.12)$$

In this model, however, another assumption is that the fuzzy subsets are symmetric on all universes of discourse, so that the above definition can be further simplified, as shown in Equation (4.13):
The same is true for $\mu_{b_t}(Y)$, where $\alpha_t = B_t$. The defuzzified value for a given fuzzy subset $C_{il}$ of the output universe of discourse is $U_{jl}$, where $C = \{C_{jl}\}$ is assumed.

Further assumptions on the membership values for a given fuzzy subset of the collection $A$ and $B$, i.e., $A_j$ and $B_i$, are given in Equation (4.14):

$$\sum \mu_{A_j}(X) = \sum \mu_{B_i}(Y) = 1$$

$$L_{j+1} \left( \frac{x_{j+1} - X}{\alpha_{j+1}} \right) + R_{j} \left( \frac{X - x_j}{\beta_j} \right) = 1$$

$$L_{i+1} \left( \frac{y_{i+1} - Y}{\alpha_{i+1}} \right) + R_{i} \left( \frac{Y - y_i}{\beta_i} \right) = 1$$

$$L_{j+1}(0) = R_{j}(0) = L_{i+1}(0) = R_{i}(0) = 1$$

The assumption in Equation (4.14) states that only the right parametric function has a value for the center of the membership function and that value is equal to 1. This assumption also assumed an implied assumption, which states that the overlapping ratio between the two adjacent membership functions is equal to 0.5.

The constraints in Equation (4.15) are also assumed to provide for sufficient overlaps between the fuzzy sets and for smoother but modifiable membership function slope and shape. Both of the assumptions and constraints listed in Equations (4.14) and (4.15) are related and implied.
If the gain parameters $K_j$ and $K_t$, as a function of the input universes of discourse and the defuzzified value on the output universe of discourse, are defined as shown in Equation (4.16):

$$U_l = K_j X_j + K_i Y_i \quad \text{for all } i, j$$

Then the change in gain between two successive gains for particular fuzzy rule can be defined as shown in Equation (4.17):

$$\Delta K_j = K_{j+1} - K_j$$
$$\Delta K_t = K_{t+1} - K_t$$

The control law can be derived based on the previous assumptions and definitions. The analytical control law based on these assumptions and definitions is obtained as shown in Equation (4.18):

$$u(t) = \mu_{A_j}(X)\mu_{B_l}(Y)U_{jl} + \mu_{A_{j+1}}(X)\mu_{B_l}(Y)U_{(j+1)l} + \mu_{A_{j+1}}(X)\mu_{B_{l+1}}(Y)U_{(j+1)(l+1)}$$
$$+ \mu_{A_j}(X)\mu_{B_{l+1}}(Y)U_{ji(l+1)}$$

$$L_{j+1}\left(\frac{x_{j+1} - x_j}{\alpha_{j+1}}\right) = R_j\left(\frac{x_{j+1} - x_j}{\beta_j}\right) = 0 \quad (4.15)$$
$$L_{j+1}\left(\frac{y_{j+1} - y_j}{\alpha_{j+1}}\right) = R_l\left(\frac{y_{j+1} - y_j}{\beta_l}\right) = 0 \quad (4.17)$$
In terms of the definitions for $K_j$ and $K_l$, and with simple but lengthy algebraic manipulation, the following can be stated:

$$U_{jl} = K_j x_j + K_l y_l$$
$$U_{(j+1)l} = K_j x_{j+1} + K_l y_l = (K_j + \Delta K_j) x_{j+1} + K_l y_l$$
$$= K_j x_j + K_l y_l + \Omega$$ (4.19)

$$U_{(j+1)(l+1)} = K_j x_j + K_l y_l + \Omega + \Gamma$$
$$U_{jl(l+1)} = K_j x_j + K_l y_l + \Gamma$$

where

$$\Delta x_j = x_{j+1} - x_j$$
$$\Delta y_l = y_{l+1} - y_l$$

$$\Omega = \Delta K_j x_j + K_j \Delta x_j + \Delta K_j \Delta x_j$$
$$\Gamma = \Delta K_l y_l + K_l \Delta y_l + \Delta K_l \Delta y_l$$

Another way to express the control law based on Equation (4.19) and Equation (4.20) is shown in Equation (4.21):

$$u(t) = K_j X(t) + \left[ \frac{\Delta K_j}{\Delta x_j} x_{j+1} + F_j (X(t) - x_j) \right] (X(t) - x_j)$$
$$+ K_l Y(t) + \left[ \frac{\Delta K_l}{\Delta y_l} y_{l+1} + F_j^* (Y(t) - y_j) \right] (Y(t) - y_j)$$ (4.21)

where

$$\mu_{X_{j+1}} (X(t)) = \left[ \frac{1}{\Delta x_j} + F_j (X(t) - x_j) \right] (X(t) - x_j)$$
$$\mu_{Y_{l+1}} (Y(t)) = \left[ \frac{1}{\Delta y_l} + F_j^* (Y(t) - y_j) \right] (Y(t) - y_j)$$ (4.22)
The expression for \( u(t) \) in Equation (4.21) is in a form that allows analysis, such as linearity or stability of the control law. The control law in Equation (4.21) contains three terms; one is a linear function of the first and the second inputs

\[ F(X,Y) = K_jX(t) + K_tY(t), \]

and the other two are non-linear functions of the membership function centers of the inputs, \( g \) and \( h \), respectively. The functions \( g \) and \( h \) are given by Equations (4.23) and (4.24), respectively.

\[
\begin{align*}
g(x_j, x_{j+1}, F_j) &= \left[ \frac{\Delta K_j}{\Delta x_j} x_{j+1} + F_j(X(t) - x_j)\Gamma \right][X(t) - x_j] \\
h(y_l, y_{l+1}, F_j^*) &= \left[ \frac{\Delta K_l}{\Delta y_l} y_{l+1} + F_j^*(Y(t) - y_l)\Gamma \right][Y(t) - y_l]
\end{align*}
\]

(4.23)  
(4.24)

Thus, the control law obtained can be thought of as a combination of linear and non-linear functions, which can be expressed using the above definitions as:

\[
u(t) = F(X,Y) + g(x_j, x_{j+1}, F_j) + h(y_l, y_{l+1}, F_j^*)
\]

(4.25)

The developed control law expressed in Equation (4.21) is applicable to both the regulatory type fuzzy logic controller and for other inputs that take on different forms and can be defined on their universes of discourse. Fuzzy logic, in general, is suitable for controlling systems that cannot be controlled by conventional control methods. In most cases, a conventional PID controller with gain scheduling or a supervisory controller can handle the regulatory type control if needed, and in those cases, fuzzy

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
logic can be the last resort, provided that the performance of the conventional controller is satisfactory. In a complex industrial process control situation where conventional controller performance is not up to standard, or in the cases where the process model is unobtainable, then the developed fuzzy logic control law can be used and, if needed, can be extended to cases where there are more than two inputs to the process.

The control law developed above can be implemented by following the following steps:

**Step 1:** Define the inputs and outputs of the process and the controller. If the controller is to be a regulator type fuzzy controller, then the inputs and outputs of the controller can be the error and either the error change (PD Type fuzzy logic controller) or integral of the error (PI-type fuzzy logic controller). The PI-type fuzzy logic controller can be constructed using the above procedure. The above control law, however, is applicable only for two input and one output control systems, but it can be easily extended to include a MIMO control system using the outlined method in addition to the decomposition procedure discussed in Chapter II. For example, for multi-input-multi-output control system, the system can be divided into sets of two-inputs-one-output subsystems, and the control law expressed in Equation (4.21) can be applied to each subsystem.

**Step 2:** Define the output of the fuzzy controller (input to the process) in advance and identify the effective range of the output, so that stable and smooth control can be achieved and output saturation and discontinuity can be avoided.
Step 3: Identify the universes of discourse for the controller inputs. This can be done by any of the methods discussed in Chapters II and III.

Step 4: Specify the membership functions number for each input such that the entire universe of discourse is covered and the membership functions have an acceptable degree of overlapping. In this way, a smooth control surface can be achieved. In the control law stated above, a change in the values of $a$ and $\beta$ affects the shape and the symmetry of the membership functions. Thus, the values of $a$, $\beta$, and $p$ should be selected carefully so that the support, overlap, and width of the membership functions are correctly assigned.

Step 5: Specify the initial parameters and tune the parameters if needed, so that symmetric membership functions, if desired, can result. The placement of membership functions on the universe of discourse also needs to be specified; this can be achieved by choosing the center of each fuzzy subset at which the membership function grade is a maximum. The value at which the degree of the membership function is a maximum can be determined by using either heuristic knowledge or other fuzzy system adaptive schemes, such as fuzzy clustering.

Step 6: Design and test the fuzzy controller using the fuzzy algorithms developed in this model. The fuzzy controller can be designed and then tested either by using commercially developed control software or by writing a computer program specifically for the developed controller. Software that has a graphic user interface (GUI) package can best simulate the fuzzy controller. In this way, the time of the design process can be greatly reduced.
The model developed here is applied to control idle speed for the spark
ignition engine. The model will be discussed in detail in Chapter V. Other methods
to express the membership function analytically can be accomplished by utilizing the
approximation ability of the spline function. A normalized spline-base membership
function has the ability to approximate linear or nonlinear functions. In the second
part of this chapter, the spline-base membership function is discussed and the
theoretical foundation is presented.

Analytical Membership Function Using Linear and Nonlinear Functions

Introduction

Piecewise, continuous, and linear functions are used to represent the
membership function so that an analytical way of expressing the control algorithm
can be accomplished. Some of the membership functions mentioned in Chapter II
can be classified into one of several categories. For example, triangular membership
functions, when expressed mathematically, can be thought of as linear membership
functions, which, in this case, are quadratic spline membership functions. On the
other hand, Gaussian membership functions are nonlinear functions but they can be
approximated by a piecewise linear continuous function, such as a normalized spline-base
membership function. A generalized class of functions that can be either linear
or non-linear is the normalized spline function and normalized spline-base
membership function. The spline families of membership function are a very
generalized case of the membership functions. A triangular, bell-shaped, Gaussian, trapezoidal, or any other shape of membership functions, can be constructed using some form of spline functions, or it can be approximated by a spline-based membership function. The following summarizes the advantages of spline-based membership functions:

1. Fuzzy relations can be represented in either linear or non-linear membership functions, which makes this approach an acceptable alternative.

2. The smoothness that spline-based membership function provides is an important advantage, since the design objective should be focused on the smoothness between control actions that the control surface should provide.

3. Using the spline-based membership function, the rate of change of the degree of membership can be assigned according to the design objective.

4. Other linear and non-linear forms of membership function can be generated using the generalized spline membership function.

There are several types of splines, including the polynomial spline, trigonometric spline, parametric spline, β-spline, and rational spline, among many others. Polynomial splines have several attractive properties among them are:

1. They are relatively smooth, are easy to store, and evaluate on a digital computer.

2. Any continuous function defined on closed interval \([a, b]\) can be approximated by a polynomial spline with arbitrary order \((m)\), provided that a sufficient number of knots is known and allowed.
3. The sign structure and the shape of a polynomial spline can be related to
the sign structure of its coefficients.

In the following section, the basic foundation of developing membership
functions using the generalized spline approximation functions is laid.

Mathematical Foundation

The mathematical foundation of spline and normalized spline-base
membership functions can be based on the characteristics of a class of piecewise
polynomial functions, with spline-base functions as a special case. In this section, the
mathematical foundation of spline-base and normalized spline-base membership
function is presented and discussed.

The definitions and assumptions used in deriving the normalized spline-base
membership function are similar to those used for deriving the interpolating functions
using piecewise linear or nonlinear functions Johnston (1982).

Definition: let $\gamma$ be a partition on the interval $[a, b] \subset \mathbb{R}$, such that Equation
(4.26) holds true

$$
\gamma: \quad a = x_0 < x_1 < \ldots < x_{n+1} = b
$$

(4.26)

then a polynomial spline of order $m$ or degree $m-1$ with nodes at the points of $\gamma$ is a
piecewise polynomial function $\mu(u)$ such that $\mu(u)$ can be represented by a
polynomial of degree $m-1$ or less.
If \( m \) is taken to be 2, 3, and 4, one respectively gets linear, quadratic, and cubic splines. For values of \( m \) greater than 4 \((m > 4)\), applying cubic splines should be considered carefully, since the accuracy provided should be weighted against the computational complexity and the complexity of the system design process. In some situations, a value of \( m \) greater than 4 is needed to obtain a better estimation for the membership function shape. In order for a function to be used as a membership function or to be designated as a fuzzy set the function must satisfy certain requirements. One important and essential requirement is that the membership function should assign a value between 0 and 1 to each and every element in the range or support of the fuzzy set. The normalized spline membership function \( \mu \) of order \( m \) or \( m-1 \) for a fuzzy subset \( A \in [a, b] \) over the partition \( \gamma \) satisfy this requirement by assigning the elements in \([a, b]\) a value in the interval \([0, 1]\) i.e., \( \mu : [a, b] \rightarrow [0, 1] \). This means that \( \mu(u) \) can be expressed as:

\[
\mu(u) = \{\mu_i(u) : \mu_i(u)\}
\]  

(4.27)

The expression in Equation (4.27) means that the normalized spline membership function is a polynomial of degree \( m - 1 \) in the interval \([x_i, x_{i-1}]\) for \( i = 0, 1, 2, \ldots, n \), where \( x_i \) is the number of knots in the interval. In addition, the continuity property of the spline functions can be utilized. The continuity property of the spline function is stated as shown in Equation (4.28):
\[
\frac{d^j}{du^j} \mu_{i-1}(u) = \frac{d^j}{du^j} \mu_i(u)
\]  

for \( u = x_i, \ j=0,1,\ldots, m-1-m, \) and \( i=1,2,\ldots,n, \) where \( m_i \) is the knot multiplicity of \( x_i. \)

The normalized basis spline membership function \( \mu_i^m \) of order \( m \) associated with the knots \( x_i, x_{i+1}, \ldots, x_{i+m} \) in \([a, b]\) is then defined as:

\[
\mu_i^m(u) = \mu(u_{x_i+m-1}) \mu_i^{m-1}(u) + \mu(u_{x_i+m}) \mu_{i+1}^{m-1}(u) \quad \text{for} \ m \geq 2
\]  

(4.29a)

\[
\mu(u_{x_i+m-1}) = \frac{u-x_i}{x_{i+m-1}-x_i}, \quad \mu(u_{x_i+m}) = \frac{x_{i+m}-u}{x_{i+m}-x_{i+1}}
\]  

(4.29b)

When \( m = 1, \) Equation (4.29) is reduced to being a crisp set as shown in Equation (4.30):

\[
\mu_i^1(u) = \begin{cases} 
1 & \text{for } x_i \leq u \leq x_{i+1} \\ 
0 & \text{otherwise}
\end{cases}
\]  

(4.30)

A second order normalized spline-base membership function \( (m=2) \) is expressed as shown in Equation (4.31):

\[
\mu_i^2(u) = \begin{cases} 
\frac{u-x_i}{x_{i+1}-x_i} & \text{for } x_i \leq u \leq x_{i+1} \\
\frac{x_{i+2}-u}{x_{i+2}-x_{i+1}} & \text{for } x_{i+1} \leq u \leq x_{i+2} \\
0 & \text{otherwise}
\end{cases}
\]  

(4.31)

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
In Chapter V, a spline membership function of order 3 is used to represent the inputs and outputs membership functions of the fuzzy control system. The cubic spline-base membership function is given in Equation (4.34a).

The properties of the normalized spline-base membership function of any order satisfy all the requirements of the fuzzy set and can be summarized as follows:

1. Transformation: \( \mu^n: [a, b] \rightarrow [0, 1] \)

2. The support of \( \mu^n(u) \) is \([x_n, x_i-m]\) and the value of \( \mu^n(u) = 0 \) outside the subinterval.

3. Adjacent membership functions only have positive contributions for specific input, i.e., \( \mu^n(u) \), \( \mu^n_{i-1}(u) \), \ldots, \( \mu^n_{i-m+1}(u) \) have positive contributions.

4. The sum of all possible contributions of all membership functions for an input at any given time add up to 1, i.e., \( \sum_{j=i-m+1}^{i} \mu^n_j(u) = 1 \), for \( u \in [x_i, x_{i+1}] \).

5. The spline membership function of order \( m \) in interval \([a, b]\) can be represented by Equation (4.32):

\[
\mu(u) = \sum_{j=i-m+1}^{i} \mu^n_j(u) C_j \quad \text{for } u \in [x_n, x_{i-1}] \quad \text{and } m \leq i \leq n \tag{4.32}
\]

where \( C_j \) are constants to be determined.

The normalized spline-base membership function of order \( m \) can be expressed as:

\[
\mu(u) = \frac{1}{|C|} \sum_{j=i-m+1}^{i} C_j \mu^n_j(u) \tag{4.33a}
\]
where

\[ |C| = \sum_{j=1}^{+m+1} |C_j| \]  \hspace{1cm} (4.33b)

In most fuzzy applications, a spline of order 2 or 3 is enough to represent the system inputs and outputs membership functions. When the output of the process cannot be designed off-line and the knowledge needed to model the system is not sufficient, then fuzzy clustering techniques, such as C-mean clustering, can be used to derive the output membership function and rules of the fuzzy system. Therefore, fuzzy clustering techniques based on the normalized spline-base membership function expression of different orders should be developed.

For limited fuzzy subsets on the universe of discourse for a two input-single output fuzzy control system, the systematic way of designing a fuzzy controller can be greatly simplified. The number of rules in the rule-based for a system with two input and one output, with each input having seven fuzzy membership functions, is 49 rules. The seven membership functions representing the linguistic values for each input variable can be denoted as: Negative Big \((NB)\), Negative Medium \((NM)\), Negative Small \((NS)\), Positive Small \((PS)\), Positive Medium \((PM)\), Positive Big \((PB)\), and Zero \((Z)\). If input 1 is defined on universe of discourse \(U_1\) and input 2 is defined on universe of discourse \(U_2\), then the collections of fuzzy subsets of each input represented here by \(A\) and \(B\) are given by:

\[ A = \{NB, NM, NS, Z, PS, PM, PB\} \]
Caution must be taken here when interpreting the meaning of a fuzzy subset from two different universes of discourse. Although \(NM\) in collection \(A\) and \(B\) looks the same, the value or the range that two fuzzy membership functions takes is very much dependent on their respective universe of discourse. For example, if input 1 represents the speed of a spark ignition engine in rpm, and the range at which the universe of discourse is defined is \([0, 4000]\), then \(NM\) might represent a speed in the range of \([0, 800]\) rpm, while \(NM\) for throttle opening, for example, might be in the range of \([10, 20]\) degrees. Other linguistic values such as \(Low\) (\(L\)), \(Very Low\) (\(VL\)), \(Somewhat Low\) (\(SL\)), etc., can be used instead of the defined linguistic values. For example, if one needs to express something with membership function \(Low\), then \(Very Low\) can be constructed using the linguistic hedges discussed in Chapter II. The use of the normalized spline-base membership function simplifies the application of linguistic hedges. The output universe of discourse can be composed of as many membership functions as needed. These fuzzy subsets can then be assigned a membership function. The membership function assigned to each linguistic value can take any form: linear or non-linear, triangular, trapezoidal, Gaussian, and second-order or cubic spline. This can be done at the initial design process, i.e., one can assign \(NM\) to some membership function, given that some parameters about the membership function are known. For example, if a second-order spline membership function is chosen, then the resulting membership function is triangular shape, so that one needs three parameters to define the linguistic value (\(NM\)).
$NM$ using a triangular membership function can be expressed as $NM = A(a, b, c)$, where $a$, $b$, and $c$ are three knots that define the range and the maximum value for the membership function. In this case, three membership grades for the three elements $a$, $b$, and $c$ are easily determined. Using the linear profile to connect the points according to the range and membership grades coordinates will result in triangular membership function. Appendix C contains a graph showing all types of membership functions that can be obtained or approximated using the normalized spline-base function.

A third order normalized spline-based membership function can be derived from the equation given for the generalized splines function as:

$$
\mu(u) = \begin{cases} 
\frac{\eta_1^3}{4} & x_{j-2} \leq u \leq x_{j-1} \\
\frac{1 + 3\eta_2^2 + 3\eta_2^2 - 3\eta_2^3}{4} & x_{j-1} \leq u \leq x_j \\
\frac{1 + 3\eta_3^2 + 3\eta_3^2 - 3\eta_3^3}{4} & x_j \leq u \leq x_{j+1} \\
\frac{\eta_4^3}{4} & x_{j+1} \leq u \leq x_{j+2} \\
0 & otherwise 
\end{cases}
$$

where
\[ \eta_1 = \frac{u - x_{j-2}}{h}, \quad \eta_2 = \frac{u - x_{j-1}}{h}, \]
\[ \eta_3 = \frac{x_{j+1} - u}{h}, \quad \eta_4 = \frac{x_{j+2} - u}{h} \]
\[ h = x_{j+1} - x_j \]

(4.34b)

where \([x_{j-2}, x_{j-1}]\) interval is equally divided with the difference between any two consecutive knots that is given by \(h\). For example, if the membership function \((NM)\) represents the range of engine speed on the interval \([0, 800]\) rpm, then the interval is divided into knots of equidistanced points. This can be accomplished by dividing the interval of interest into a number of subintervals, depending on the kind of the spline used. For cubic spline, four equal intervals are needed to express the \((NM)\) membership function and five parameters need to be specified. The following parameters are needed to define the overall membership function:

\[ x_{i-2} = 0, \ x_{i-1} = 200, \ x_i = 400, \ x_{i+1} = 600, \ x_{i+2} = 800 \]

The overall membership function that covers the specific interval \([0, 800]\) is then:

\[ \mu_{NM}(u) = \begin{cases} 
\frac{1}{4} \left( \frac{u}{800} \right)^3 & 0 \leq u \leq 200 \\
\frac{1}{4} \left( 1 + \frac{u - 200}{200} \right) + 3 \left( \frac{u - 200}{200} \right)^2 - 3 \left( \frac{u - 200}{200} \right)^3 & 200 \leq u \leq 400 \\
\frac{1}{4} \left( 1 + \frac{600 - u}{200} \right) + 3 \left( \frac{600 - u}{200} \right)^2 - 3 \left( \frac{600 - u}{200} \right)^3 & 400 \leq u \leq 600 \\
\frac{1}{4} \left( \frac{800 - u}{200} \right)^3 & 600 \leq u \leq 800 
\end{cases} \]

Figure 28 shows the linear spline and cubic spline representation of the membership function that denotes the negative medium for the engine speed \((NM)\) on
the universe of discourse for the input 1 (engine speed). The above procedure can be equally applied to each linguistic value that defines the linguistic variable.

![Membership Function for Linguistic Value NM Defining Engine Speed Range of [0, 800].](image)

Figure 28. The Membership Function for Linguistic Value NM Defining Engine Speed Range of [0, 800].

One should keep in mind that the triangular, bell-shaped membership functions are all special cases of the more generalized spline functions. For example, the difference between linear and cubic membership functions representing linguistic value *NM* is shown in Figure 29.

The idea of expressing the linguistic values analytically and in more generalized forms for certain control applications is an important one and should be considered a step toward the generalization of certain operators of the fuzzy inference mechanisms. The initial design processes involve the specification of the
control objective and the control design method. Therefore, the following guidelines are useful to follow:

1. Define the inputs and outputs of the fuzzy controller.

2. Define, if possible, the universes of discourse for each input and output.

3. Choose the number of fuzzy subsets or membership functions that will cover the entire universe of discourse and provide smooth control action. Too many membership functions might not add to the performance of the controller, and too few membership functions might give discontinuous control surface and lead to an unstable system. A common way is to try three, five, or seven membership functions at the initial design stage and either increase or decrease the number depending on the achievement of the desired control objectives. The number of rules increases proportionally as the number of membership functions increases. For example, for a system of two inputs with seven membership functions defined on each input
universe of discourse, the number of rules in the rule base is 49 rules. If 11 rules are used, then a total of 121 rules can be obtained. As the number of rules increases, so do the computation time and the complexity of the system.

4. Choose the degree and order of the normalized spline-based membership functions and express each linguistic value that is defined on each input in its functional spline type membership function so that implementation through the development of computer algorithm can be achieved. The algorithm is easy to develop because the parameters of each linguistic value can be assigned in advance or calculated using adaptive subroutine if online design is desired.

5. Define the fuzzy logic inference system, i.e., the conjunction/disjunction \( t \)-norm/\( t \)-conorm operators, implication operator, and the defuzzification method.

6. Simulate the model for sets of inputs with known output in cases where knowledge about the inputs/outputs can be inferred or anticipated. In cases where knowledge about the system behavior cannot be predicted with certain sets of inputs and outputs, other testing methods, such as experimental verification, should be used. In cases where the inputs to the fuzzy controller are fuzzy sets, one can use fuzzy filtering techniques, which involve designing a simple fuzzy controller to produce outputs in crisp forms that can be applied to the direct fuzzy controller as crisp inputs. In this way, one can simplify the design process and the mathematical formulation.

In concluding this chapter, certain points are worth mentioning about these and any other attempts that have been made to simplify the fuzzy logic system design process. The procedures and methods outlined are only as effective as the designer
who is using them. Not all the models can be applied successfully to every application if the assumptions made throughout the development processes are not taken into consideration. For example, if the analytical model is developed on the assumption that certain defuzzification methods are utilized, then using the model with different defuzzification methods will not guarantee successful application of the model. Also, systems that are specifically designed for certain applications have inherently within their model some kind of assumptions based on the process. Thus, applying the same model for different processes might not be a good idea. In these kinds of cases and in cases like them, one can see the need for generalized fuzzy system parameters. The generalized fuzzy system consists of the generalization and integration of the defuzzification method, implication and aggregation methods, and, as was shown here in the case of spline membership function, fuzzification method. Spline membership function is the generalized method for the fuzzification process. This method, if supported by strong software development tools, can substitute for any other methods in use today. Chapter V discusses fuzzy controllers that are designed based on the models developed in this chapter. The fuzzy control model developed using the normalized spline-base membership functions is verified using the inverted pendulum control problem, while the $LR$ parameterization model is used in an important nonlinear control system represented by the idle speed control problem.
CHAPTER V

PRACTICAL APPLICATIONS OF FUZZY LOGIC CONTROL

Introduction

Fuzzy logic control has gained high popularity with control system designers and engineers primarily because of its simplicity and ease of design. In the introduction of Chapter I, some of the practical applications in which fuzzy logic control has been used successfully were discussed. The practical applications of fuzzy logic control are not limited to certain areas, but they encompass a wide range of applications. From a very complex system, such as a subway train system, to a very simple system, such as a rice cooker, the fuzzy logic control concept has been employed successfully. In this chapter, five practical applications of fuzzy logic control are presented. The first two deal with the idle speed control problem. Two approaches are employed in order to control idle speed for a spark ignition engine. The first is the use of the analytical model developed in Chapter IV using left and right parameterization to express the membership functions to control the idle speed in its basic form, taking into account all the assumptions and definitions used in the derivation of the model, in addition to other simplification assumptions in the process control algorithm. In this application, only the throttle control is used, while a conventional PID controller controls the spark control. In the second approach, fuzzy
clustering and experimental data are used to derive the control law for the idle speed control system using both spark advance and throttle control with different engine operating regions. The concepts and the design of fuzzy logic control system methods using experimental data presented in Chapter III are used extensively in this approach.

The third practical application deals with the use of adaptive fuzzy logic control applications to control automotive cruise control. Since the cruise control is closely related to air/fuel ratio control, the control of air/fuel ratio is also discussed, and a fuzzy logic controller is designed for that purpose. The last practical application is the control of an inverted pendulum. This application is chosen to demonstrate the practical use of the developed analytical expression of the membership function developed in Chapter IV, which was based on the concept of base spline and normalized base spline membership functions format.

A Systematic Approach for Designing a Fuzzy Logic Controller Using Analytical Method: Application to Idle Speed Engine Control

Fuzzy control is based on either expert knowledge or experimental data and, therefore, it possesses intrinsic qualities like robustness and ease of implementation. The mathematical modeling for fuzzy control systems has been attempted, but, until now, many models did not extend beyond the application for which they were developed. A general class of fuzzy linguistic control algorithms that can be formulated analytically and can capture the nonlinear aspect of a given fuzzy control
scheme has been formulated using interpolating functions. The interpolating functions map the process error and its rate or cumulative sum into control action. The systematic approach makes it more desirable for use in control systems that have nonlinear dynamics. The analytical method developed in Chapter IV is employed here to design a controller for an idle speed control system. In the following sections, the design process and the simulation results are presented. In Chapter VI, the experimental verification is presented.

Fuzzy logic controllers (FLC) have been used extensively as an alternative to other control methods, such as PID control, adaptive control, observer-based control, and supervisory control. The FLC is best suited to control dynamic systems that have nonlinearities in their plants. Three methods of designing a fuzzy logic control are often used in one form or another, as was discussed in Chapter II. In the Mamdani fuzzy logic controller (MFLC), the control actions corresponding to particular conditions of the system are described in terms of If-Then rules. The fuzzy controllers in this type are rule-based systems, where fuzzy sets are used for specifying qualitative values of the controller inputs and outputs. The basic idea of such a fuzzy controller is to formulate the control protocol of human operator knowledge into a collection of If-Then rules, in a way that is tractable for computers and suitable for mathematical analysis. For example, a fuzzy control rule can be expressed as:

*If speed error is Low and error rate is Small, then throttle is Normal.*

The engine speed error and its rate in this fuzzy control rule are the inputs to the fuzzy controller. *Low, small, and normal* are linguistic variables represented by
fuzzy sets that divide the universe of discourse for each corresponding input and output. Throttle can be considered as the output of the fuzzy controller. Many other forms of stating fuzzy knowledge and rules can be put into If-Then rules. If the error and integral or rate of the error is known or can be measured, then a PD or PI type fuzzy logic controller can be designed. In that case, the general form of fuzzy If-Then rules can be expressed as:

\[
\text{If } e(t) \text{ is } PB \text{ and } D(e(t)) \text{ is } PS, \text{ Then } u(t) \text{ is } PM
\]

where \(D(e(t))\) represents the derivative or cumulative sum of the error. The rules should be based on the knowledge of the controlled process or based on experimental data obtained by specifying the inputs and observing the outputs of the plant. The design process of such a controller is discussed in Chapter III.

The other two fuzzy logic models that were discussed in Chapter II are the TSK and the Simple Additive Model (SAM). The TSK is conceptually different from the MFLC. The rule-base for the TSK fuzzy logic controller consists of rules of the form

\[
R_j: \text{If } x_1 \text{ is } A_{j,1} \text{ and } x_2 \text{ is } A_{j,2} \text{ and } \ldots \ldots \ldots \text{ and } x_n \text{ is } A_{j,n} \text{ then } u = f_j(x_1, x_2, \ldots \ldots x_n)
\]

where \(f_j\) is a linear function expressed as a constant multiple of the inputs. The difference, as can be seen from the rule, is in the consequent part of the If-Then rules, which, in this case, is expressed by the function \(f_j\). For more detailed discussion of the TSK and SAM fuzzy control models, the reader is referred to Chapter II.
The design process usually involves formulating the rules, which is a part of the knowledge base. In order to provide a smooth interface to the numerical process variable and set points, membership functions defining the linguistic terms are required. A fuzzifier determines the membership degrees of the controller input values in the If part of the rules. The inference mechanism combines the information with the knowledge stored in the rules and determines the outputs of the rule-based system. The output usually is a fuzzy set, although it can be a crisp set or singleton. Defuzzification is needed to calculate the overall value and produce a crisp controller output. The fuzzy controller reasoning mechanism consists of five stages and actions, as stated in Chapter III and briefly listed below:

1. Fuzzification: Finding the membership degrees of the measured values of the antecedent variables.

2. Implication: Finding the firing strength or the degree of fulfillment for the antecedent of each rule using fuzzy logic operator; also called t-norm and t-conorm operators.

3. Inference: The degree of the fulfillment of the antecedent of each rule is used to modify the consequent of that rule.

4. Aggregation: The consequent of all the rules that are fired at any particular values of the antecedent are combined into a single fuzzy set using the Max-Min, Max-Product operator, or any other user defined aggregating operator.
5. Defuzzification: The resulting fuzzy set from the aggregation step is defuzzified using any of the defuzzification methods discussed in Chapter II, to yield a crisp control action value.

Based on the fuzzy controller reasoning mechanism mentioned above, one can see that the STK fuzzy controller model combines the inference, aggregation, and defuzzification actions into a single action that can be expressed analytically as:

\[
U = \frac{\sum_{i=1}^{k} \alpha_i f_i(x_1, x_2, \ldots, x_n)}{\sum_{i=1}^{k} \alpha_i} \quad (5.1)
\]

where

\[
\alpha_i = \min(\mu_{A_{i,1}}(x_1), \mu_{A_{i,2}}(x_2), \ldots, \mu_{A_{i,n}}(x_n)) \quad (5.2)
\]

The reasoning mechanisms of a fuzzy controller with all of its reasoning actions are usually applied in the design process. The choices of universe of discourse and its division into fuzzy sets are dictated by the system. The membership function shape and the defuzzification methods can be chosen from many methods that are available. However, there is no systematic way of designing a fuzzy controller. Previous works used different methods for different systems. Shigeyasu et al. (1991) used knowledge acquired from PD control to design the fuzzy logic controller. A PID type fuzzy controller using a parameter adaptation method to produce a parameter adaptive fuzzy controller was designed by Wu and Mizumoto (1996). Ostergaard...
(1977) expressed the fuzzy sets in a functional form; each fuzzy set is associated with a linguistic label, such as Positive Big (PB), as shown in Equation (5.3):

\[ PB = 1 - e^{-u(x)} \]
\[ u(x) = \left( \frac{0.5}{|1 - x|} \right)^{0.25} \]  \hspace{1cm} (5.3)

In order to systematically set a guideline for the design process, certain defuzzification, aggregation, and membership function shapes are assumed. For example, using the product instead of Max-Min in the aggregation steps and center of area method (COA) for defuzzification process, a functional fuzzy logic control algorithm can be developed. A general functional control algorithm has been attempted (Langari, 1992; Wang, 1997; Passino and Yurkovich, 1998). Some attempts have taken the form of empirically based comparative studies or analytical studies. Langari (1992) developed a nonlinear formulation of fuzzy linguistic control algorithms. In this practical application of fuzzy logic control, the analytical model developed in Chapter IV is applied to the idle speed control problem. The idle speed control problem has been an active research subject lately because of the desire to produce low fuel consumption vehicles and to satisfy the stringent environmental regulation of low pollutant emissions. The model is applied with some modification to fit the problem requirements. The validation of the model is then assessed by comparing the controller performance with other control methods that are used for this type of problems.
A typical PI-type fuzzy based control system is shown in Figure 30.

Figure 30. A PI-Type Fuzzy Based Control System.

The inputs to the fuzzy logic controller in Figure 30 are the error and the cumulative sum of the error or the integral of the error. The error is defined as the difference between a set reference value and the measured value, i.e. \( r(t) - y(t) \).

The output of the fuzzy controller \( u(t) \) is the input to the plant or the process. The fuzzy logic controller control rules can be stated in general form as:

\[
R_{ij}: \text{If } e(t) \text{ is } A_j \text{ and } De(t) \text{ is } B_i \text{ Then } u(t) \text{ is } C_{ij}
\]  

\[(5.4)\]

where \( De(t) \) stand for \( \dot{e} \) in the case of PD-type fuzzy logic controller or \( \int \dot{e} dt \) for the PI-type fuzzy logic controller. The symbols \( A_j, B_i \) are the linguistic terms, such as small, high, short, medium, etc. The linguistic variables are subsets of the universe of discourse of the input variable \( e(t) \) and \( De(t) \), respectively. Figure 31 shows the universe of discourse of the error (\( e \)), its rate and output (\( u \)). The symbols NB, NM,
NS, Z, PS, PM, PB in the figure are the linguistic variables that are given by the subsets $A_j$, and they represent Negative Big, Medium, and Small; Positive Big, Medium, and Small, and Z stands for Zero membership function.

![Linguistic Values Represented by Membership Functions](image)

Figure 31. Linguistic Values Represented by Membership Functions.

The fuzzy set can be constructed using the left and right parameterization (LR), which is given as discussed in Chapter IV by Equation (5.5):

$$
\mu_A(u) = \begin{cases} 
L\left(\frac{u_0 - u}{\alpha}\right) & \text{if } u \leq u_0 \\
R\left(\frac{u - u_0}{\beta}\right) & \text{if } u > u_0
\end{cases}
$$

(5.5)

In most practical applications in which fuzzy logic control is used, the fuzzy sets employed are symmetric fuzzy sets. The left and right parameterization functions

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
for symmetrical fuzzy sets are equal and $\alpha = \beta$. Also, for symmetric fuzzy sets, the left or right parametric function can take any of the forms in Equation (5.6):

$$L(x) = \begin{cases} 
\max(0, 1 - |x|^p) \\
\exp(-|x|^p) \\
1/(1 + |x|^p)
\end{cases}$$

(5.6)

In the case where $L(x) = \max(0, 1 - |x|^p)$ is used, where the Max operator takes on the maximum value in the brackets, the resulting membership function is the well-known triangular membership function. The argument $x$ in this case can take on the values bracketed for the $R$ or $L$ function. The parameters $\alpha$ and $\beta$ are defined as the inverse of the slopes $m_L$ and $m_R$, respectively, as shown in Figure 32, and

$$\mu_{\mathcal{A}(u_0)} = 1.$$

![Figure 32. Left and Right Parameters for Defining the Membership Function. Triangular (Left) and General Shape (Right).](image)

Figure 33 shows the effects of the parameters $\alpha$, $\beta$, and $p$ on the shape and the linearity of the membership function defined by left and right parametric function.
given by:

\[
L(x) = R(x) = \frac{1}{1+|x|^p}
\]  

(5.7)

At certain values of the parameters \(\alpha, \beta,\) and \(p,\) the effect on the membership function becomes very small and can be considered negligible. The value at which the effects of \(p\) become negligible is over 10 \((p > 10),\) provided that the number of membership functions defined on the input or output domain is sufficient and the parameters \(\alpha\) and \(\beta\) are properly chosen.

![Figure 33. The Effects of the Membership Function Parameters \(\alpha, \beta,\) and \(p\) on the Shape and Width of the Membership Function.](image)

When the number of membership functions is small and the universe of discourse is large, then the effects of \(\alpha, \beta\) are very strong and should be utilized instead of the parameter \(p.\) Values of \(\alpha, \beta\) below 1.5 result in a triangular membership function, regardless of the value of the parameter \(p.\) The values of \(\alpha, \beta\)
are proportional to the number of the membership functions specified on the universe of discourse. If the number of the membership functions increases, the values of $\alpha$, $\beta$ should also be increased to provide for the needed overlap between the two adjacent membership functions. Higher values of $\alpha$, $\beta$ yield more overlap.

In the design process, the values of the parameters should be chosen carefully and, depending on the desired shape, overlap, number of membership functions, and the range of the universe of discourse, a compromise value can be found that will satisfy the design requirements. Since the fuzzification step is a very important step in the design process, a simple computer program can be used to find the best combination of the three parameters that will satisfy the design requirements.

The degree of certainty or the truth-value ($T_v$), or the degree, at which a certain rule is satisfied, is obtained using the product operator as:

$$T_v = \mu_{A_j}(e(t)) \cdot \mu_{B_l}(D_e(t))$$  \hspace{1cm} (5.8)

The crisp control value obtained in the defuzzification process using the center of the area (COA) can be obtained using either Equation (5.9) or Equation (5.10):

$$u(t) = \frac{\int u \mu_{C(u)}}{\int \mu_{C(u)}}$$  \hspace{1cm} (5.9)
where $i$ is the number of rules, $\mu_i$ is the truth-value of the $i$th rule, and $U_i$ is the COA if the $i$th rule was the only fired rule. In the case in which the particular $C_j$ or $C_{j+1}$ had a linear profile, then the defuzzified value is given by Equation (5.11).

$$U_j = \frac{1}{3}(\beta_j - \alpha_j) + u_{0,j}$$

The parameters in equation (5.11) are shown in Figure 34.

![Figure 34. The Parameters in Equation (5.11) for Triangular Membership Function.](image)

The assumptions used in deriving the control law are stated in Chapter IV and repeated here for the development of the membership functions and the control law. The first assumption states that the sum of all the membership values is unity, i.e., if $A = \{A_j\}$ and $B = \{B_i\}$ are collections of fuzzy subsets over the universe of discourse for the error and its change, then for each element in the universe $e \in E$, and $De \in DE$ Equation (5.12) should hold true.
\[ \sum_j \mu_{A_j}(e) = 1 \]
\[ \sum_i \mu_{B_i}(De) = 1 \]  
(5.12)

Also, for \( A_j \in A \) and \( B_i \in B \) and for \( \forall e \in E, \forall De \in DE \) then Equations (5.13a, b), (5.14), and (5.15) should hold true for the error and the error change.

\[ L_j \left[ \frac{E_{j+1} - e}{\alpha_{j+1}} \right] + R_j \left[ \frac{e - E_j}{\beta_j} \right] = 1 \]  
(5.13a)

\[ L_i \left[ \frac{DE_{i+1} - De}{\alpha_{i+1}} \right] + R_i \left[ \frac{De - DE_i}{\beta_i} \right] = 1 \]  
(5.13b)

\[ L_{j+1}(0) = R_j(0) = 1 \]
\[ L_j \left[ \frac{E_{j+1} - E_j}{\alpha_{j+1}} \right] = R_j \left[ \frac{E_{j+1} - E_j}{\beta_j} \right] = 0 \]  
(5.14)

\[ L_{i+1}(0) = R_i(0) = 1 \]
\[ L_i \left[ \frac{DE_{i+1} - DE_i}{\alpha_{i+1}} \right] = R_i \left[ \frac{DE_{i+1} - DE_i}{\beta_i} \right] = 0 \]  
(5.15)

The control law based on the above assumptions is given by Equation (5.16):

\[ u(t) = [(1 + m_j E_j)DK_j(e(t) - E_j) + K_j e(t)] \]
\[ + [(1 + m_i DE_i)DK_i(de(t) - DE_i) + K_i de(t)] \]  
(5.16)

where
The detailed derivation of the control law shown in Equation (5.16) is stated in Chapter IV for more general cases. The derived control law is then used in idle speed control. In the following section, a brief introduction to the idle speed control problem is discussed and the problem statement is presented.

**Idle Speed Control of Spark Ignition Engine**

**Introduction**

The automotive companies are more interested nowadays in producing lower emission and fuel-efficient vehicles due to strict environmental regulations and the desire for model fuel-efficient vehicles. Improved engines design and detailed engine calibrations had successfully maintained the use of the internal combustion engines as the primary power plant for automobiles. The evolution of engine control from mechanical to electronic and from open loop to closed loop is seen in the improvement in both hardware and software. The use of electronics improves sensing accuracy, actuation capability, and flexibility in control law design. Applying
the modern control and estimation theory requires a dynamic engine model that
describes the effects of control variables on the engine outputs. Based on the model, a
microprocessor can be developed and real time implementation can be achieved. The
objective for idle speed control is to maintain the engine idle speed as close to the
desired set idle speed as possible. Load torque disturbances, such as turning on the air
conditioning, power steering, and transmission shift, can cause engine speed to
fluctuate widely around the idle set speed. Aging of automotive parts, variation in
fuel efficiency, emission, and automotive vibrations can also contribute to the
fluctuation of the idle speed set point. In this study, a previously developed fuzzy
analytical control presented in Chapter IV and briefly stated in this chapter is applied
to control the idle speed of a spark ignition engine. An analytical model to develop a
fuzzy control law makes designing a fuzzy logic controller for such a purpose more
attractive than traditional approaches. The experts’ knowledge or the data collected
from on-line experimentation can be used to derive the rule-base fuzzy logic control
law. The advantage of using the analytical model is in the simplification of the
overall design process mentioned earlier. The biggest drawback of using fuzzy logic
controllers is that there is no systematic way of going about designing such a control
law. In the presence of such a systematic analytical method, the steps or the outlines
of the design process can be clearly stated and the design can be developed using the
analytical model. Previous work related to the work presented in this application can
be seen in Boverie et al. (1997), in which experimental engine data were used to
develop the rule-based control system. The obtained results were very comparable to
those obtained using the conventional PID controller and linear quadratic (LQ) optimal controller. A neural network based procedure developed by Feldkamp and Puskorius (1993) was used to train fuzzy systems to control vehicle idle speed; the results were comparable to linear controller schemes and, in some cases, performed better than a linear model based controller. In the following section, the problem statement for the idle speed control is presented, and the analytical model is used to develop the control law.

Idle Speed Control Using the Developed Model

To achieve a good idle speed control, several requirements must be considered. Optimization of the drivability and comfort, as well as the minimization of fuel consumption and pollutant emissions, are factors that can fall into the requirement categories. Stiffening of the rules against pollutant emissions has led to a reduction of the idle speed set point. This reduction, in turn, has great effects on the stability of the engine at that lowered set speed. Engine process at idle speed can be analyzed in terms of inputs and outputs. The input control variables are the fuel injection pulse width, spark advance, and throttle angle, which determine the air mass flow entering the cylinders. The measured outputs are the manifold pressure and engine speed. The idle speed engine controller is usually designed to control simultaneously the amount of air at the inlet of the engine by controlling the opening of the throttle valve and the spark advance, which provide the ignitions to power stroke. In this study, the spark advance part of the control is assumed to be controlled
by a conventional PID controller. The throttle angle opening and closing can be used to control the amount of air mass flow into the cylinders. The fuzzy controller designed here is then the one that controls the air mass flow. The main inputs to the fuzzy controller are the engine speed error \(e\) and the rate of the error \(De\). Figure 35 shows the general structure for the proposed fuzzy controller.

The idle speed controller is switched on, provided that any one of the following conditions occur:

1: The engine speed falls below threshold value, which is a preselected fixed value, i.e., \(N \leq N_0\).

2: The rate of the measured engine speed or the rate of the engine speed error is positive, i.e., \(N(k) > N(k-1)\).

3: The release of the accelerator pedal, i.e., \(N(k) \ll N(k-1)\).

![Figure 35. The Proposed Fuzzy Logic Controller Structure.](image)

The idle speed controller must be applied successfully when the vehicle is either in motion or stationary. If the vehicle is in motion, then the dynamics of the
engine and the vehicle should be considered, while in the stationary state only the engine dynamics need to be considered.

In designing the proposed fuzzy controller, the inputs and outputs are identified on their respective universe of discourse. The universe of discourse for the error is divided into seven fuzzy sets \( A_j, j = 1 \ldots 7 \). Each fuzzy set is given a linguistic value ranging from Negative Big \((NB)\) to Positive Big \((PB)\) with Zero \((Z)\) representing the fuzzy set around or close to the desired set speed. The same is done for the rate of the error \((De)\); only the fuzzy sets in this case are represented by \( B_l, l = 1 \ldots 7 \). The output universe of discourse is divided into fuzzy sets of the form \( C_{j,l} \).

Figures 36 shows the partitions of the universes of discourse mentioned above. In Figure 36 the \( NB \) corresponds to \( A_{-3}, B_{-3}, C_{-3} \), \( NM \) corresponds to \( A_{-2}, B_{-2}, C_{-2} \), \( NS \) corresponds to \( A_{-1}, B_{-1}, C_{-1} \), \( Z \) corresponds to \( A_0, B_0, C_0 \), \( PS \) corresponds to \( A_2, B_2, C_2 \), and \( PB \) corresponds to \( A_3, B_3, C_3 \). The value at which the membership function or a fuzzy set is 1, i.e., \( E_j, DE_j, U_{j,l}, j = l = -3 \ldots 3 \) is represented by \( \beta \).

The control rules are expressed as:

If \( e \) is \( A_j \) and \( De \) is \( B_l \) Then \( u \) is \( C_{j,l} \)

This rule can be expressed in terms of the linguistic values as:

If \( e \) is \( PB \) and \( De \) is \( NM \) Then \( u \) is \( PS \)

At most, four rules can be fired at anytime, i.e., given the error and change of the error, then, at most, four combinations of the error and its change can be obtained to
Figure 36. Throttle Universe of Discourse (Left). Normalized Universe of Discourses for the Inputs Error (e), Change of the Error (De), and Output (u) (Right).

give the control action u. This is because of the overlapping of the membership functions. Figure 37 shows the membership functions and the error input along with the degree of the memberships (α_j, α_{j-1}). The same plot for the error change and the output universe of discourse can be obtained. The rules fired can be expressed by:

If \( e \) is \( A_j \) and \( De \) is \( B_l \) Then \( u \) is \( C_{jl} \)

If \( e \) is \( A_j \) and \( De \) is \( B_{l+1} \) Then \( u \) is \( C_{jl+1} \)

If \( e \) is \( A_{j+1} \) and \( De \) is \( B_l \) Then \( u \) is \( C_{j+1l} \)

If \( e \) is \( A_{j+1} \) and \( De \) is \( B_{l+1} \) Then \( u \) is \( C_{j+1l+1} \)

From data that can be obtained experimentally, one can derive the fuzzy rules. Many methods can be used to derive the fuzzy rules and numbers for the fuzzy logic controller, such as neuro-fuzzy and fuzzy clustering, as was discussed in Chapter III.
Neuro-fuzzy combines knowledge from two powerful data training methods, i.e., neural network and fuzzy clustering and approximation techniques.

Figure 37. The Degree of Membership for Specific Input $e$ for Two Adjacent Sets.

Fuzzy clustering using the gradient method is employed in this work. This method was discussed in more detail in Chapter III. In using fuzzy clustering, it is always easier and more practical to do it using the TSK fuzzy model rather than the MFLC model. The mathematical formulation of TSK makes it more desirable for such application. If a MFLC model is desired, then the training can be accomplished using TSK, and the resulting TSK fuzzy model can be converted into a Mamdani type fuzzy system. This process to derive the fuzzy rules for the analytical model was employed in this work. A conversion algorithm is then used to convert it to MFLC model. A fuzzy associate memory (FAM) table is then created for all of the rules. Table 3 shows the rules structure for the proposed fuzzy controller. The rules are obtained by using If-Then statements such as:

$$\text{If } e \text{ is PS and } De \text{ is NM Then } u \text{ is NS}$$
The selected set speed is 750 rpm. The scaling factors for the error, its change, and the output are determined from the normalization of their respective universes of discourse. The general formula, which is shown in Equation (5.19) is used in the normalization of the universes of discourse for the inputs.

Table 3

Idle Speed Fuzzy Associate Memory (FAM) Table

\[
\begin{align*}
E_N &= K_1 + K_2 E \\
K_1 &= \frac{E_L + E_R}{E_R - E_L} E_{CL} \\
K_2 &= \frac{2}{E_R - E_L} E_{CL}
\end{align*}
\] (5.19)
Equation (5.19) transforms the universe of discourse of the error from the range defined by \([E_L, E_R]\) to the normalized range defined by \([-E_{CL}, E_{CL}]\). The normalized universe of discourse \(E_N\) resulting from this transformation is then defined on the interval \([-E_{CL}, E_{CL}]\). For symmetric universe of discourse with the interval spread evenly between negative and positive values, the transformation is a very simple process and can be expressed as:

\[
K_N = \frac{1}{|E_{\text{max}}|} \tag{5.20}
\]

where \(E_{\text{max}}\) is the maximum value that the input/output takes before saturation. An example is that if the error is fluctuating within the interval \([-500, 500]\), then the scaling factor or the transformation factor \(K_N\) is given by \(K_N = \frac{1}{500}\). Applying \(K_N\) to the universe of discourse transforms it into a normalized universe of discourse with a range of \([-1, 1]\). Normalized universes of discourse for the error, its change, and the output are as shown in Figure 35.

The fuzzy control action based on the developed control law, along with the design procedure outlined, can be found by different methods. Depending on the method chosen and the membership functions being either symmetric or nonsymmetric, two methods that can be used are as follows:

1. For the choice of symmetric membership functions and using Min-Max \(t\)-norm operator, the control action can be calculated using Equation (5.21):
\[ u_{\text{MinMax}} = \frac{\vee [\wedge (\alpha_j, \beta_i, \alpha_{j+1}, \beta_{i+1})] C_{j,l}}{K} + \frac{\wedge (\alpha_{j+1}, \beta_i) C_{(j+1),l} + \wedge (\alpha_j, \beta_{i+1}) C_{j,(i+1)}}{K} \]  
(5.21)

where \( K \) is given in Equation (5.22):

\[ K = \vee \left[ \wedge (\alpha_j, \beta_i) \wedge (\alpha_{j+1}, \beta_{i+1}) \right] + \wedge (\alpha_{j+1}, \beta_i) + \wedge (\alpha_j, \beta_{i+1}) \]  
(5.22)

and the symbols \( \wedge \) and \( \vee \) indicate the Minimum and Maximum operation for the bracketed quantities, respectively.

2. For the choice of nonsymmetric membership functions and Min-Max operator, the control law can be calculated using Equation (5.23):

\[ u_{\text{MinMax}} = \frac{\vee [\wedge (\alpha_j, \beta_i, \alpha_{j+1}, \beta_{i+1})] C_{j,l}}{K_s} + \frac{\wedge (\alpha_{j+1}, \beta_i) C_{(j+1),l}}{K_s} \]  
(5.23)

where \( K_s \) is given by Equation (5.24):

\[ K_s = \vee [\wedge (\alpha_j, \beta_i) \wedge (\alpha_{j+1}, \beta_{i+1})] + \wedge (\alpha_{j+1}, \beta_i) \]  
(5.24)

If the product-Max t-norm operator is used instead of the Min-Max operator, then the control action for the symmetric and nonsymmetric membership functions choices can be calculated by:

1. For symmetric membership functions, the control law can be expressed as shown in Equation (5.25):
\[ u_{\text{prod-max}} = \frac{\bigvee[(\alpha_j \cdot \beta_l);(\alpha_{j+1} \cdot \beta_{l+1})] \cdot C_{jl}}{K_p} \times \frac{\alpha_{j+1} \cdot \beta_j \cdot C_{(j+1)l} + \alpha_j \cdot \beta_{l+1} \cdot C_{j(l+1)}}{K_p} \]  
\tag{5.25}

where \( K_p \) is given in Equation (5.26):

\[ K_p = \bigvee(\alpha_j \cdot \beta_l; \alpha_{j+1} \cdot \beta_{l+1}) + \alpha_{j+1} \cdot \beta_l + \alpha_j \cdot \beta_{l+1} \]  
\tag{5.26}

2. For nonsymmetric cases, the control law can be expressed as shown in Equation (5.27):

\[ u_{\text{prod-max}} = \frac{\bigvee((\alpha_j \cdot \beta_l);(\alpha_{j+1} \cdot \beta_{l+1});(\alpha_j \cdot \beta_{l+1})) \cdot C_{jl} + \alpha_{j+1} \cdot \beta_l \cdot C_{(j+1)l}}{K_{sp}} \]  
\tag{5.27}

\[ K_{sp} = \bigvee(\alpha_j \cdot \beta_l; \alpha_{j+1} \cdot \beta_l; \alpha_j \cdot \beta_{l+1}) + \alpha_j \cdot \beta_{l+1} \]  
\tag{5.28}

where \( K_{sp} \) is given by Equation (5.28) and the symbols \( \bigvee \) and \( \cdot \) designate the product and Max operation, respectively. In this dissertation, where the error and the error change are used as the inputs to the controller and a symmetric membership function is used to represent the linguistic values, in addition to using the product-Max t-norm operator, the control action is calculated using Equation (5.29):

\[ u = \frac{(DE_{l+1} - e) \cdot (E_{j+1} - \dot{e}) \cdot C_{jl} + (e - DE_j) \cdot (E_{j+1} - \dot{e}) \cdot C_{(j+1)l}}{D} \]  
\tag{5.29}

\[ + \frac{(DE_{l+1} - e) \cdot (\dot{e} - E_j) \cdot C_{j(l+1)} + (e - DE_j) \cdot (\dot{e} - E_j) \cdot C_{jl}}{D} \]
where $D$ is given by:

$$D = (DE_{i+1} - DE_{i}) \cdot (E_{j+1} - E_{j})$$  \hspace{1cm} (5.30)$$

In terms of the degree of membership functions and with crisp consequent fuzzy sets, the same control action in Equation (5.30) can be expressed as shown in Equation (5.31):

$$u = (\alpha_j \cdot \beta_i + \alpha_{j+1} \cdot \beta_{i+1}) \cdot C_{jl} + \alpha_{j+1} \cdot \beta_j \cdot C_{(j+1)l} + \alpha_j \cdot \beta_{i+1} \cdot C_{j(l+1)}$$  \hspace{1cm} (5.31)$$

**Simulation**

In order to test the fuzzy controller performance, one needs to either conduct an experiment or simulate the controller algorithm by writing a computer program or using commercially available control system software. For simulation purposes, a mathematical model for the different subsystems of the controlled process is needed. In the case of idle speed, throttle, intake manifold, air/fuel flow dynamics, compression stroke, and torque generation and acceleration, subsystems are needed to be mathematically modeled. It is worth noting that an embedded fuzzy logic controller is used to generate the amount of air mass rate into the manifold. This fuzzy controller is designed using experimental data that express the air mass flow rate passing the throttle plate as a function of the throttle angle and engine speed. Table 4 is a partial listing of the data that map the throttle angle against engine speed. The entries to Table 4 are the air mass flow rate in g/s.
The control surface for the air mass flow rate is shown in Figure 38. The controller is designed using fuzzy c-mean clustering to derive the rules and the membership functions for the controller inputs. The output of the controller is expressed in the same format as the TSK fuzzy model, i.e., \( \dot{m}_a = k_a + k_1\theta + k_2N \), where the air mass flow rate is expressed as a function of the throttle opening and engine speed in revolution per minute (rpm).

Table 4

Experimentally Obtained Map for the Air-Mass Flow Rate as a Function of Throttle Angle and Engine Speed

<table>
<thead>
<tr>
<th>( \theta )</th>
<th>800</th>
<th>900</th>
<th>1200</th>
<th>1300</th>
<th>1400</th>
<th>1500</th>
<th>1600</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>0.29843</td>
<td>0.30068</td>
<td>0.25688</td>
<td>0.25392</td>
<td>0.25414</td>
<td>0.24685</td>
<td>0.22137</td>
</tr>
<tr>
<td>45</td>
<td>0.56688</td>
<td>0.55593</td>
<td>0.50484</td>
<td>0.48241</td>
<td>0.45687</td>
<td>0.42678</td>
<td>0.3907</td>
</tr>
<tr>
<td>50</td>
<td>0.56518</td>
<td>0.56281</td>
<td>0.52086</td>
<td>0.49932</td>
<td>0.47346</td>
<td>0.44157</td>
<td>0.40193</td>
</tr>
<tr>
<td>60</td>
<td>0.56598</td>
<td>0.56995</td>
<td>0.53413</td>
<td>0.51351</td>
<td>0.48807</td>
<td>0.4553</td>
<td>0.41269</td>
</tr>
<tr>
<td>65</td>
<td>0.56649</td>
<td>0.57228</td>
<td>0.53846</td>
<td>0.51819</td>
<td>0.49295</td>
<td>0.46002</td>
<td>0.41669</td>
</tr>
<tr>
<td>70</td>
<td>0.5666</td>
<td>0.5739</td>
<td>0.54459</td>
<td>0.5248</td>
<td>0.49943</td>
<td>0.46594</td>
<td>0.4218</td>
</tr>
<tr>
<td>80</td>
<td>0.56652</td>
<td>0.57478</td>
<td>0.54965</td>
<td>0.53024</td>
<td>0.50465</td>
<td>0.47061</td>
<td>0.42584</td>
</tr>
<tr>
<td>85</td>
<td>0.56652</td>
<td>0.57478</td>
<td>0.54965</td>
<td>0.53024</td>
<td>0.50465</td>
<td>0.47061</td>
<td>0.42584</td>
</tr>
<tr>
<td>90</td>
<td>0.56652</td>
<td>0.57478</td>
<td>0.54965</td>
<td>0.53024</td>
<td>0.50465</td>
<td>0.47061</td>
<td>0.42584</td>
</tr>
</tbody>
</table>

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Figure 38. Air-Mass Flow Rate Control Surface Generated Using C-Mean Clustering.

For the intake manifold, the model developed by Grossley and Cook (1991) is used with exhaust gas recirculation (EGR) being neglected. The model is given by:

\[
\frac{dP_m}{dt} = \frac{RT}{V_m} \left( \frac{dm_{ai}}{dt} - \frac{dm_{ao}}{dt} \right)
\]  

(5.32)

where \(P_m\), \(m_{ai}\), \(m_{ao}\), \(V_m\), \(T\), and \(R\), are the manifold pressure, air mass flow into the manifold, air mass flow out of the manifold, manifold volume, temperature, and specific gas constant, respectively. The mass flow rate out of the manifold and into the cylinder as a function of the engine speed and the manifold pressure is given by Equation (5.33):
\[
\frac{dm_{ao}}{dt} = a + bNP_m + cNP_m^2 + dN^2P_m
\]  

(5.33)

where \(a \), \(b \), \(c \), and \(d \) are constants that are experimentally determined. The values used here are \((a = -0.366, b = 0.08979, c = -0.0337, d = 0.0001)\). \(N \) denotes the engine speed in revolutions per minute (rpm). The intake, compression stroke, power stroke, and exhaust stroke in a four-stroke engine are assumed to occur simultaneously at any given time for each cylinder. The separation between the ignition of each successive stroke is 180 degrees, so the ignition occurs once per two-crankshaft revolutions. The engine torque \((T_e)\) is expressed as a function of air mass flow \(m_a\), the air fuel ratio \((\delta)\), the spark advance \((\chi)\), and the engine speed \(N\), as shown in Equation (5.34):

\[
T_e(m_a, \chi, \delta, N) = -1813 + 379.36m_a + \chi(21.91 - 0.85\chi) + \delta(0.26 - 0.0028\delta) \\
+ N(0.027 - 0.000107N) + \chi(0.00048N + 2.55m_a - 0.05\chi m_a)
\]  

(5.34)

The engine acceleration is defined as the difference between the engine torque and the load torque \((T_L)\) as is expressed in Equation (5.35):

\[
J \frac{dN}{dt} = T_e - T_L
\]  

(5.35)

where \(J\) is the engine rotational moment of inertia.
The model is then used to simulate the fuzzy logic controller by inserting it into the closed loop feedback control system. Figure 39 shows the schematic layout for the proposed closed loop control.

Figure 39. Schematic Layout of the Idle Speed Control for Spark Ignition Engine.

Results and Discussion

The simulation for the above controller is then conducted using SIMULINK with fuzzy logic toolbox and Matlab control toolbox. The SIMULINK model is shown in Figure 40.

Figure 40. SIMULINK Simulation Model.
The range of the engine speed is taken to be in the interval of \((600,3000)\) rpm. The air valve opening, expressed in percentages, is chosen to be defined on the interval of \([0, 100\%]\), where the range 0-15% throttle is what is known as closed throttle, 70-90% throttle indicates a wide open throttle, and any value in between is known as the part throttle situation. The spark advance is assumed constant when the engine is idling, and it is chosen to be in the interval of \([0, 45]\) degree Before Top Dead Center (BTDC). A conventional proportional plus derivative plus integral (PID) controller controls the spark advance.

Figure 41 shows the response of the controller for step throttle input. It can be seen that the controller works well in bringing down the speed to set point within a short settling time, a zero steady state error, and minimum overshoot.

![Figure 41. Step Response for Idle Speed Control with Desired Idle Speed Set at 650 rpm.](image)

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Figure 42 shows the controller performance when load torque is applied along with the throttle angle variations. The figure shows how the engine perturbation goes to zero after a short time of the application of the load torque. The load torque applied in this case is 20 N.m at $t = 1$ second and removed at $t = 6$ seconds. The engine error registered is within $\pm 30$ rpm or $\pm 0.76$ mph.

![Figure 42. Engine Speed Error ($N_{\text{error}}$). Throttle Input (Top), Load Torque of 20 N.m. (Middle). Engine Speed Perturbation (Bottom).](image)

Figure 43 shows the system response for the acceleration case, where again we see the controller was able to bring the transition speed after the acceleration pedal is off to a desired idle engine speed. The measured engine speed error is within a range of $\pm 29$ rpm. The first part of Figure 43 shows the acceleration with engine speed error of $-29$ rpm starting at $t = 2$ seconds and varies within that range until it settles down to zero error in a time span of 1 second. The second part shows the
deceleration with an engine speed error starting at 29 rpm and varies in that range until it settles to zero error in a time span of one second.

Figure 43. Engine Speed Error For Acceleration and Deceleration Cases.

Figure 44 shows the throttle command (top figure), the load torque (middle figure), and the response for the throttle command (bottom figure). Again, the controller responded very well to the perturbation, which is represented by the load torque. The load torque in this case can be an electrical or mechanical load, such as turning on the air conditioning or power window. The controller in this situation was able to bring the idle speed error to zero within a very short time, as can be seen in the figure. The time span it took the controller to bring the engine speed error to within accepted steady state error is 1 second. The steady state error after a short settling time of 1 second is brought down to zero as can be seen in Figure 44.
Figure 44. Idle Speed System Response. Load Torque Input (Top) and the System Response (Bottom).

Figure 45 shows the throttle input and the system response when a positive load torque of 20 (N.m) is applied. The initial system response has an engine speed error margin of ± 30 rpm. The settling time for the error to become zero is around 1.5 seconds with a steady state error of zero rpm as can be seen in the figure.

Figure 46 shows the system response for applied load torque along with the desired idle speed. It can be seen that the overshoot and settling time are relatively small and within the accepted limits. Also, Figure 46 shows a zero steady state error. These results compare well with other results obtained by other researchers with a speed error of ±150 rpm in the early stage of the system response. One can see that the fuzzy controller was able to bring down the deviation within a range of ± 50 rpm in the first 2 seconds of the system response. The deviation between the measured
and set speed in Figure 46 is within a small limit, which, in the case of idle speed control, is acceptable.

Figure 45. Idle Speed Response for Throttle Control and Constant Load Torque.

Figure 46. Desired and Measured Engine Speed When Load Is Applied.
Figure 47 shows the engine idle speed control for a random input. The engine speed perturbations are very small and are much better than the ones obtained with other control strategies. The random torque input in this case is limited to the range of [0,20] N.m. The speed error is in this range is [-80, +80] rpm, as shown in the figure. The result obtained in this study compares well with other previously obtained results using conventional and other control techniques, which range ± 150 rpm.

Figure 47. Random Load Disturbance Input and System Response.

Figure 48 is a comparison between the Linear Quadratic Gaussian (LQG) control scheme and the fuzzy logic control method used in this study. Figure 48 clearly shows that the steady state error, the response time, and the overshoot of the fuzzy logic controller compares well with the performance of the LQG control.
method. In fact, the fuzzy controller in this case outperformed the LQG controller as the figure clearly shows.

Figure 48. Comparison Between Linear Quadratic Gaussian (LQG)(+) and Fuzzy Logic Controller (Solid).

This concludes the case study. In this case study, the design of the fuzzy logic controller using analytical methods was implemented. The fuzzy logic controller was then used in the closed loop feedback control system to control idle speed. The results obtained using the fuzzy logic designed in this work are shown to be comparable to conventional based control methods that are presently used or are under investigation.

The idle speed for the different operations of the vehicle was investigated, and the effectiveness of the idle speed controller designed here is shown to be valid under those conditions. The idle speed error was reduced by 70 rpm in comparison to the previously obtained results, which were achieved using conventional control
methods. The design methodology used in this study can be also applied to other engine control systems or vehicle control systems, such as cruise control, air/fuel ratio control, and many other automotive-related applications. This study shows that one can design a fuzzy logic controller systematically if certain information about the plant and the inputs and outputs of such a plant are known. The developed controller is economical in terms of cost and time spent in the design process and expedient in terms of prototyping and real time application.

Spark Ignition Engine Control Using Fuzzy Logic: A Three Phases Control With Combined Control Variables

Sensor technology has given the control engineer more tools to design a robust and more efficient engine management system. The Environmental Protection Agency’s recent strict guidelines for pollution emissions control and consumer desires for more efficient fuel vehicles have led many automotive industries to invest in the research and development of more efficient engine management systems. In recent years, much research has concentrated on developing a control scheme based on controlling the Air-Fuel Ratio (AFR). Won et al. (1998) developed a direct adaptive control method using Gaussian neural networks to compensate for transient fueling dynamics and bias in mass air flow rate measurements into the manifold. Powell et al. (1996) used a closed loop system with three-way catalysts to control air-fuel ratio based on experimentally obtained data. Ault et al. (1982) used a nonlinear least squares parameter identification technique to obtain accurate values for the
model parameters from data collected during normal engine operation. The model is then used to control the air-fuel ratio by taking the fuel injector pulse time and the drive-by-wire throttle as inputs and the air-fuel ratio measured by the oxygen sensor as an output. Powell et al. (1998) used a linear observer to control the engine air-fuel ratio based on measured exhaust oxygen sensor output. Other similar works can be seen in Hamburg and Shulman (1980) and Cook and Powell (1988). Other research was conducted to control the engine idle speed around a predetermined set point using different methods such as PID, Linear Quadratic Regulator (LQR), $\ell_1$ optimal control, and fuzzy logic control. Vachtsevanos et al. (1993) developed a controller to control idle speed using fuzzy control based on partitioning of the state space into rectangle cell groups and the quantization of the states and the variable controls into finite bins. The membership functions were then assigned to the state and controls. Other similar work can be found in Abate and Dosio (1990) and Feldkamp and Puskorius (1993). The present work is based mainly on the application of fuzzy logic schemes to control engine idle speed during different engine operation conditions. This work differs from other previous work in which the control laws are derived from experimentally data using fuzzy clustering techniques, and two fuzzy logic controllers are used, one for spark advance (SA) and one for throttle control. The spark advance and throttle control schemes are defined as a function of engine speed and load. Pervious works concentrated on designing only one fuzzy logic controller, while the other control scheme was developed using conventional control methods,
such as a PID controller. The method employed in this application is referred to as neuro-fuzzy control.

Figure 49 shows the schematic diagram for the SIMULINK control model. The switch over mechanism shown in Figure 49 is based on threshold value for the engine speed determined by the transient conditions.

Figure 49. Proposed Engine Controller Using Two Fuzzy Controllers, One for Controlling the Throttle and One for Spark Advance Control.

Basic Fuzzy Logic Controller

Using fuzzy sets as linguistic terms and fuzzy set operation (Passino and Yurkovich, 1998), the fuzzy logic controller can be designed. A typical FLC structure stated here is again shown in Figure 50.

The control protocol, as was discussed earlier in the previous chapters, is stored in the form of If-Then rules. The membership function is then used to define
the linguistic terms, providing a smooth interface to the numerical process variables and set points.

Figure 50. Basic Structure of the Fuzzy Logic Controller in a Control Application.

The blocks shown in Figures 50 are explained earlier and thus will not be explained here. A relatively new type of fuzzy controller was introduced by Takagi-Sugeno-Kang (TSK) (see Chapter III and the discussion presented in the first application in this chapter). For the TSK fuzzy system, the antecedent part is the same as the conventional rule-based system or Mamdani fuzzy system, as it is often called, while the consequent part is expressed as a function of the inputs. For example, the following can be stated for the TSK fuzzy system:

If $u_1$ is $A_1$ and $u_2$ is $A_2$ and ...and $u_n$ is $A_n$ Then $b_i = G_i(*)$ \hspace{1cm} (5.37)

where

$$b_i = G(*) = a_{i,0} + a_{i,1}(u_1)^2 + \ldots + a_{i,n}(u_n)^2$$ \hspace{1cm} (5.38)

and $a_{i,n}$ are constants multiple of the input variables.

The defuzzification method usually used with the TSK system is the center of
gravity (COG) or the center of area (COA) method. Other defuzzification methods can be equally used, provided that certain limitations and assumptions are incorporated. The TSK fuzzy model is best suited when the fuzzy clustering is needed in the initial stages of the design of the fuzzy control systems.

**Universe of Discourse**

For most industrial applications, the values being measured are analog and can be regarded as continuous within a given range. These values are made discrete by analog to digital conversion to allow them to be input into a digital computer system. This process is called *quantization*, and it discretizes the universe of measurement into a certain number of segments. These segments replace the original analog values and form the universe of discourse for the variable in discrete form rather than in continuous form. The fuzzy set is then defined by assigning a degree of membership values to each generic element of the new discrete universe. The choice of the number of segments influences the accuracy of the control, that is, the number of segments affects the control resolution. For example, if a universe of discourse is quantified for every 10 units of measurement instead of 20 units, then the controller is twice as sensitive to the observed variables. Usually coarse resolutions are used for large errors, and fine resolution for small errors. For a given control resolution ($C_r$) and the range of the universe of measurement $[X_{\text{min}}, X_{\text{max}}]$, the quantization level ($Q_L$) can be determined from Equation (5.39):
A PI-type fuzzy logic control system is shown in Figure 51.

![Figure 51. PI-type Fuzzy Control Schemes Used in Practical Application.](image)

The symbols $e$, $C_e$, and $C_i$ in Figure 51 represent the error, the change of the error, and the change of the process input or the fuzzy controller output, respectively. The fuzzy controller has two inputs and one output (MISO). The inputs are the error and change of the error and they are defined as:

\[
e(t) = r(t) - y(t)
\]
\[
C_e(t) = e(t) - e(t-1)
\]

The output of fuzzy logic controller is the integral of the change of its output, which is also the input to the plant. Chapter III includes an in-depth discussion of this type of control.
Introduction

Recent application of fuzzy logic to engine control has shown some promising results that can be extended toward more efficient control systems. Many control methods used to control air/fuel ratio and idle speed have been studied and good results were reported (Butts et al., 1999). In designing control systems for idle speed or engine management systems, many factors should be taken into consideration. Factors such as the modeling of the system mathematically, the nonlinearities of the overall model obtained for control purposes, the simplicity and ease of implementation of the designed system, and the cost in terms of time and money that is required for the design and implementation of the control system should be considered. On the performance side of the designed fuzzy system, factors such as the conditions at which the control system functions properly are very essential in obtaining an optimal control system. In the case of controlling idle speed or designing an engine management system, design factors such as the modes of operation play an important part in designing optimal control system.

There are five control modes of the engine that should be considered when designing the engine control system. Engine crank and start is the first mode. In this mode, the crank speed is less than 200 rpm. The air/fuel ratio at cold start should increase if the engine coolant temperature is very low to compensate for the lack of fuel evaporation and wall wetting. The control in this mode is a table lookup that
maps the engine coolant temperature and the injection pulse width or fuel mass flow rate. The start sequence consists of rich fueling before a predefined rpm threshold and then reducing fuel after an rpm threshold is exceeded. The ignition timing for a cold engine should be very close to top dead center (TDC). In the warm-up mode, the coolant temperature is less than 200°F. The engine should be kept operating smoothly. The oxygen sensor is activated in this mode so that emission is at minimal when the \( \lambda \)-feedback control is activated. The warm-up mode can be achieved faster if the idle speed is increased and the ignition timing is slightly retarded. In the transient mode, the objective here is to provide a smooth transition from one engine operating condition to another with no stalls or bumps, and also to keep exhaust emissions and fuel consumption to a minimum. This can be achieved by increasing the fuel mixture richness during acceleration and decreasing fuel or using a leaner mixture or fuel shut-off during deceleration. In the full load mode such as climbing a hill, the air/fuel mixture and ignition timing should be controlled for maximum power and to limit the engine and exhaust temperature. For maximum engine torque, the equivalence ratio should be between 0.9 and 0.95. Full load is identified using a throttle valve sensor and usually it is indicated by a wide-open throttle (WOT). The ignition timing at full load is determined from the engine speed and either the throttle sensor or the fuel pulse width modulation. In the idle speed mode, the objectives are:

1. Provide a constant engine speed suitable for engine/exhaust temperature at idle conditions under changing electrical and mechanical load, such as air conditioning, fan, and power windows, etc.
2. Provide the lowest idle speed to achieve low exhaust emission and fuel consumption.

A typical idle speed controller inputs and outputs are depicted in Figure 52.

![Figure 52. Idle Speed Controller Inputs and Outputs.](image)

The idle speed controller, as can be seen in Figure 52, can control the air mass flow into the manifold or the spark advance or both. Controlling the air mass flow entering the intake manifold gives a coarse speed control, while controlling spark advance gives a fine speed control.

In this case study, fuzzy logic is used to design a controller that can be implemented within the electronic control unit as a part of the closed loop control. Previously designed controllers employing throttle control, in which perturbation in the throttle is used to control the deviation of the engine speed from idle speed, employ conventional control techniques. The inputs can vary from one design method to another, depending on the information available for the designer and the ability to mathematically model different subsystems and their associated effects on the overall system. Some experimental work mapped the strategies for spark advance and
throttle to control idle speed for certain conditions. Previous experimental data obtained from literature for spark advance and throttle control are used to design the fuzzy controllers. In the following sections, problem statement and design process are discussed.

**Problem Statement and Design Process**

The purpose of this study is to design a fuzzy control algorithm to control idle speed for automotive engines under different conditions, the first of which is the start-up condition. In this case, the engine operates in an open loop with no feedback from the oxygen sensor. The second is the acceleration stage; in this case, the aim is to keep the idle speed around or close to a set point. The third one is idling control; this case is employed when the engine is in idle state with no load applied. A good example of this is waiting on a light signal. The aim is to keep engine speed at the idle set speed limit. As was shown in Figure 52, throttle and spark advance play an important role in controlling idle speed. The throttle opening and spark advance, however, are limited in range. The range of the throttle angle is limited in the interval \([0, 90]\), i.e., \(0 \leq \theta \leq 90^\circ\). The range of spark advance is limited in the interval \([0, 45^\circ]\), i.e., \(0 \leq \alpha \leq 45^\circ\). The parameters \(\theta, \alpha\) are the throttle and spark advances, respectively. The number of the membership functions and their distribution on the universe of discourse for the input and the output variables are shown in Figure 53. The number of the membership functions is taken to be seven but can be any number, as long as it covers the entire universe of discourse, keeping
in mind though that as the number of membership increases, so do the complexity of
the system and the number of the rules. The shape of the membership function can be
triangular, trapezoidal, Gaussian, or any other shape, as discussed in Chapter II.

Figure 53. Fuzzy Representation of the Throttle (Left) and Spark Advance (Right).

Figure 54 shows the normalized universe of discourse for both throttle and
spark advance. The normalized universe of discourse is obtained using the procedure
discussed in the earlier sections.

Figure 54. Normalized Universe of Discourse for Throttle (Left) and Spark Advance
(Right).
The throttle and spark advance fuzzy logic controllers are obtained from experimental data and graphical information that were collected on a four-cylinder spark ignition engine. The data do not represent actual testing data and are considered for illustrative purposes only. The real experimental data can be obtained easily using a very common approach, such as engine mapping schemes that are currently being utilized by the automotive industries. It should be pointed out that a fuzzy logic controller based on experimental engine mapping should perform better than the one presented here because of inaccurate data representation and different resources from which the data were collected, with their inherent experimental parameters and procedure differences. The throttle controller is obtained from data that map the manifold pressure and engine speed with the throttle angle opening. The throttle is therefore expressed as $\theta(e, \Delta e)$. A fuzzy logic controller that takes the two inputs and produces throttle output is then obtained using TSK and the gradient fuzzy clustering method. The obtained fuzzy model is then converted into the Mamdani model and is represented in a Fuzzy Associative Memory (FAM) table or rules table, such as the one shown in Table 5.

Ignition timing has strong effects on the starting and performance of the engine management systems. Adjusting the ignition timing can help start a cold engine. For example, if the engine is turning over slowly due to reduced battery voltage, then setting the ignition timing close to the top dead center can help a cold engine start smoothly. Advancing the timing during warm-up under part-throttle load can aid drivability. Retarding timing during warm-up under closed-throttle
Table 5
FAM Table for the Throttle as a Function of the Engine Speed Error and Its Change

<table>
<thead>
<tr>
<th>de</th>
<th>PB</th>
<th>PM</th>
<th>PS</th>
<th>Z</th>
<th>NS</th>
<th>NM</th>
<th>NB</th>
</tr>
</thead>
<tbody>
<tr>
<td>NS</td>
<td>Z</td>
<td>NS</td>
<td>NM</td>
<td>NB</td>
<td>NB</td>
<td>NB</td>
<td>NB</td>
</tr>
<tr>
<td>NM</td>
<td>PS</td>
<td>Z</td>
<td>NS</td>
<td>NM</td>
<td>NB</td>
<td>NB</td>
<td>NB</td>
</tr>
<tr>
<td>NS</td>
<td>PM</td>
<td>PS</td>
<td>Z</td>
<td>NS</td>
<td>NM</td>
<td>NB</td>
<td>NB</td>
</tr>
<tr>
<td>Z</td>
<td>PB</td>
<td>PM</td>
<td>PS</td>
<td>Z</td>
<td>NS</td>
<td>NM</td>
<td>NB</td>
</tr>
<tr>
<td>PS</td>
<td>PB</td>
<td>PM</td>
<td>PM</td>
<td>PS</td>
<td>Z</td>
<td>NS</td>
<td>NM</td>
</tr>
<tr>
<td>PM</td>
<td>PB</td>
<td>PB</td>
<td>PB</td>
<td>PM</td>
<td>PS</td>
<td>Z</td>
<td>NS</td>
</tr>
<tr>
<td>PB</td>
<td>PB</td>
<td>PB</td>
<td>PB</td>
<td>PB</td>
<td>PM</td>
<td>PS</td>
<td>Z</td>
</tr>
</tbody>
</table>

deceleration will reduce hydrocarbon emissions. In general, the engine should idle at
the lowest speed at which the engine will still run smoothly enough to satisfy the
driver without stalling. Reducing idle speed to a minimum reduces noise and fuel
consumption. The biggest obstacle to low idle speed is the variation in load on the
engine at idle. At idle, small changes have big impacts. Friction loads change with
temperature; the power required to operate the charging system varies with the
electrical load, such as headlights, air conditioning compressor switching on and off,
and shifting gears from drive to reverse or neutral to drive, all of which increase the
load. The idle speed controller keeps the engine running smoothly at the lowest
possible speed with no stalling under different kind of loads.

A typical ignition timing advance curve for best ignition timing control in
terms of power, fuel economy, emission control, and drivability is shown in Figure
55.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Figure 55. Ignition Timing Advance Curve for Best Performance from Starting to Acceleration.

The spark advance controller is designed from data that express the spark advance as a function of engine speed and injection time. The spark advance is expressed in degrees before top dead center (BTDC).

Figure 56 shows the controller surface for the spark advance controller. The fuzzy control rule table consists of 25 rules, as shown in Table 6.

The proposed control scheme with the mathematical model used to simulate the controller performance is shown in Figure 57.

The mathematical model used in this case study is taken from Butts et al. (1999). The mathematical model used describes the complex engine rotational dynamics, manifold dynamics, and combustion dynamics. The functions and variables shown in Figure 57 are defined by Equation (5.41).
Table 6

Spark Advance Fuzzy Control Rules as a Function of $T_{mj}$ and $N$

<table>
<thead>
<tr>
<th>N</th>
<th>$T_{mj}$</th>
<th>NM</th>
<th>NS</th>
<th>Z</th>
<th>PS</th>
<th>PM</th>
</tr>
</thead>
<tbody>
<tr>
<td>NM</td>
<td>Z</td>
<td>NS</td>
<td>Z</td>
<td>Z</td>
<td>PS</td>
<td></td>
</tr>
<tr>
<td>▼ NS</td>
<td>Z</td>
<td>NS</td>
<td>Z</td>
<td>PS</td>
<td>Z</td>
<td></td>
</tr>
<tr>
<td>Z</td>
<td>PS</td>
<td>PS</td>
<td>PS</td>
<td>Z</td>
<td>PS</td>
<td></td>
</tr>
<tr>
<td>PS</td>
<td>PS</td>
<td>PM</td>
<td>PM</td>
<td>PM</td>
<td>PS</td>
<td></td>
</tr>
<tr>
<td>PM</td>
<td>PS</td>
<td>PS</td>
<td>PM</td>
<td>PS</td>
<td>PM</td>
<td></td>
</tr>
</tbody>
</table>

Figure 56. Spark Advance Control Surface.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Figure 57. The Block Diagram and Simulation Model for Spark Advance and Throttle Controls Scheme.

\[
F(N, P) = -0.998 - 0.00658N + 0.1455P + 3.13e - 4NP + 31 - 6N^2 - 2.32e - 3P^2 + 128e - 5P^3 + 1e - 7NP^2 \quad (g/s)
\]

\[
G(M_r, N, \phi, \alpha) = f(N) + f(M_r) + f(\phi) + f(\alpha) - 0.0068M_r N + 0.51M_r - 0.0445M_r \alpha - 8e - 4\phi, N + 9e - 4N\alpha + 0.1262\phi, \alpha
\]

\[
F(s) = \frac{18.079}{s}
\]

\[
E(s) = \frac{22.717}{s}
\]

\[
f(N) = -38.2365 - 2658N + 0.001N^2
\]

\[
f(M_r) = 22.8457M_r - 1.345M_r^2
\]

\[
f(\phi) = 18.77\phi_r - 0.848\phi_r^2
\]

\[
f(\alpha) = -0.4149\alpha - 0.0348\alpha^2
\]

where

\( P \): Is the manifold pressure (bar)

\( N \): Is engine speed (rpm)

\( \theta \): Is the throttle angle

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
\( \alpha \): Is the spark advance (DBTDC)

\( T_d \): Is the load torque (N.m)

\( \dot{m}_a \): Is the mass flow rate into the manifold.

\( \dot{M} \): Is the mass flow rate out of the manifold and into the cylinder.

\( T_r \): Is the load torque in (N.m) equal load torque and shaft torque

\( T_i \): Is the internally developed torque (N.m)

\( \dot{M}_{fr}, \dot{\varphi}_r \): Are the delayed values of \( \dot{M} \) and \( \varphi \), respectively.

As can be seen in Figure 57, a fuzzy controller to calculate the air mass flow rate as a function of the engine speed and throttle angle is included. The switch over fuzzy controller is not shown in the block diagram for simplicity. The table is constructed so that the error takes on the linguistic values seen at the top of the table, and the error rate is the first column in the table. The table is read as:

\[
\text{If } e \text{ is NS and } D_e \text{ is NB Then } u \text{ is NB}
\]

In the case of switch over control, the rules are in the form:

\[
\text{If speed error rate is PS Then controller is one}
\]

Only three fuzzy values are taken for the switch over control. The table constructed for that is a 3×3-rule table, which totals 9 If-Then rules. The switch over controller need not be a fuzzy controller; any conventional control method can be used. In some cases, a three-way switch can accomplish the same purpose as the switch over fuzzy controller.
Results and Discussion

The previous stated mathematical model, along with the block diagram, is used to simulate the performance of the proposed control scheme. The simulation strategy consists of using throttle alone, throttle and spark, constant throttle with spark control, and throttle control with constant spark advance. The results are compared to other works, such as the linear quadratic integral control (LQI), observer event-based model, and the conventional PID control. It should be noted that the results obtained using the conventional control methods are based mostly on a linear mathematical model of the process. However, the mathematical model used to simulate the controller should not be interpreted as a linear control system of the plant. It can be considered as a black box that the controller is designed to control when certain inputs and outputs of the box are known or can be measured, regardless of the linearity or the nonlinearity of the plant model. The simulation results obtained are depicted graphically, as shown in the figures displayed in this section.

Figure 58 shows the throttle command. Figure 59 shows the engine speed deviation from set idle speed (750 rpm). The spark advance is kept constant and no load torque was applied. The speed deviation or error obtained in this case is 35 rpm as depicted in Figure 59. The response for the throttle step command is as expected and it is similar to other results obtained using conventional control methods, such as a PID controller.
Figure 58. Throttle Command for Throttle Control.

Figure 59. Engine Speed Deviation From Set Idle Speed (750 rpm), While Spark Advance Is Kept Constant and No Load Torque Applied.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Figure 60 shows the throttle command input and the engine idle speed deviation from idle \( (N_e = N_d - N) \) where \( N_d \) is the desired idle speed. Figure 60 shows zero steady state error. The overshoot resulting from the throttle suddenly opening is shown to be very small and can be considered negligible. In this case, the controller responded well to the throttle command, as the figure clearly shows.

Figure 60. The Throttle Control With Spark Advance Kept Constant, and No Load Torque Applied.

Figure 61 shows the combined control of spark and throttle inputs commands. In this situation no load was applied and the engine is at idle. A steady state speed error of ±10 rpm can be seen with negligible overshoot. An engine speed error of ±10 rpm can be considered negligible in situations where the aim of the idle speed

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
controller is to keep the desired idle speed in the range of [750 900] rpm or higher.

The response shown in Figure 61 is for both the spark advance and the throttle control as they changed simultaneously or one at a time.

![Graph showing throttle and spark command with no load applied.](image)

Figure 61. Throttle and Spark Command With No Load Applied.

Figure 62 shows the same conditions as in Figure 61, but with different command values for the throttle and spark. A load torque of -100 rpm is applied. A steady state error of 10 rpm and minimum overshoot can be seen in the speed error plot. The response depicted in Figure 61 also shows that the engine speed error when the applied load torque is ≤ -100 rpm, and when the load is removed, the response is the same as the one obtained in Figure 60.
In Figure 63, the spark was kept constant, which indicates an idling situation; a load torque of -50 rpm was then applied. According to Figure 63, the engine speed error was increased slightly, indicating that in this situation the throttle control alone is not as good as the spark advance control or the throttle and spark advance combined control. A steady state error of 50 rpm can be seen in Figure 63. The results obtained in this situation, however, compare well to the results obtained using conventional control techniques under similar but not identical situations. In the studies that utilized fuzzy logic control in an idle speed control problem, usually one of the control algorithms used is not fuzzy. Some studies used conventional PID controller or any other conventional control methods for the spark advance or the throttle control, while the other is controlled using the fuzzy logic control algorithm.
Figure 63. Idle Speed Error With Throttle Input, Constant Spark, and a Load Torque of -50 rpm.

Figure 64 shows the spark control for a load of -100 rpm at constant throttle and under spark control. The spark advance fuzzy controller performed better than the throttle fuzzy controller, even at a higher load torque. The speed error is ±10 rpm. This result matches very well with the result obtained in Figure 63 for the combined control, provided that the throttle is kept constant at certain intervals of the controller simulation time span. One should keep in mind that even though the performance for the spark advance alone control is better than the throttle alone control, there are situations where each control mode is more desirable to be employed than the other. Therefore, combined control algorithms with criteria for selecting the suitable control mode for certain conditions and circumstances should be employed.
Figure 64. Idle Speed Change From Desired When Spark Control Is Utilized at Constant Throttle and a Load Torque of -100 rpm Is Applied.

In Figure 65, the engine speed error with throttle input and constant spark set at 10° BTDC is shown. The error in engine speed is within a range of ±35 rpm with a very short settling time and an overshoot of 20 rpm.

Figure 66 depicts the throttle command input applied at t = 9 and at t = 20 seconds. This situation depicts an acceleration which starts at time t = 9 seconds and a release at time of t = 20 seconds.
Figure 65. Engine Speed Error With Throttle Command, Spark Advance Set to 10° BTDC, and a Load Torque of -50 rpm Applied at $t = 8$ Seconds.

Figure 66. Throttle Step Input Indicating Acceleration.

Figure 67 shows the response for throttle command input shown in Figure 66.

The spark in this case is kept constant at 10° BTDC. The fuzzy controller performed
very well. The engine speed error is zero with a small overshoot on application of the throttle command.

![Graph showing engine speed deviation from set idle speed for step throttle input and constant spark advance.](image)

Figure 67. Engine Speed Deviation From Set Idle Speed for Step Throttle Input and Constant Spark Advance.

Figure 68 shows the same situation, except that the throttle controller is off and the spark fuzzy controller is turned on. The performance is similar to that of the throttle control, except that the larger overshoot occurred before the throttle command took place.

In this case study, two fuzzy logic controllers, one for spark advance control and one for the throttle command control, were designed using experimental data and fuzzy clustering techniques. The application of each or both depends on the operating conditions of the vehicle. The performance for each controller separately and as a combined control was shown to be comparable to the results obtained using other...
control methods (Feldkamp and Puskorius, 1993; Tennant et al., 1979). In some cases, the performance obtained is better than the results obtained using conventional control techniques. This performance, however, should not be considered an indication that the designed controller is better than the presently in-use conventional controller. The controller presented in this study is built based on certain assumptions, different sources of experimental data, and simplification in the design of the fuzzy control algorithm that made the design possible, especially when no mathematical modeling was used.

This application also shows that a fuzzy logic controller can be designed and implemented with ease, which in turn shows that a complete engine management fuzzy logic control system can be designed. A complete engine management control
system based on fuzzy logic control principles can be of great benefit to the automotive industry because of the characteristics of fuzzy logic system in the area of the design, performance, robustness, and stability. The fuzzy controllers can be designed solely based on the knowledge of the operation of the plant that can be obtained from many different sources. The fuzzy logic controller in most cases is more economical in terms of money and time that can be spared in the design, implementation, modification, simulation, prototyping, and real time application.

Fuzzy Logic Control Applications in Automotive Cruise Control And Spark Ignition Engines Air-Fuel Ratio Control

The introduction of fuzzy control in the automotive industry has given rise to extensive research in this area. The advantages that a fuzzy control system has over other conventional methods can be summarized in its approximation ability, its ability to formulate human knowledge, and its ability to incorporate these into engineering applications. The limitations of conventional controllers, such as plant nonlinearity, plant uncertainty, and the uncertainty in measurements, make fuzzy logic controllers with characteristics such as wide range operating conditions, robustness in performance, economy in terms of time and money, and ease in terms of design, an attractive alternative. In this study, two applications of fuzzy logic control in the area of automotive control systems are presented. The results show that fuzzy control can perform as good as conventional control and, in some cases, surpass it without the hassles of obtaining a mathematical model for the highly nonlinear
control systems such as the two applications discussed in this study. The first application is the automotive cruise control problem. The controller for this system is designed using a Proportional plus Integral plus Derivative (PID) type fuzzy logic controller. The second application is developed for air-fuel ratio control using experimental data and fuzzy clustering techniques.

**Introduction**

First, the automotive cruise control systems. Automotive cruise control systems have been widely used and installed on most cars due to benefits such as fuel economy and comfort. Many factors affect the performance of cruise control. The road grade is one factor that the cruise controller should be able to respond to in a very short time. Also, unexpected factors, such as high wind speed and road surface condition, play an important role in the performance of the controller. The standardization of the hardware, such as sensors and actuators, in addition to the advancement in the controller’s hardware, have made the task of designing and implementing any type of controllers an easy one. Vehicles equipped with cruise control have the controller as a part of the engine control unit (ECU), which takes signals from the engine load and speed in addition to the vehicle speed and usually controls the throttle opening. Conventional control in the form of a PID controller is usually used. Even though the PID controller has performed satisfactorily, some other techniques have been explored, as can be seen in Song (1999) and Ishida et al. (1994). The nonlinear characteristics of the vehicle dynamics and other factors, such
as the different kind of disturbances and their interactions, which must be taken into consideration in the design process, present a formidable task for designing a robust and optimal cruise control system. In this dissertation, an alternative to the conventional design process is employed which, in essence, takes into consideration all of the factors mentioned above without the need to model them mathematically. The proposed controller is a parameter adaptive PID type fuzzy logic controller, which modifies the performance of the controller and guards against performance deterioration due to time and manufacturing tolerances.

The proposed controller is designed based on data obtained from simulating the cruise control model that is found in the literature (Song, 1999; Ishida et al., 1994) and from common knowledge of how cruise control functions. The model will account for large road grade changes and road surface conditions, as well as drag force. The fuzzy rules will be derived using the Takagi-Sugeno-Kang (TSK) type fuzzy logic control through using fuzzy clustering techniques, which are in turn converted to the Mamdani type fuzzy logic controller using a conversion algorithm. An adaptive PID type fuzzy logic controller is then derived from the fuzzy inference engine in the Mamdani type fuzzy inference model. The first goal of this study is to extent the model of the adaptive PID type fuzzy logic controller from that of having crisp consequent part to a more general situation. The developed model in this study is applicable to both crisp and fuzzy consequent parts of the fuzzy control system. The second goal is to use the developed model to design an adaptive PID-type fuzzy logic controller for the cruise control problem. The cruise control problem is chosen
because of its importance. It has been an active research area, which makes the
developed model an important contribution in this regard. No mathematical model
was needed to develop the controller for the cruise control problem. However, a
mathematical model was used to simulate the controller in the simulation part of this
study. The results obtained using the developed model are then compared to the
performance of linear control of PID and Linear Quadratic Regulator (LQR).

Proposed Controller and a Review of Previous Works

The proposed controller is an extension of the PID type fuzzy logic controller
that was developed on the assumption that the consequent part of the If-Then rules in
the rule-base is a crisp, which limits the applicability of that model. The model used
here is a generalized model that can be used when the consequent is either crisp or
fuzzy. A typical cruise control model takes the error between a set speed and the
measured speed of the vehicle as an input to the controller. The output of the
controller, which is the input to the engine, is the throttle opening, which is a function
of load and speed. Figure 69 shows the block diagram of the cruise control system
used in most cars.

Figure 69. Typical Cruise Control Block Diagram.
Previous research in this area can be found in the work conducted by Song (1999). Song proposed longitudinal controllers that are based upon dynamic surface control and fuzzy logic. The combination of using fuzzy logic to modify the gains of the mathematical model based observer control system with dynamic surface control was shown to help reduce chattering and control effort, as well as improve tracking performance in decoupled dynamics systems. Both a first-order nonlinear longitudinal vehicle dynamics model and a brake model were developed with two modes of controls, throttle and brake control. Ishida et al. (1999) studied time delay control (TDC) using identification experiments with frequency response under different driving conditions and road inclination. A first-order system model was then derived and used in simulating a time delay control model.

Previous work into developing an adaptive PID type fuzzy logic controller was conducted by Qiao and Mizumoto (1996). Utilizing knowledge gained from conventional controllers, such as Proportional plus Derivative (PD), Proportional plus Integral (PI), and Proportional plus Integral plus Derivative (PID), a model was developed for a PID type fuzzy logic controller by assuming a crisp consequent of the If-Then rules. In the following section, a brief introduction to adaptive fuzzy logic control and a brief description of the model employed in this study are given.

**Adaptive Fuzzy Logic Control**

If the initial design of a fuzzy logic controller does not perform satisfactorily after a time due to changes in the process dynamics, then adaptive fuzzy logic can be
utilized. In an adaptive fuzzy logic controller, the initial design parameters, such as membership functions, universe of discourse, scaling factors, and control rules, can be modified to accommodate the changes in the process dynamics. The adaptive fuzzy logic controller is often constructed with self-tuning and learning capabilities. The adaptive fuzzy logic controller is capable of generating new fuzzy rules, modifying existing fuzzy sets and the designed ones, and adjusting membership function, operating range, and scaling factors. A typical adaptive fuzzy logic controller is shown in Figure 70. The tuning module shown in the figure shows the fuzzy controller parameters that can be utilized in an adaptive scheme. The choices for each parameter are many. For example, the membership function shape can be adjusted or changed altogether within the fuzzy adaptive control scheme. The performance indices are a function of the error \( (R(t) - Y(t)) \) and they dictate the tuning module function. The most commonly used performance indices are least square error (LSE), process error \( (e) \), and process error change \( (de/dt) \).

![Figure 70. Typical Adaptive Fuzzy Logic Controller.](image-url)
**PID Type Fuzzy Logic Controller**

The fuzzy logic controller can be implemented as a Proportional plus Derivative (PD) type fuzzy logic controller, Proportional plus Integral (PI) type fuzzy logic controller, and Proportional plus Integral plus Derivative (PID) type fuzzy logic controller. A conventional PI controller is known to eliminate the steady state error, and a well designed conventional PID controller yields small overshoot, zero steady state error, and fast rise time. Figure 71 shows the block diagram for PI and PD type fuzzy logic controllers, repeated here from Chapter III with the addition of the adaptive control scheme.

![Block Diagram](image_url)

**Figure 71.** Typical PD Type Fuzzy Control (a) and PI Type Fuzzy Controller (b).
The Symbols $K_c$, $K_d$, and $k_i$ in Figure 71 are the scaling factors. A PID type fuzzy logic controller combines the effects of the PD type and PI type fuzzy logic controllers by connecting them in such a way as shown in Figure 72.

Figure 72. A PID Type Fuzzy Logic Controller (FLC).

The controller output is expressed by:

$$u_c = K_a u + k_i u$$

(5.42)

where $K_a$ and $k_i$ express the emphasis of the derivative and integral component of the PID type fuzzy logic controller. Letting the integration component take on a large value at the early stage of the response and reducing it afterward with time, the damping of the system can be increased; then the system becomes stabilized with fast rise time and short settling time. If Equation (5.42) is expressed in terms of the error, which is the main input to the controller, then Equation (5.42) becomes

$$u_c = (K_a K_e + k_i K_d) e + k_i K_e \int e dt + K_a K_d \frac{de}{dt}$$

(5.43)
From Equation (5.43) it can be inferred that if \( k_i \) is decreased gradually, the integral action is also decreased and the damping of the system is increased, which results in a stable system. A parameter adaptive PID type fuzzy logic controller can then be designed based on this idea. The parameter adaptive fuzzy logic controller is depicted in Figure 73. The idea here is to adjust the parameters, knowing that if the parameter \( k_i \) is decreased, then the integral action decreases and the damping of the system response is increased, which results in stable system. Also, if \( k_i \) is decreased, the proportional component decreases and the system response time increases. If \( k_i \) is decreased and the scaling factor for the change of the error is increased at the same rate as \( k_i \) goes down, then the proportional control strength is unchanged and the system response time is shorter.

![Figure 73. Adaptive PID Type Fuzzy Logic Controller.](image)

The idea here is to find the values of \( k_i \) and \( K_a \) so that the resulting system response is stable and meets design specifications. The observer observes the system output, measures the absolute peak value at each peak time, and sends a signal to the
parameter regulator. The parameter adjustment or the tuning module adjusts the values of $K_d$, $k_i$ and $K_a$ at each peak of the time signal, according to Equation (5.44):

\[
K_d = \frac{K_{do}}{\rho_i} \quad (5.44)
\]

\[
k_i = \rho_i k_{io}
\]

where

$K_{do}$: Is the initial values of $K_d$.

$k_{io}$: Is the initial values of $k_i$.

$\rho_i$: Is the absolute peak value at the peak time $t_i (i = 1,2,...)$.

Cruise Control Problem Statement and Design

To formalize a mathematical model for the cruise control system, one needs to take into consideration many factors. Most, if not all of them, are complex, highly nonlinear, and, in some cases, their mathematical models do not exist. The interactions between vehicle dynamics, which include the fuel, brake, ignition, transmission, and combustion subsystems, are highly complex and nonlinear. The external disturbances, such as road grade/conditions, wind speed/directions, and rolling resistance, make the formulation of a mathematical model a formidable task. If such a model is developed, then it must be linearized around a certain operation region for it to be applied successfully. In this study, an adaptive fuzzy logic controller that does not need a mathematical model was proposed. This can be obtained either by conducting an identification experiment or by using expert

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
knowledge and intuition to formulate the rule-base system and apply a trial-and-error approach in the design process. Both of these approaches are applied in different phases of the design process in this case study. The data correlating load and engine speed are used to correlate the engine speed to the vehicle speed. The error between measured vehicle speed and desired speed is used as the input to the fuzzy controller, and the throttle valve opening is the controller output. The output values in this case are fuzzy sets rather than crisp. The adaptive scheme used is shown in Figure 73 with a supervisory fuzzy controller. It is often the case that the universe of discourse is normalized using the scaling factor (see Chapter II), the normalized universe of discourse is shown in Figure 74. Figure 75, on the other hand, shows the normalized universe of discourse for the error and the error change. Since the output cannot take on negative values, its normalized universe of discourse is in the range of [0, 1], as shown in Figure 75.

![Normalized Universe of Discourse for the Error and Its Change defined on the Range [-1, 1].](image)

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
The error and the error change are defined in Equation (5.45):

\[
e(t) = v_d - v(t) \\
\frac{de}{dt} = e(t) - e(t-1)
\]  

(5.45)

Figure 75. Normalized Universe of Discourse for the Output on [0, 1].

The general expression for the If-Then rules for the adaptive PID type fuzzy logic controller is given by:

\[
If \text{ e is NB and de is PB Then u is PS and } K_a \text{ is Z}
\]  

(5.46)

The rule expressed in Equation (5.46) can be decomposed into two equivalent rules that have the same effect as:

\[
If \text{ e is NB and de is PB Then u is PS}
\]  

(5.46a)

\[
If \text{ e is NB and de is PB Then } K_a \text{ is Z}
\]  

(5.46b)

The fuzzy rules are derived from the experimental data using fuzzy clustering techniques. In order to have the output expressed in a more generalized form, the
fuzzy model obtained using TSK and fuzzy $c$-mean clustering method is converted to
the Mamdani fuzzy mode using a conversion algorithm. The resulting fuzzy rules are
expressed as shown in Equation (5.46). Table 7 is Fuzzy Associate Memory (FAM)
table that expresses the relation between the inputs error and change of the error and
the output throttle. The table consists of $5 \times 5$ entries totaling 25 rules. A similar table
that expresses the relationship between the same inputs as in Table 7 and $K_a$ was
obtained and is shown in Table 8. The table is constructed so that the error takes on
the linguistic values seen at the top of the table, and the change of the error is the first
column in the table. The rules in the table are read as:

If error (E) is PB and change of error (DE) is NB Then throttle (u) or $\theta$ is NS

The highlighted cell in Table 7 shows the above rule. The linguistic variables
Negative Big (NB), Negative Small (NS), Zero (Z), Positive Small (PS), and Positive
Big (PB) stand for fuzzy sets that cover the entire universe of discourse for the inputs
and output.

Table 7

FAM Table for the Error, Change of Error Inputs, and Throttle Output
Table 8

FAM Table for the Error, Change of Error Inputs, and the Output $K_a$

<table>
<thead>
<tr>
<th>DE</th>
<th>PB</th>
<th>PS</th>
<th>Z</th>
<th>NS</th>
<th>NB</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NB</td>
<td>Z</td>
<td>NS</td>
<td>NB</td>
<td>NB</td>
<td>NB</td>
</tr>
<tr>
<td>NS</td>
<td>Z</td>
<td>Z</td>
<td>NS</td>
<td>NS</td>
<td>NB</td>
</tr>
<tr>
<td>Z</td>
<td>PS</td>
<td>PS</td>
<td>Z</td>
<td>Z</td>
<td>NS</td>
</tr>
<tr>
<td>PS</td>
<td>PB</td>
<td>Z</td>
<td>PS</td>
<td>Z</td>
<td>Z</td>
</tr>
<tr>
<td>PB</td>
<td>PB</td>
<td>PB</td>
<td>PS</td>
<td>Z</td>
<td>Z</td>
</tr>
</tbody>
</table>

In order to test the performance of the proposed control scheme, a mathematical model is needed. The mathematical models developed in Song (1999), Ishida et al. (1994), and Hucho (1987) are utilized here for simulation purposes only. The model used in Song (1999) is expressed as:

$$J_{w,e} \frac{d\omega_w}{dt} = T_e - hF_{fr} - M_r$$

$$m \frac{dv}{dt} = F_{tr} - F_d - F_g$$

where

$$J_{w,e} = J_w + \frac{J_e}{R^2}$$

is the effective rotational inertia of the wheels

$h$: The effective wheel radius
$F_a$: The attractive force

$m$: The vehicle mass

$F_d = \alpha v^2 \text{sgn}(v)$: The aerodynamic drag force

$F_g$: The road grade force

$v = h\omega$: The vehicle velocity

$M_r$: The rolling resistance moment (experimentally determined)

Figure 76 shows the simulation block diagram that is used to test the controller performance. Two fuzzy controllers are shown in the figure; the upper one is for the throttle control, while the lower produces the output so that the PID type fuzzy logic controller includes both the derivative as well as the integral component effects. Similar models for the linear conventional PID controller and Linear Quadratic Regulator (LQR) are also simulated for comparison purposes.

Figure 76. SIMULINK Simulation Model Used in Simulation.
Simulation Results and Discussion

Simulation for the proposed controller is carried out using MATLAB software, which includes SIMULINK, fuzzy logic toolbox and control system toolbox. The simulation diagram shown in Figure 76 was used to simulate the proposed fuzzy logic controller. The results obtained were then compared with the linearized model that is used to simulate the linear quadratic regulator (LQR) and the conventional PID controller. The mathematical model used in simulating the developed model is the one mentioned in the earlier sections of this case study with minor modification and values that either are assumed or taken from Hucho (1987).

It is worth noting that the resistance to vehicle motion, which consists of the aerodynamic drag, rolling resistance, climbing or gravitational resistance, and acceleration resistance were included in the mathematical model used in the simulation.

In Figure 77, the PID type fuzzy logic controller step response is shown along with the control output, which, in this case, is the throttle valve opening. As can be seen, the controller has a fast response time, zero steady state error, and relatively short settling time with minimal control effort.

In Figure 78, the vehicle was traveling at cruising speed of 40 (m/s) when the desired speed was set at 45 (m/s) at time equal to 5 seconds. Again, the controller performed as expected.
Figure 77. Step Response of the Parameter Adaptive PID Type Fuzzy Logic Controller (a) and Control Effort (b).

Figure 78. Response of the Parameter Adaptive PID Type Fuzzy Logic Controller When Set Speed Is 45 (m/s) at t = 5 seconds.
Figure 79 shows the controller performance when vehicle motion resistance is included. The resistance from rolling, climbing, and acceleration are assumed to total 30% in the first case and 15% in the second, which means that the vehicle is facing an extra 30% resistance due to this added resistance. The figure shows that the controller performed well with minor differences, even though the 30% added resistance is relatively high compared to 15% or no resistance at all. The only difference that is noticeable is in the controller efforts, which is, of course, higher for higher resistance.

![Graph showing controller performance under different added resistance percentages.](image)

Figure 79. Comparison of the Parameter Adaptive PID Type Fuzzy Logic Controller Under Different Added Resistance Percentages.

Figures 80 and 81 show the comparison in performance and control efforts between three types of controllers. The first one is the parameter adaptive PID type fuzzy logic controller, and the other two are the conventional PID controller and
linear quadratic regulator (LQR). Figure 80 shows that the linear conventional PID controller performed better than the other two. The parameter adaptive PID type fuzzy logic controller performed well with smaller overshoot less settling time and zero steady state error in comparison with the LQR and compared very well with the conventional PID controller. Figure 81, on the other hand, shows that the parameter adaptive PID type fuzzy logic controller has lower control effort than the other two, which means that it has an overall better performance.

Figure 80. Performance Comparison Between LQR/Linear PID Controllers and Proposed Controller.

Figure 81. The Control Efforts Correspond to Figure 80 for the LQR/Linear PID Controllers and the Parameter Adaptive PID Type Fuzzy Logic Controller.
Figures 82 and 83 show the performance and the control effort of each controller as a total resistance of 15% is applied. In this case, the performance of the parameter adaptive PID type fuzzy logic controller is better than the other two kinds of controllers, namely conventional PID /LQR. The LQR performance performed better than the conventional PID controller did in this case. The control effort is also on the side of the parameter adaptive PID type fuzzy logic controller. The controller spent much less effort than the other two which, in automotive language, translates to less fuel and therefore less emissions, smoother ride, and cleaner environment. In this case study, the robustness of the parameter adaptive PID type fuzzy logic controller is apparent in its performance. While the performance of the other two controllers went down by different degrees, the PID type fuzzy logic controller performance improved. Also, it is worth noting that the PID type fuzzy logic controller is global in its application, i.e., with minor or no modification at all, it can be used on any type of vehicle.

![Figure 82](Image)

Figure 82. Performance Comparison of Conventional PID, LQR, and Parameter Adaptive PID Type Fuzzy Logic Controller With a Total Resistance of 15% Applied.
Conventional Fuzzy Logic Controller for Air/Fuel (A/F) Ratio Control

In this practical application, a conventional fuzzy logic controller for air-fuel ratio control is designed. The necessity for more precise engine control systems to meet more demanding government regulations has driven the scientific community to develop dynamic engine models for control design. The developed models are then used to design control systems such as air-fuel ratio control, idle speed control, engine fault detection and compensations, etc. The models include descriptions of intake air dynamics, EGR dynamics, torque production, engine/load inertia, and process delays. Previous research in this area was concerned with different aspects of the models. Fuel film formation on the intake manifold wall with its effects on engine
transient performance, details for an intake manifold with exhaust gas recirculation (EGR), and delay of fuel delivery are some of the concentrations in this area. Both time and event-based domains were employed in previous research. Modern control system and estimation theory, such as PID-based control, observer-based control, and sliding mode-based control, are some of the conventional control methods that have been used in air fuel ratio control. More detailed information of previous work in this area can be found in the bibliography. The studies worth mentioning here include Chang (1993), who utilized a discrete, nonlinear, fuel-injected spark ignition engine model for air-fuel ratio event-based control algorithms. Sekozawa et al. (1992) used an internal state estimation method with conventional air-fuel ratio control methods for fuel injection control. Hendricks and Sorenson (1991) developed algorithms for control of air-fuel ratio, EGR, and spark advance based on a real time driver/vehicle model. Czadzeck and Reid (1980) gave a description of a Ford fuel injection system that explains the control strategy for both open loop and closed loop with the experimental derived relationship of different engine components, such as cooling temperature, load, engine speed, and throttle angle. Ishii et al. (1988) developed and experimentally verified a wide range of air-fuel ratio control system algorithm for single-point injection that uses PID control and a built-in learning control. Noble and Beaumont (1991) also discussed system identification techniques and advance dynamic modeling for a gas engine using urban driving cycles.

In this practical application, a conventional fuzzy logic controller is designed to control the air-fuel ratio. The proposed controller is based on engine mapping data.
The advantages of such a controller include its robustness, insensitivity to sensors and actuators nonlinearities, and the lack of mathematical models that are otherwise needed to model the system's highly nonlinear dynamics. Also, the proposed controller is more economical because fewer sensors and actuators are needed. It will be shown that the performance of the proposed controller surpasses the performance of other mathematical model-based conventional controllers. The data used in deriving the fuzzy logic control rules are obtained from graphical information that is found in Hendricks and Sorenson (1991). Two fuzzy controllers, one for the manifold pressure output with throttle and engine speed inputs and the other for volumetric efficiency output for the same inputs as in the controller one are developed.

Figures 84 and 85 depict the surface control for each fuzzy controller used in this application. The fuzzy rules generated are shown in Tables 9 and 10.

![Figure 84. Fuzzy Generated Control Surface for Manifold Pressure.](image-url)
Figure 85. Fuzzy Generated Control Surface for Volumetric Efficiency.

Table 9

Fuzzy Rules Table for Manifold Pressure

<table>
<thead>
<tr>
<th>N</th>
<th>θ</th>
<th>01</th>
<th>02</th>
<th>03</th>
<th>04</th>
<th>05</th>
<th>06</th>
<th>07</th>
</tr>
</thead>
<tbody>
<tr>
<td>a1</td>
<td>—</td>
<td>P7 (.6)</td>
<td>P7 (.5)</td>
<td>P6 (.7)</td>
<td>P6 (.7)</td>
<td>P6 (.7)</td>
<td>P7 (.7)</td>
<td></td>
</tr>
<tr>
<td>a2</td>
<td>P6 (.5)</td>
<td>P7 (.7)</td>
<td>P6 (.3)</td>
<td>P6 (.8)</td>
<td>P6 (.8)</td>
<td>P7 (.7)</td>
<td>P7 (.7)</td>
<td></td>
</tr>
<tr>
<td>a3</td>
<td>P7 (.8)</td>
<td>P7 (.7)</td>
<td>P6 (.7)</td>
<td>P6 (.8)</td>
<td>P6 (.7)</td>
<td>P5 (.9)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a4</td>
<td>P6 (.6)</td>
<td>P7 (.7)</td>
<td>P7 (.8)</td>
<td>P7 (.7)</td>
<td>P6 (.5)</td>
<td>P5 (.8)</td>
<td>P4 (1)</td>
<td></td>
</tr>
<tr>
<td>a5</td>
<td>P7 (.7)</td>
<td>P7 (.7)</td>
<td>P6 (.8)</td>
<td>P5 (1)</td>
<td>P5 (.9)</td>
<td>P6 (.7)</td>
<td>P7 (.7)</td>
<td></td>
</tr>
<tr>
<td>a6</td>
<td>P7 (.9)</td>
<td>P7 (.8)</td>
<td>P5 (.6)</td>
<td>P5 (.8)</td>
<td>P6 (.7)</td>
<td>P6 (.7)</td>
<td>P7 (.7)</td>
<td></td>
</tr>
<tr>
<td>a7</td>
<td>P7 (.7)</td>
<td>P6 (.7)</td>
<td>P6 (.7)</td>
<td>P5 (.8)</td>
<td>P7 (.8)</td>
<td>P7 (.7)</td>
<td>P7 (.5)</td>
<td></td>
</tr>
</tbody>
</table>

Note. The numbers in parentheses indicate the rules strength.
Table 10
Fuzzy Control Rules for Volumetric Efficiency Control

<table>
<thead>
<tr>
<th>( N )</th>
<th>( \theta )</th>
<th>01</th>
<th>02</th>
<th>03</th>
<th>04</th>
<th>05</th>
</tr>
</thead>
<tbody>
<tr>
<td>n1</td>
<td></td>
<td>Ve5 (.9)</td>
<td>Ve5 (.9)</td>
<td>Ve5 (.8)</td>
<td>Ve4 (1)</td>
<td>Ve4 (.8)</td>
</tr>
<tr>
<td>n2</td>
<td></td>
<td>Ve5 (.9)</td>
<td>Ve5 (.7)</td>
<td>Ve5 (.8)</td>
<td>Ve4 (.8)</td>
<td>Ve3 (9)</td>
</tr>
<tr>
<td>n3</td>
<td></td>
<td>Ve5 (1)</td>
<td>Ve5 (1)</td>
<td>Ve4 (1)</td>
<td>Ve3 (.9)</td>
<td>Ve2 (.9)</td>
</tr>
<tr>
<td>n4</td>
<td></td>
<td>Ve5 (.9)</td>
<td>Ve5 (.9)</td>
<td>Ve5 (1)</td>
<td>Ve5 (1)</td>
<td>Ve5 (1)</td>
</tr>
<tr>
<td>n5</td>
<td></td>
<td>Ve5 (.8)</td>
<td>Ve5 (.8)</td>
<td>Ve4 (1)</td>
<td>Ve4 (1)</td>
<td>Ve5 (1)</td>
</tr>
</tbody>
</table>

Note. The numbers in parentheses indicate the rules strength.

Figure 86 shows the membership functions for the inputs and output of each controller. The controller is simulated using a mathematical model that can be found in all of the references mentioned earlier in one form or another. The manifold pressure is expressed as:

\[
\frac{dP_m}{dt} = \frac{V_d N \eta_v}{2V} P_m - \frac{RT_m}{V} \frac{dm_{at}}{dt} (\theta, P_m) \tag{5.48}
\]

where \( P_m, T_m, N, \eta_v \) are the manifold pressure, manifold temperature, engine rotational speed, and volumetric efficiency, respectively. \( V \) and \( V_d \) are the manifold volume and engine displacement, respectively. \( R \) is the universal gas constant. The
The mathematical model used for the engine torque production is found in Moskwa et al. (1987) and is given by Equation (5.49a,b):

\[
T_e = a + bm_a + c(A/F) + d(A/F)^2 + e\sigma + f\sigma^2 + gN + hN^2 + kN\sigma + l\sigma m_a + q\sigma^2 m_a
\]  

(5.49a)

\[
\frac{dN}{dt} = T_e - T_L
\]  

(5.49b)

where \(a, b, c, d, e, f, g, h, k, l, q\) are constants and \(T_e, T_L, (A/F), \sigma, m_a, J, N\), are engine torque, load torque, air-fuel ratio, spark advance (BTDC), air mass flow into the cylinder, engine rotational moment of inertia, and engine rotational speed, respectively.

Figure 86. Membership Function for Inputs and Output of Each Controller. Manifold Pressure (Upper), Volumetric Efficiency (Lower).

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
The equivalence ratio is given by:

\[ \phi = \frac{(A/F)_s}{(A/F)_a} \]  

which is the ratio of stoichiometric air-fuel ratio to the actual one.

The fuel dynamics model for port fuel injection (PFI) used is the same one used in Chang (1993), and it is expressed as in Equation (5.51):

\[
\begin{align*}
  m_{fp}(k+1) &= (1-f_b)m_{fp}(k) + (1-f_a)m_f(k) \\
  m_f(k) &= f_b m_{fp}(k) + f_a m_{fp}(k)
\end{align*}
\]  

where \( m_{fp}, m_f, f_a, f_b \) are the liquid fuel puddle mass, injection fuel mass, fuel mass entering the cylinder, fraction of the injection fuel that enters the puddle, and fraction of fuel puddle that evaporates and enter the cylinder, respectively.

**Results and Discussion**

The results of the simulation for the proposed control scheme are shown in Figures 88, 89, 90, and 91. In Figure 88, the response to the load torque and throttle input shown in Figure 87 is presented. The simulation in this case is done without compensation for fuel puddle dynamics and part throttle situation, i.e., it is assumed that the fuel injected is the same as that which enters the cylinder, and throttle angle is within the range of 0-40 degrees. The fuzzy logic controller was able to keep the equivalence ratio equal to 1, which is an ideal value in this situation.
Figure 87. Throttle and Load Applied to the System at Time Indicated in the Graphs.

Figure 88. Part Throttle Air-Fuel Ratio Control Response When an Applied Load Torque as Shown in Figure 87 and Fuel Compensation Is Neglected.

In Figure 89, the fuel puddle dynamics were taken into consideration, and the fuzzy logic controllers were able to keep the equivalence ratio very close to 1 under the same throttle and load torque conditions shown in Figure 87. The fuzzy
controllers in this situation performed as good as any conventional control methods, such as observer-based, sliding mode, or PID control methods.

Figure 89. Response for the Same Load and Throttle as Shown in Figure 87 and Fuel Compensation Is Taken Into Consideration With $f_a = f_b = 0.5$.

Figure 90 shows the idle case or closed throttle case with no fuel compensation. Again, the fuzzy controllers performed similar to the part throttle control situation. This indicates the global applicability, robustness, and adaptation of the proposed controllers.

Figure 90. The Response With Same Torque Load as in Figure 87, and Throttle as Shown Above; Fuel Compensation Is Neglected.
Figure 91 shows the response under the same conditions as in the closed throttle case mentioned above with consideration of the fuel compensation. The performance is similar to the part throttle case shown in Figure 89, with minor differences. Even though the graphical approximations used to derive the fuzzy logic controllers’ rules and membership functions were not extremely accurate, the fuzzy controllers performed well and surpassed other conventional control methods. This study shows the fuzzy system’s approximation ability in its finest case scenario.

From these simulation results, we see that the fuel injection response under a different throttle/torque and compensated/uncompensated fuel system in terms of injection times and fluctuation is very good. Fluctuation in injection timing for a fuel injection system is highly undesirable.

Figure 91. Part Throttle Air-Fuel Ratio Control Response with Load Torque as in Figure 87, Throttle as Shown in Figure 90, and Fuel Compensation Factors $f_a = f_b = 0.5$. 

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
This application study contributes to the ongoing research efforts by showing that fuzzy logic control can be utilized in automotive control systems. The study shows that a well-designed fuzzy logic controller performs as good as and in some cases better than its conventional counterpart.

In two areas of automotive control systems, fuzzy logic control was applied to design global, robust, and optimal fuzzy logic controllers. The design of such a controller from approximated data and graphical information is shown to be possible. A controller that is based on actual data might be different than the proposed controller, and that difference might be in the rule-base, membership functions, or the overall control schemes, but the applicability of such a controller should be better than the proposed one. Adaptive and sliding mode fuzzy logic controllers can be designed for any type of application that has a high degree of uncertainty and not enough information about the controlled process available. The simulation results show that fuzzy logic is a valuable alternative to conventional control.

Fuzzy Logic Controller Using Normalized Spline Membership Functions for Inverted Pendulum Control Problem

The last practical application that is considered in this chapter is the inverted pendulum control problem. This is a pinch mark nonlinear control problem. Many control methods, conventional or otherwise, deal with this control problem in one way or another. The mathematical model for the systems yields two nonlinear-coupled differential equations. The solution can be obtained by using either
numerical methods or a simplified model that can be obtained based on some assumptions imposed on the system. In conventional control methods, some assumptions are made about the operating region, and the system is then linearized based on those assumptions. Usually the most often used assumption is that the vertical angle will not deviate from vertex beyond a small range. Another assumption is imposed on the linearization process, which is based on neglecting the higher-order terms of the Taylor series expansion. Both of these assumptions yield a linearized mathematical model that can be easily used in the design of the system controller. The controller functions well within the assumed parameters range. If the system goes beyond the assumed parameters range, then the system becomes unstable. Other more advanced conventional control methods deal with this problem, each in its own way. A Linearized model with reduced-order observer is one way to deal with this problem. Adaptive control also has been used in this kind of problem. In adaptive control, for example, a learning mechanism was included into the control algorithms. The learning mechanism will adapt the control system parameters utilizing previous knowledge acquired from the behavior of the system. In the next section, a discussion of the problem and a review of some control schemes that used the fuzzy logic control approach to deal with this problem are presented.

**The Inverted Pendulum Problem**

The inverted pendulum system consists of a rail, a cart, and a pendulum pivoted to the cart, as shown in Figure 92.
The dynamic model of the system can be derived using the Lagrange-Euler equation to yield two nonlinear, coupled differential equations. The derivation is not included here but can be found in any advance dynamics textbook. The system equations are given by Equation (5.52):

\[
\begin{align*}
\ddot{x}(m_c + m_p) + \dot{\theta}Lm_p \cos \theta - \dot{\theta}^2 Lm_p \sin \theta + \dot{x}f_c &= u \\
\ddot{x}Lm_p \cos \theta + \dot{\theta}^2 m_p - Lm_p g \sin \theta + \dot{\theta}f_p &= 0
\end{align*}
\]

(5.52)

where \(m_c, m_p, l, \theta, f_c, f_p, \) and \(x\) are the cart mass, pendulum mass, pendulum length, pendulum deviation angle from the vertical, friction factor resisting the cart movement, friction factor resisting the pendulum motion, and the transitional distance of the cart, respectively. Previous studies that used other nonconventional control techniques to control the inverted pendulum problem can be seen in work conducted by deSylva and Vagners (1995). DeSylva and Vagners used a hybrid controller that combines fuzzy logic deterministic rules and energy based control concepts to balance an inverted pendulum supported on a linear track cart. Sheng et
al. (1994) used knowledge acquired from sliding mode control to design a fuzzy logic controller. Sincak et al. (1996) developed the neural approximate reasoning approach (NARA) and adaptive neural inference (ANFIS) control schemes and demonstrated the developed schemes on the nonlinear inverted pendulum control problem. In this case study, the analytical fuzzy control model developed in Chapter IV is used to represent the membership functions for the inputs and output. The objective is to design a fuzzy logic controller using the developed model for spline membership function representation and to test the validity of the developed model by comparing its performance to other methods, such as the ones mentioned above. The inverted pendulum control diagram is shown in Figure 93.

The membership functions distributed over the universes of discourse for the inputs and outputs are obtained using the methods that were discussed in Chapter IV. Using normalized spline membership function of order 3 the functional format can be used to express the linguistic values mathematically, such as the linguistic value

*Position* FLC

*Angle* FLC

Figure 93. Block Diagram for the Proposed Control Schemes for Inverted Pendulum.
corresponding to Zero (Z) can be expressed analytically as:

\[
Z = \frac{1}{4h} \begin{vmatrix}
\psi_1^3 \\
(1 + 3\psi_2 + 3\psi_2^2 - 3\psi_2^3) \\
(1 + 3\psi_3 + 3\psi_3^2 - 3\psi_3^3) \\
\psi_4^3
\end{vmatrix}
\]

(5.53)

where

\[
\begin{align*}
\psi_1 &= u_1 + 50 & -50 \leq u_1 \leq -25 \\
\psi_2 &= u_2 + 25 & -25 \leq u_2 \leq 0 \\
\psi_3 &= 25 - u_3 & 0 \leq u_3 \leq 25 \\
\psi_4 &= 50 - u_4 & 25 \leq u_4 \leq 50 \\
h &= 25
\end{align*}
\]

(5.54)

The rest of the membership functions are expressed in the same way as above.

The universes of discourse and the number of the membership functions defined for the inputs and output are shown in Figures 94 through 97. The inputs are the cart position, the pendulum angle from the vertical, and their derivatives. The output is the force that will move the cart so that the pendulum is balanced in the vertical upright position.

The rule derived for this system is based on common knowledge of controlling inverted pendulum system. The mathematical model in its linearized form is also used as a base to derive the generalized rules to control the system. The comparison is conducted using Adaptive Neural Inference (ANFIS), which was developed by Sincak et al. (1996), but with different training data and different
ANFIS structure due to the unavailability of the original ones. The rules derived for this study are expressed in the following format:

*If \( x \) is \( N \) and \( v \) is \( P \) Then \( u \) is \( PS \), Also, If \( \theta \) is \( N \) and \( \partial \theta \) is \( Z \) Then \( u \) is \( PS \)*

Figure 94. Universe of Discourse and Membership Functions for the Cart Position and Velocity.

Figure 95. Universe of Discourse and Membership Functions for the Pendulum Angle \( \theta \).
The fuzzy system is then constructed with the minimum operator for the connective (AND), Min-Max for implication and aggregation, and COA for defuzzification. Table 11 lists the fuzzy rule derived and used for this system.
The adaptive neural inference system (ANFIS) used in the comparison study is based on the TSK fuzzy control model as shown in Figure 98.

Table 11
Fuzzy Rules for the Inverted Pendulum With Four Inputs and One Output

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>θ</th>
<th>ωθ</th>
<th>u</th>
</tr>
</thead>
<tbody>
<tr>
<td>Z</td>
<td>Z</td>
<td>Z</td>
<td>Z</td>
<td>Z</td>
</tr>
<tr>
<td>P</td>
<td>Z</td>
<td>Z</td>
<td>Z</td>
<td>NS</td>
</tr>
<tr>
<td>N</td>
<td>Z</td>
<td>Z</td>
<td>Z</td>
<td>PS</td>
</tr>
<tr>
<td>Z</td>
<td>P</td>
<td>Z</td>
<td>Z</td>
<td>NS</td>
</tr>
<tr>
<td>Z</td>
<td>N</td>
<td>Z</td>
<td>Z</td>
<td>PS</td>
</tr>
<tr>
<td>Z</td>
<td>Z</td>
<td>P</td>
<td>Z</td>
<td>PS</td>
</tr>
<tr>
<td>Z</td>
<td>Z</td>
<td>N</td>
<td>Z</td>
<td>NS</td>
</tr>
<tr>
<td>P</td>
<td>P</td>
<td>Z</td>
<td>Z</td>
<td>NM</td>
</tr>
<tr>
<td>N</td>
<td>N</td>
<td>Z</td>
<td>Z</td>
<td>PM</td>
</tr>
<tr>
<td>P</td>
<td>P</td>
<td>P</td>
<td>Z</td>
<td>NM</td>
</tr>
<tr>
<td>N</td>
<td>P</td>
<td>N</td>
<td>Z</td>
<td>PS</td>
</tr>
<tr>
<td>P</td>
<td>P</td>
<td>P</td>
<td>P</td>
<td>PM</td>
</tr>
<tr>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>PM</td>
</tr>
<tr>
<td>Z</td>
<td>Z</td>
<td>P</td>
<td>P</td>
<td>PM</td>
</tr>
<tr>
<td>Z</td>
<td>P</td>
<td>Z</td>
<td>N</td>
<td>Z</td>
</tr>
</tbody>
</table>

Figure 98. ANFIS Model Using TSK Fuzzy Model.
The layers shown in Figure 98 functions as follows:

In layer one, each node is adaptive and it represents the membership function, which can be any of the previously discussed ones, but the most often and practical ones to use take the form of nonlinear differentiable functions, such as bell-shaped or Gaussian. In a functional form this layer can be presented by \( L_i(x) = \mu_{A_i}(x) \), where \( x \) is the input and \( A_i \) represent the linguistic value, such as Negative Medium (NM). The membership function \( \mu_{A_i}(x) \) can be expressed analytically as:

\[
\mu_{A_i}(x) = \frac{1}{1 + \left[ \frac{(x - c_i)}{\sigma_i} \right]^p} 
\]

where \( p, c_i, \) and \( \sigma_i \) are the parameters that determine the membership function shape and width in the antecedent part of the If-Then rules.

Layer two functions as a simple multiplication of the inputs that produce the output strength \( \alpha_i \).

Layer three calculates the ratio of rule strength according to Equation (5.56):

\[
W_i = \frac{\alpha_i}{\alpha_1 + \alpha_2}, \quad i = 1, 2, \ldots 
\]

Layer four is an adaptive layer that calculates the transfer function \( W_iF_i \), according to Equation (5.57):
\[ W_i f_i(x) = f_i(x) = W_i (a_i + n_i x + m_i y) \tag{5.57} \]

where \( a_i, n_i, m_i \) are the consequent parameters of the output functions expressed by \( f_i(x) \).

Layer five adds the inputs, as given in Equation (5.58):

\[ f(x) = \frac{\sum_{i} W_i f_i(x)}{\sum_{i} W_i} \tag{5.58} \]

The system is trained using training data and checking data until a small and satisfactory training and checking error is obtained. The system is then tested using testing data. It should be pointed out that in training the system, only the parameters that define the membership function and the transfer function are adjusted.

**Simulation Results and Discussion**

The developed model and ANFIS fuzzy model are both simulated for the inverted pendulum depicted in Figure 93. The parameters for the pendulum used are as follows. The pendulum mass \((m_p)\) is 0.1 kg, the cart mass \((m_c)\) is 1 kg, the pendulum length \((L)\) is 1 meter, and the gravity constant \(g = 9.81\). The initial conditions for the position, velocity, vertical angle, and angular velocity are all chosen to be zeros. The ANFIS model that is used for comparison is shown in Figure 99. The simulation diagram for the model developed here is shown in Figure 100. The ANFIS is developed in the same way as that of the developed model except for
only a minor difference. The ANFIS model uses only one fuzzy logic controller with four inputs and one output, while the developed model used two fuzzy logic controllers each with two inputs and one output as depicted in the Figure 100.

The aim of the simulation is to balance the inverted pendulum in the vertical position by moving the carts on the rails in either direction using the input force $u$. Also, the position should be controlled so that for different angle positions the cart should move to the position that achieves balanced and stable system.

Figure 101 shows the response of the developed model for the angle and the tracking position of the measured and target position. Figure 102 shows the same response for the ANFIS system under the same conditions as those for the developed model.

Figure 99. SIMULINK Simulation Diagram for ANFIS Model.
Figure 100. SIMULINK Simulation Diagram for the Developed Model.

From the simulation results shown in the Figures 101 through 102, the performance of the developed and ANFIS model is very comparable. The response for the angle and position appears to be better for the developed model because of its flexibility in introducing modification without changing the overall control algorithm. Once the ANFIS is modeled, the modification for better performance is somewhat limited and involves changing the membership functions and the training and checking error tolerance. This, however, is overcame by incorporating an adaptive
Figure 101. The Response for the Developed Model. Angle Response (Left), Cart Position Response (Right).

Figure 102. The Response for the ANFIS Model. Angle Response (Left), Cart Position Response (Right).

fuzzy logic control scheme. Figure 103 shows a comparison between the developed model and ANFIS model. The figure shows that the developed model performed better than the ANFIS model. These results validate the developed model and show
that the developed model can be successfully used in developing a fuzzy logic controller that has the advantages of being expressed in a functional form which can be standardized for designing a control system for similar applications.

![Figure 103. The Angle Response Comparison Between The Developed And ANFIS Models.](image)

This concludes the practical applications of fuzzy logic systems developed and other models that use fuzzy logic in different formats, such as the adaptive, PI, PD, or PID type fuzzy logic controller. The fuzzy controllers used in this chapter encompass a wide variety of applications that are inherently nonlinear systems and are current research topics in the scientific community.

This chapter served two purposes; first, it has shown the practicality of fuzzy logic control in its different design formats. Second, it has shown that the approach used in this dissertation can be used to build upon research that is currently going on in the area of automotive industries, especially in the area of engine management.
control and air/fuel ratio control, which are becoming very important research topics
for environmental and economical reasons. In Chapter VI, experimental results that
are obtained for the first two controllers developed in this chapter will be presented.
In Chapter VII, the inverted pendulum problem is used again, but in a different
format, to study the effects of the parameter selections in the fuzzy logic controller
design process. Parameters such as membership function shapes, connective t-norms,
and the defuzzification process, in addition to the aggregation operator, are presented.
Also, guidelines for choosing such factors and tuning the controller for better
performance are presented.
CHAPTER VI

EXPERIMENTAL SETUP AND PROCEDURE

Introduction

The models developed in Chapter V for the idle speed control are used here to verify the simulation results obtained using Matlab and GUI SIMULINK software. The first experiment verification was conducted at the Applied Dynamics International (ADI) facilities. Applied Dynamics International (ADI) is a specialized company that provides state-of-the-art software and hardware tools to the automotive, aerospace, and defense industries for the design and testing of embedded control systems. The hardware and software provides an integrated environment called SIMsystem for the control system design, from the conception of the design to the real applications, using rapid prototyping.

The SIMsystem is comprised of a host personal computer (PC) and/or workstation with a network connection to the real-time station (AD RTS). The AD RTS is connected to the hardware under testing and provides the requisite real-time computing power and I/O devices. All program development occurs on the host. The run-time operation uses client/server technology, allowing the AD RTS to be a network resource accessible from the host on the network. The SIMsystem software include Advantage Integrated Development Environment (IDE) and interactive run-
time software. A typical experimental setup that ADI provides is shown in Figure 104.

![Figure 104. SIMsystem Setup at ADI Facilities.](image)

Advantage IDE supports the migration of non-real-time development models to real time hardware-in-the-loop (HIL) execution. IDE accepts models created using popular graphical programming packages such as SIMULINK, SystemBuild™, C, and FORTRAN and prepares them for real-time execution in the compute engines of the AD RTS as shown in Figure 105. The SIMsystem run-time software allows a user to become interactively involved with the simulation in progress without impacting the real-time execution. This provides a powerful basis for optimizing analysis of the system being tested. The host software of Interact, SIMplot, and SIMpanel operates with the RTexec and Data Acquisition System (DAS) on the AD RTS. The DAS allows run-time selection of simulation variables and supplies these data to the host for further analysis. Figure 106 shows the detail of how this can be accomplished.
RTexec is a powerful real-time, multi-thread kernel that guarantees microseconds level determination.

Figure 105. Diagram of Integrated Software and Hardware at ADI.

The AD RTS is an open architecture as can be seen in Figure 106. VME bus-based system features support for multiple compute engines and a wide variety of fully integrated I/O products. The Interface processor handles all network operations.
The compute engines employ a wide range of microprocessors. SIMsystem supports broad spectrums of I/O devices.

Figure 106. Diagram of AD Real Time Station (RTS).

SIMsystem software allows for easy interconnect of I/O devices to models. Using either point-and-click interface or automatic connect, model I/O is quickly associated with the actual hardware. The Parallel Intelligent Resource (PIR) operates the I/O in parallel with model execution. In a PIR, local I/O processor controls a complete I/O subsystem, including initiating I/O sequences and data conversion, so that only refined data appear on the system bus instead of all the intermediate
activity. SIMsystem simplifies rapid prototyping of controllers. The prototype control software can be implemented with the actual plant hardware in the loop or by simulating the plant on the same bus. The experimental procedure then becomes a very simple task of using the ADI experimental setup and equipment in addition to the developed fuzzy logic control models that were described in Chapter V. Figure 107 shows how the SIMULINK model is used in the testing process.

![Figure 107. Block Diagram Showing How the Model Developed Is Downloaded Onto the AD RTS Using Real Time Workshop (RTW).](image)

Processor Selection

The choice of processor for a control system is critical and many factors have to be considered, such as (a) word length (8, 16, 32-bit); (b) data type (fixed point--
floating point); (c) processing speed; (d) mathematical capability; (e) time processing; and (f) interrupt handling. The selection of an appropriate processor depends on the application criteria, such as (a) desired performance (sampling time, control accuracy, response time, etc.); (b) hardware (bus system, availability, etc.); (c) software (availability and performance of development and debugging tools); and (d) system cost.

Processors to handle complex real-time control problems can fall into the following categories:

1. General-purpose microprocessors: They are central processing units (CPUs) that require external memory and support chips to complete the control system.

2. Microcontrollers: They are control-oriented devices that offer a high level of integration on a single chip. They typically incorporate a CPU and peripheral units required for real-time operation, such as programmable timer system, A/D converter, interrupt controller, pulse width modulator, and digital I/O.

3. Application Specific Integrated Circuits (ASICs): They are custom, semi-custom, or standard ICs that may integrate on-chip several analog and digital components necessary for a specific application, such as microprocessor, A/D converter, PNM modulator, or I/O interface.

4. Digital Signal Processors (DSPs): They are high-speed processors with Harvard architecture designed for efficient execution of signal processing algorithms. In this processor, which is described in greater detail since it is the one that was chosen for our experimental verification, the multiply/accumulate (MAC) operation
is optimized and repeats instruction is implemented.

5. Reduced Instruction Set Computing (RISC) microprocessors: They are high-speed processors with pipeline architecture and simple instructions. In this kind of processor, the complex operations are executed by software.

6. Parallel processors: They are high-performance microprocessors equipped with communication links which can be connected in network to operate concurrently in order to increase processing speed and to improve control flexibility and performance.

Digital Signal Processors (DSPs)

Several DSP generations with increasing performance have been introduced by different manufacturers. These chips have been developed specifically for real-time computing in digital signal processing applications. Many DSPs function as embedded real-time processors with special purpose hardware, such as modems, speech coders, image processing systems, etc. Most DSPs are built with Harvard architecture, where data and instructions occupy separate memories and travel over separate buses. Because of this dual bus structure, the processor can process simultaneously an instruction and a data operand. Pipelined operation of instruction and data transfer is thus possible, resulting in a higher instruction throughput rate. In order to optimize the processing speed, important operations, such as multiplication and shift, are implemented in the hardware instead of using software. The execution speed can be enhanced by using several independent units, multiple bus sets, and
additional units, such as instruction cache, register file, and dual-access memories.

The operation of DSPs is optimized so that most of the instructions are executed in a single cycle. DSPs can also perform parallel multiply and arithmetic-logic unit (ALU) operations on integer or floating-point data in a single cycle. The multiply/accumulate (MAC) operation is the basic operation that is optimized in DSPs. This operation is used in most signal processing and control algorithms, such as digital filters, FFT, PID controllers, fuzzy controllers, etc. Special instructions are also used to enhance the execution speed of signal processing and control algorithms.

The high computation capability of DSPs is used to increase the sampling rate and to implement complex signal processing and control algorithms. Implementation of conventional algorithms or fuzzy control algorithms utilizes the same basic operations as signal processing algorithms and can thus benefit from the DSP’s capabilities. The signal processing capability of DSPs can also be used to reduce the number of sensors. Sensorless operation is thus possible because the system variables usually provided by sensors can be estimated from the electrical variables. In adaptive control, for example, the system parameters and/or state variables can be estimated using a state observer that can be implemented using a DSP.

Microprocessor-Based Control System Design

The development of a microprocessor-based control system is a complex task consisting of several stages, usually completed by several engineers. It involves the design of both software and hardware components and their integration while
considering various factors such as system performance specifications, processor computing capacities, hardware availability, software development debugging tools, and system cost.

The system requirements have to be established in the format of specifications. The system specifications must be as detailed as possible since the subsequent stages depend on them. The functional specifications must detail the control strategy and configuration, as well as the different control and regulation tasks that the control system has to accomplish. The system specification then leads to the selection of one or many appropriate microprocessors capable of accomplishing the tasks. Processor selection is a major task that requires a good knowledge of the desired functions and the performance of the final system.

The trade-off between hardware and software approaches to implement the different functions in control systems is done on the basis of performance-cost compromise. Software or hardware can implement most of the functions. Implementation by software, processing time allowed to a given function, will increase the sampling period, resulting in performance decrease. Implementation by hardware, additional devices required by the functions, will increase necessarily the system complexity and cost. In general, hardware implementation is used to reduce the computing load of the CPU.

Two alternatives often confront the control system designer when choosing the hardware. The first is to build the processor board to suit a specific application. The second is to use commercial processor boards that are available in different bus
systems. Building a processor board for a specific application ensures optimum utilization of the hardware, but the development time is very long. The use of commercial boards reduces the hardware development and debugging time. The boards, however, are not always optimized for the specific application.

The development and debugging of control software for a microprocessor-based control system require different tools, depending on the selected processors and system complexity. The software developed for real-time control typically contains a real-time operating system that schedules and manages different specific functions under the form of tasks or processes. The development of real-time control software can be done following three main steps: (1) development and testing of the algorithms by simulation, (2) coding and testing in off-line mode, and (3) testing of developed codes in real-time mode. Assembler language or higher-level language can write the control software. Assembler is effective programming language for real-time control systems because it gives access to the processor internal structure. The codes can be optimized to efficiently use the available memory space and to optimize execution speed. A major drawback of assembler programming resides in the processor's dependence on the developed software. Higher-level language such as C-language is widely accepted as a programming language for real-time control systems because of its portability and effectiveness in manipulating hardware resources. The developed codes using C-language can be brought to another processor generation or to another microprocessor with minimum modifications. However, the generated code is less compact than that produced by assembler.
language.

The integration of the hardware and software is a necessary step in testing the control system performance. The interaction between the two parts can be studied in real time by running the developed software on actual hardware or by using an in-circuit emulator. The performance of the control system can be thus evaluated under real operating conditions and compared to the specifications established during the initial design process. Iteration is a necessary step to ensure that the system functions conform to the specifications and the performance is satisfactory. In the case of engine control, the microcontroller can take many input signals and produce output signals that can be used to control different aspects of engine control variables. A typical controller block diagram is shown in Figure 108.

**DSP Processor Chosen for Simulation**

The DSP processor that is selected for the simulation purpose is TMS320BC52 abbreviated by 'C52. The TMS320BC52 CPU device is the newest member of the 'C5X generation of Texas Instruments. The combination of advanced Harvard architecture, on-chip peripherals, on-chip memory, and a highly specialized instruction set is the basis of the operational debug user code under control of the monitor program. User code is assembled into an Intel HEX format using the EZ-DSP CPU manager software installed on a host computer and is then downloaded to the EZ-DSP CPU's RAM via a RS-232 serial port. The monitor program is then used to execute and debug the assembled user code.
Figure 108. Typical Use of a Microcontroller in Engine Management Control Problem.

The EZ-DSP CPU provides a low-cost method for debugging and evaluating the TMS320BC52 Texas Instruments microprocessor. The TMS320BC52 architecture maximizes the processing power by maintaining two separate memory bus structures, program and data, for full-speed execution. Instruction support data transfer between the two spaces. This architecture eliminates the need for a separate
coefficient ROM. It also makes available immediate instructions and subroutines based on computed values. Control signals and instructions provide floating-point support, block memory transfers, communication to slower off-chip devices, and multiprocessing implementations. The TMS320BC52 microprocessor that is used on the EZ-DSP CPU-52 microprocessor board also offers the following: (a) 25, 35, and 50-ns single-cycle instruction execution time with 3-v or 5-v operation; (b) single-cycle 16 x 16 bit multiply/add; (c) 128K words total data/program space; (d) 4K x16-bit single access on-chip program ROM; (e) iK x 16-bit dual access on-chip program/data ROM; (f) full duplex synchronous serial port for CODEC interface; (g) hardware and software wait state generation; (h) on-chip timer for control operations; (i) multiply-by-two and divided-by-two clocking options; (j) on-chip scan based emulation logic; and (k) low-power dissipation and power down modes-2.35 MA/MIP at 5v, 40-Mhz; 3 mat 5v, 40-Mhz (IDLE2); 5 μA at 5v, blocks off (standby state).

The 'C52 implements three separate address spaces for programming memory, data memory, and I/O. Each space accommodates a total of 64K 16-bit words. Within the 64K words of data space, the 256 to 32K words at the top of the address range can be defined to be external global memory in increments of powers of two, as specified by the contents of the GREG register.

The EZ-DSP CPU-52 microprocessor board has analog interface circuit (AIC) that uses Texas Instruments TLC320AC02 device. This device is an audio-band processor that provides an analog-to-digital (A/D) and digital-to-analog (D/A)
input/output interface system, a 14-bit resolution (A/D) converter, and a 14-bit resolution (D/A) converter. Serial port is used for data and control transfers. The maximum sampling rate for the (A/D) channel is 43.2 kHz. The sampling rate maximum rate for the (D/A) channel is 25 kHz. A 10 MHz timer output provides the master input clock to the AIC from the TMS320C52.

The 'C52 device is designed to execute more than 28 MIPS (million instructions per second). Hardware timer for 'C52 features a 16-bit timing circuit with a 4-bit prescaler. The timer clocks between one-half and one-thirty-second the machine rate of the device, depending upon the programmable timer’s divide-down ratio. The timer can be stopped restarted, reset, or disabled by specific status bits.

Access to external parallel I/O ports is multiplexed over the same address and data bus for program/data memory address. I/O space access is distinguished from program/data memory accesses by the IS signal going active low. All 65536 ports are accessed via the IN and OUT instruction, such as “IN DAT7, Offfeh,” which reads data into data memory from external device on port 65534, and “OUT DAT7, Offffh,” which writes data from data memory to external device on port 65535.

The I/O ports can be accessed with the IN and OUT instructions along with any instruction that reads or writes a location in data space.

All of the analog interface functionality on the EZ-DSP board is provided by an analog interface chip. The TLC320AC02 analog interface circuit provides an analog-to-digital and digital-to-analog input/output interface system on a single
monolithic CMOS chip. This device integrates a band pass switch capacitor anti-aliasing input filter, and a 14-bit-resolution A/D and D/A converter.

One of the main functions of the TLC320AC02 is to provide the interface and control logic to transfer data between its serial input and output terminals and a DSP. The A/D and D/A channels operate synchronously, and the data received at the D/A channel and data transmitted from A/D channel occur at the same time.

**Laboratory Board**

The lab board is designed to interface with the EZ-DSP board. It provides the following features: (a) 4 row x 3 column keypad for DTMT generation and DSP input, (b) three 7-segment displays with drivers for DSP output, (c) microphone and preamp circuit, (d) audio amp and speaker with volume control, and (e) digital output. The board has one analog input channel and one analog output channel.

Testing the assembly language code generated by DSP software is conducted using the laboratory board just described. The laboratory board serves as a communication channel between the DSP and the outside world. Figure 109 shows the connection between the laboratory board and the DSP board.

**Engine Simulator**

The engine simulator used to test the controller consists of a switchboard and hardware that are programmed to function in the same way as a real engine does. The engine model is designed based on experimental data that were obtained through an
extensive experimental data collection scheme under different operating conditions and a combination of conditions. The idle speed control condition has to take into account many factors and variations, such as, the variation of the spark advance, throttle opening, acceleration and deceleration, as well as cold start, transient conditions, and torque application. The torque engagement considered in the engine simulator model can be the situation that includes the gear engagement from neutral to drive or gear shifting in standard manual shifting transmission, air-conditioning, power steering, and any other similar function or a combination of functions. The engine simulator, therefore, functions exactly as the real engine would under certain driving or idling conditions. The engine simulator used in this work consists of two units. The first unit is the engine controller test stand unit. The second unit is the engine controller load box. Chrysler Corporation makes the engine controller test stand unit and the engine controller load box unit, which are the same units used in Chrysler testing and simulation facilities.
consists of a board that contains knobs and dials for each function of the engine operation under different situations. For example, an air conditioning knob simulates the air conditioning being switched on in real situations. The injection pulse width, as well as the vehicle speed, is displayed on a digital graph display box. The engine speed in miles per hour (mph) or revolutions per minute (rpm) can also be obtained in graphical display format. A digital to analog converters and vice versa can be connected to the controller box to facilitate the experimental process and to speed up the experimentation time. The oxygen sensor at different settings and locations can be simulated to study the impact of certain situations of the engine operating conditions. This can be utilized to study the effect of controlling air-fuel ratio on pollutant emission, as well as the effect of the idle speed controller on the air-fuel ratio and fuel economy. The setting of the throttle and spark advance can also be altered by using their respective knobs. The engine controller test stand is connected to the engine controller load box through hardware and harnesses. The engine controller load box is connected to the data acquisition as well as to a computer terminal for data display. The engine controller load box, which houses the controller unit of the engine in addition to the tested controller unit, has built into it the sensors, such as the check engine sensor, A/T sensor, S/C sensor, SIL sensor, and EMR sensor. It also houses the ignition switch, the spark blocks, and the distributor. Appendix B shows a photo of the engine simulator test stand. A detailed description of how the experimental setup works can be found in the user's manual, which is specifically written for that purpose.
The experiment setup for the engine simulator as explained above is then utilized to test the idle speed controller developed as was discussed in Chapter V. The controller is loaded on a DSP board, which in turn functions as the idle speed controller with the controller inputs simulated using the engine controller test stand knobs and dials. The output is sensed and observed on the computer screen as well as the test stand digital display screen. The complete set of the experimental data could not be obtained due to the lack of a data acquisition board. To rectify that problem, many attempts were made to use the testing facilities at Ford and Chrysler Corporations, but none of the attempts succeeded due to security concerns. The experimental results obtained showed that the controllers developed performed well and the simulation performance, which should be enough to show the controllers' satisfactory performance, is supported by two kinds of experimental verification, as discussed earlier in this chapter.

The experimental results showed that the controllers, even though they performed as expected, are also robust controllers in the sense that when used in two different experimental setups with the difference in the hardware and software utilized for each setup, the controllers performed extremely well, and minor modifications to the internal structure of the developed controllers were needed when conducting the experiment in both cases. This unintended result of showing the flexibility associated with stability of the developed fuzzy controllers under different operating points or testing methods can be of great benefit. This was shown in the simulation and was documented in the experiment results obtained. The stability and
the robustness of the fuzzy control system are an entire different subject and were not the aim of this work. Appendix A contains a description of the fuzzy logic controller implementation and numerical valued tables obtained for the throttle and spark advance control situations. The experimental results that were collected using the graphical and digital displays on the engine controller test stand and the revised fuzzy rules used in the experiment are also explained in Appendix A.

Experimental Results and Discussion

The experimental setups discussed in the previous sections, both for the DSP and the ADI are used to verify the two models developed in Chapter V for the idle speed control using a systematic approach and combined variables control approach. The experimental results obtained are presented, and a discussion of these results will conclude this chapter.

ADI Experimental Results

The ADI results obtained using the developed model for the fuzzy logic controller and using the setup depicted in Figure 107 were very comparable to the simulation results obtained using developed controller algorithms. The experimental results were obtained under a different set of inputs and disturbances due to the difficulties in duplicating the inputs and disturbances used in simulation. The ADI setup allows inputs during the execution time of the control algorithm, while in simulation the inputs and disturbance were specified ahead of time in the developed
algorithms. The results, however, were very close to the simulation results, which indicates that under a set of operating conditions, the controller should function as expected with some tolerated variations due to the differences between software and system parameter interpretations.

The first part of the experimental results is devoted to the results obtained for the systematic fuzzy logic controller design with LR parameterization of the membership functions. The model is explained in more details in Chapter V. Figures 110 to 114 shows the experimental results. Figure 110 shows the step response for the throttle change. Figure 110 shows the same trend as the simulation trend shown in Figure 41 of the simulation results obtained in Chapter V. The desired speed was set at 650 rpm with no load applied. Figure 110, however, shows that the response is oscillatory and has a much higher overshoot. The differences between the two figures is due to the fact that in the experimental setup many parameters such as the software, frame rate, hardware and software integration, platform fuzzy software, and other normal factors that characterize the experimental work. The steady state error is the same in both cases. Figure 111, on the other hand, shows the response when a load torque of 20 N.m is applied at the time instances shown in the figure. The throttle input and the speed error are shown in the top and bottom figures, respectively. The speed error increases to around 30 rpm at the instant the load is applied. The controller was able to eliminate the error in only 2 seconds.
Figure 110. Experimental Step Response for Idle Speed Control With Desired Speed Set at 650. The corresponding simulation is Figure 41, Chapter V.

Figure 111. Experimental Engine Speed Error for Throttle Control (Top), Load Torque (Middle). This figure corresponds to Figure 42, Chapter V.
Figure 112 shows the response when a load torque is applied as the instances shown in the top figure. The figure shows that both the experimental and the simulation curves have similar response trend, although the experimental curve shows higher overshoot. Both, however, have a zero steady state error at the instances when the load torque is changed, reduced, or removed. The average engine speed error for both the simulation and experimental response is within a range of ±30 rpm as shown in the figure. The simulation figure that was shown in Figure 44 depicts a similar trend in the control action and system response.

![Graph showing experimental and simulation curves](image)

Figure 112. Experimental Idle Speed Error ($N_m - N_d$) (Bottom), Load Torque (Top). This figure corresponds to simulation Figure 44, Chapter V.

In Figure 113, the measured speed is shown to deviate by 150 rpm from the desired idle speed, which is set at 600 rpm. It should be noted here that an idle speed
of 600 rpm is a relatively very low idle speed and any small disturbances will have a large effect on the controller performance and system response. The steady state error corresponds well with the simulated resulted error shown in Figure 45. This steady state error is acceptable in idle speed control under step load disturbance and at such a low idle speed setting. This case, however, shows that stalling of an engine can occur due to this lower setting of the idle speed.

![Graph showing experimental idle speed control response](image-url)

Figure 113. Experimental Idle Speed Control Response When a Step Load Torque Is Applied. This figure corresponds to simulation Figure 46, Chapter V.

Figure 114 shows the step load response for both simulated and experimental cases. The difference in overshoot and steady state error is, in this case, around 75 rpm, even though the desired set idle speed was 650 rpm, which makes the difference around 125 rpm and 50 rpm for the experimental and simulation, respectively.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
The results for the second case study that was developed for idle speed using combined throttle and spark control under different operating conditions are presented in Figures 115 through 120. In Figure 115, the response of the system is shown for the throttle control. The system responded in a similar manner as simulation results predicted but with a high oscillatory transitional response and a high value of the overshoot. This can be seen if Figure 115 is compared to Figure 60 in Chapter V. The comparison between Figure 115 and Figure 60 shows that same control action trend and system response are similar but not the same due to different experimental inputs and disturbances.
Figure 115. Experimental Result of Engine Speed Error (Bottom), Throttle Control (Top). This figure corresponds to Figure 60, Chapter V.

Figure 116 shows the combined spark and throttle Controls and the system response. This figure corresponds to the simulation figure shown in Figure 61 of Chapter V. If Figure 116 is compared with Figure 115, one can see that the throttle alone control response is better than the combined control actions of the throttle and spark. However, the combined throttle and spark reduced the oscillatory tendency in the system.

Figure 116. Experimental Engine Speed Error Under Throttle and Spark Control With No Load Torque Applied.
Figure 117 shows the response of the system under a load disturbance of -100 rpm. The system responded very well to the load disturbance, and the error is kept within a small range of ± 25 rpm. The throttle and spark control is better used when the disturbance load is applied, since the disturbance load causes a sudden reduction in the engine rpm. The spark control can compensate for the delay that only throttle control can cause. This is the same trend as was shown in the simulation in Figure 62, but with different throttle and spark control. Figure 117 shows the oscillatory transition response when the load is on and off at the indicated time instances. The oscillatory amplitude, however, is small and of a short, duration, which can be considered acceptable in this situation.

Figure 117. Experimental Spark and Throttle Control When a Load Torque of -100 rpm is Applied. See Figure 62 for simulation results.
Figure 118 shows the only throttle control situation, where spark is held constant at 10° BTDC and a disturbance load of -50 rpm is applied at the instances shown in the figure. The response is similar to the spark and throttle control and compares well with Figure 62, which shows a similar response trend.

![Graph](image-url)

Figure 118. Experimental Idle Speed Error Under Throttle Control With a Load Torque of -50 rpm and Spark Set at 10° BTDC. See Figure 63 for simulation results.

Figure 119 shows the situation when a load of -100 rpm is applied at the time indicated in the figure and the throttle is held constant, while the spark control is activated. The spark control alone performed within an error of ±100 rpm. In comparing this response within the throttle alone response, the throttle alone response performed better under the same conditions.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Figure 119. Experimental Idle Speed Error Under Spark Advance Control, Constant Throttle, and a Load Torque of -100 rpm. The simulation result is shown in Figure 64.

Figure 120 shows the response of the system under the throttle and spark control with the disturbance load reduced to -50 rpm. The system shows an initial oscillatory response in the absence of the applied load. When the load was applied, the system oscillatory response improved due to the fact that the control action tried to reduce the disturbance effect. The steady state error obtained is within ± 20 rpm.

DSP Microcontroller and Engine Simulator Experimental Results

The second part of the experimental verification consists of two stages. The first stage is loading the fuzzy logic controller algorithm onto a DSP board. The algorithm is then tested using only the DSP and the laboratory board as was shown in Figure 109 to test and debug the assembly language codes of the control algorithms.
The DSP board generates the data for certain inputs and measures the output of the fuzzy controller defined in both case studies. The code of the fuzzy controller is converted to assembly language code using EZ-DSP manager software. The EZ-DSP manager software communicates directly to the EZ-DSP CPU-52 using the host computer, as explained earlier in the previous sections. The controller file is converted to the assembly language source file. The source file is then converted to the Intel HEX file, which is downloaded to the EZ-DSP CPU 52 microprocessor board and executed. Appendix A contains a sample of the assembly language file.

Tables 12 and 13 are a sample of the data generated. In Table 12, the data generated are those for the spark advance (degrees before top dead center (BTDC) as a function of the fuel injection time (in ms) and engine speed (rpm). The data obtained agree well with the simulated data obtained in Chapter V. The table shows a partial data set and should not be interpreted as the whole data set obtained. This
type of data, when obtained in either the simulation or the experimental stage, can be used to design for better fuel efficient car and can also be utilized in a timing scheme that could be activated to yield better fuel and speed optimization. A similar table is obtained for the throttle as a function of the error and the error change. Table 13 shows the data in the case of the throttle controller output. The table shows the throttle as a function of the error and error change. The first column gives the value of the throttle in degrees as a function of the error and its change.

The second stage consists of using the DSP controller board as a part of the engine controller module to control the idle speed. In this stage, as explained in the earlier section, an engine simulator is used. The complete description of the engine simulator and the experimental setup is explained in Appendix A and a photo of the engine simulator experimental setup unit is included in Appendix B. The experiments conducted in this stage are based on using the assembly language developed fuzzy control algorithms.

The experimental results obtained in this situation are similar to the results obtained from the ADI experiment results in graphical format but are different in their numerical values due to the difficulties in duplicating the same experimental procedure for two totally different experimental platforms and setup. The controllers performance observed using the Oakland University engine simulator testing facilities similar to the performance obtained using the Applied Dynamics International (ADI) facilities.
Table 12

Spark Advance as a Function of Injection Time ($T_{inj}$) and Engine Speed ($N$).

<table>
<thead>
<tr>
<th>Spark Advance (DBTDC)</th>
<th>$T_{inj}$ (ms)</th>
<th>$N$ (rpm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.25E+01</td>
<td>0.00E+00</td>
<td>0.00E+00</td>
</tr>
<tr>
<td>2.25E+01</td>
<td>2.56E-15</td>
<td>1.71E-12</td>
</tr>
<tr>
<td>2.25E+01</td>
<td>1.80E-01</td>
<td>1.20E+02</td>
</tr>
<tr>
<td>2.25E+01</td>
<td>3.60E-01</td>
<td>2.40E+02</td>
</tr>
<tr>
<td>2.25E+01</td>
<td>5.40E-01</td>
<td>3.60E+02</td>
</tr>
<tr>
<td>2.25E+01</td>
<td>7.20E-01</td>
<td>4.80E+02</td>
</tr>
<tr>
<td>2.23E+01</td>
<td>9.00E-01</td>
<td>6.00E+02</td>
</tr>
<tr>
<td>2.22E+01</td>
<td>1.08E+00</td>
<td>7.20E+02</td>
</tr>
<tr>
<td>2.19E+01</td>
<td>1.26E+00</td>
<td>8.40E+02</td>
</tr>
<tr>
<td>2.15E+01</td>
<td>1.44E+00</td>
<td>9.60E+02</td>
</tr>
<tr>
<td>2.10E+01</td>
<td>1.62E+00</td>
<td>1.08E+03</td>
</tr>
<tr>
<td>2.03E+01</td>
<td>1.80E+00</td>
<td>1.20E+03</td>
</tr>
<tr>
<td>1.94E+01</td>
<td>1.98E+00</td>
<td>1.32E+03</td>
</tr>
<tr>
<td>1.84E+01</td>
<td>2.16E+00</td>
<td>1.44E+03</td>
</tr>
<tr>
<td>1.74E+01</td>
<td>2.34E+00</td>
<td>1.56E+03</td>
</tr>
<tr>
<td>1.64E+01</td>
<td>2.52E+00</td>
<td>1.68E+03</td>
</tr>
<tr>
<td>1.57E+01</td>
<td>2.70E+00</td>
<td>1.80E+03</td>
</tr>
<tr>
<td>1.53E+01</td>
<td>2.88E+00</td>
<td>1.92E+03</td>
</tr>
<tr>
<td>1.62E+01</td>
<td>3.24E+00</td>
<td>2.16E+03</td>
</tr>
<tr>
<td>1.74E+01</td>
<td>3.42E+00</td>
<td>2.28E+03</td>
</tr>
<tr>
<td>1.89E+01</td>
<td>3.60E+00</td>
<td>2.40E+03</td>
</tr>
<tr>
<td>2.08E+01</td>
<td>3.78E+00</td>
<td>2.52E+03</td>
</tr>
<tr>
<td>2.29E+01</td>
<td>3.96E+00</td>
<td>2.64E+03</td>
</tr>
<tr>
<td>2.52E+01</td>
<td>4.14E+00</td>
<td>2.76E+03</td>
</tr>
<tr>
<td>2.73E+01</td>
<td>4.32E+00</td>
<td>2.88E+03</td>
</tr>
<tr>
<td>2.91E+01</td>
<td>4.50E+00</td>
<td>3.00E+03</td>
</tr>
<tr>
<td>3.05E+01</td>
<td>4.68E+00</td>
<td>3.12E+03</td>
</tr>
<tr>
<td>3.12E+01</td>
<td>4.86E+00</td>
<td>3.24E+03</td>
</tr>
<tr>
<td>3.16E+01</td>
<td>5.04E+00</td>
<td>3.36E+03</td>
</tr>
<tr>
<td>3.17E+01</td>
<td>5.22E+00</td>
<td>3.48E+03</td>
</tr>
<tr>
<td>3.16E+01</td>
<td>5.40E+00</td>
<td>3.60E+03</td>
</tr>
<tr>
<td>3.14E+01</td>
<td>5.58E+00</td>
<td>3.72E+03</td>
</tr>
<tr>
<td>3.12E+01</td>
<td>5.76E+00</td>
<td>3.84E+03</td>
</tr>
<tr>
<td>3.12E+01</td>
<td>5.94E+00</td>
<td>3.96E+03</td>
</tr>
<tr>
<td>3.15E+01</td>
<td>6.12E+00</td>
<td>4.08E+03</td>
</tr>
</tbody>
</table>
Table 12—Continued

<table>
<thead>
<tr>
<th>Spark Advance (DBTDC)</th>
<th>$T_{\text{ms}}$ (ms)</th>
<th>$N$ (rpm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.33E+01</td>
<td>6.48E+00</td>
<td>4.32E+03</td>
</tr>
<tr>
<td>3.44E+01</td>
<td>6.66E+00</td>
<td>4.44E+03</td>
</tr>
<tr>
<td>3.53E+01</td>
<td>6.84E+00</td>
<td>4.56E+03</td>
</tr>
<tr>
<td>3.60E+01</td>
<td>7.20E+00</td>
<td>4.80E+03</td>
</tr>
<tr>
<td>3.57E+01</td>
<td>7.38E+00</td>
<td>4.92E+03</td>
</tr>
<tr>
<td>3.54E+01</td>
<td>7.56E+00</td>
<td>5.04E+03</td>
</tr>
<tr>
<td>3.51E+01</td>
<td>7.74E+00</td>
<td>5.16E+03</td>
</tr>
<tr>
<td>3.51E+01</td>
<td>7.92E+00</td>
<td>5.28E+03</td>
</tr>
<tr>
<td>3.54E+01</td>
<td>8.10E+00</td>
<td>5.40E+03</td>
</tr>
<tr>
<td>3.59E+01</td>
<td>8.28E+00</td>
<td>5.52E+03</td>
</tr>
<tr>
<td>3.67E+01</td>
<td>8.46E+00</td>
<td>5.64E+03</td>
</tr>
<tr>
<td>3.75E+01</td>
<td>8.64E+00</td>
<td>5.76E+03</td>
</tr>
<tr>
<td>3.84E+01</td>
<td>8.82E+00</td>
<td>5.88E+03</td>
</tr>
<tr>
<td>3.93E+01</td>
<td>9.00E+00</td>
<td>6.00E+03</td>
</tr>
</tbody>
</table>

Figure 121 is obtained using the data in the first and second columns of Table 12 to show the variation of spark advance as the injection time varies.

In conclusion, the purpose of this chapter was to show experimentally that the fuzzy logic controllers developed in Chapter V for idle speed control performed satisfactory. The experimental results are shown to be similar to the simulated results obtained in Chapter V. The similarities and, in some cases, the differences between the results obtained here and in Chapter V are to be expected. The results obtained here, however, are valid only under the assumptions, parameters, and experimental variables used for the controllers' design and during the experimentation procedure.
Table 13

Throttle ($\theta$) as a Function of the Error ($e$) and the Error Change ($De$)

<table>
<thead>
<tr>
<th>Throttle Angle (deg.)</th>
<th>De</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.00E+01</td>
<td>1.12E+02</td>
<td>1.20E+01</td>
</tr>
<tr>
<td>6.00E+01</td>
<td>1.28E+02</td>
<td>2.80E+01</td>
</tr>
<tr>
<td>6.00E+01</td>
<td>9.51E+01</td>
<td>1.55E+02</td>
</tr>
<tr>
<td>6.00E+01</td>
<td>9.51E+01</td>
<td>1.55E+02</td>
</tr>
<tr>
<td>5.77E+01</td>
<td>1.12E+02</td>
<td>1.72E+02</td>
</tr>
<tr>
<td>3.32E+01</td>
<td>1.27E+02</td>
<td>1.87E+02</td>
</tr>
<tr>
<td>2.83E+01</td>
<td>1.44E+02</td>
<td>4.44E+02</td>
</tr>
<tr>
<td>9.00E+00</td>
<td>8.00E+01</td>
<td>4.60E+02</td>
</tr>
<tr>
<td>9.00E+00</td>
<td>1.18E+02</td>
<td>4.98E+02</td>
</tr>
<tr>
<td>2.31E+01</td>
<td>1.28E+02</td>
<td>5.08E+02</td>
</tr>
<tr>
<td>4.65E+01</td>
<td>1.44E+02</td>
<td>5.24E+02</td>
</tr>
<tr>
<td>6.00E+01</td>
<td>1.53E+02</td>
<td>5.33E+02</td>
</tr>
<tr>
<td>6.00E+01</td>
<td>1.53E+02</td>
<td>5.33E+02</td>
</tr>
<tr>
<td>9.00E+00</td>
<td>8.00E+01</td>
<td>5.40E+02</td>
</tr>
<tr>
<td>1.15E+01</td>
<td>9.60E+01</td>
<td>5.56E+02</td>
</tr>
<tr>
<td>2.70E+01</td>
<td>1.12E+02</td>
<td>5.72E+02</td>
</tr>
<tr>
<td>4.56E+01</td>
<td>1.28E+02</td>
<td>5.88E+02</td>
</tr>
<tr>
<td>6.00E+01</td>
<td>1.40E+02</td>
<td>6.00E+02</td>
</tr>
<tr>
<td>6.00E+01</td>
<td>1.40E+02</td>
<td>6.00E+02</td>
</tr>
<tr>
<td>6.00E+01</td>
<td>1.44E+02</td>
<td>6.04E+02</td>
</tr>
<tr>
<td>1.92E+01</td>
<td>8.00E+01</td>
<td>6.20E+02</td>
</tr>
<tr>
<td>3.16E+01</td>
<td>9.60E+01</td>
<td>6.36E+02</td>
</tr>
<tr>
<td>4.54E+01</td>
<td>1.12E+02</td>
<td>6.52E+02</td>
</tr>
<tr>
<td>6.00E+01</td>
<td>1.28E+02</td>
<td>6.68E+02</td>
</tr>
<tr>
<td>6.00E+01</td>
<td>1.28E+02</td>
<td>6.68E+02</td>
</tr>
<tr>
<td>6.00E+01</td>
<td>1.28E+02</td>
<td>6.68E+02</td>
</tr>
<tr>
<td>1.88E+01</td>
<td>8.00E+01</td>
<td>7.00E+02</td>
</tr>
</tbody>
</table>
Another important factor that could contribute to the discrepancy between the simulation and experimental results is that the fuzzy controller algorithm is simulated using the math works software, i.e., SIMULINK, Matlab, Fuzzy Logic toolbox, and Control toolbox, while in the work done at ADI the controller algorithm is converted to C language using real time workstation (RTW) and then simulated on the real time station AD RTS, which also used different software platform. The simulation time is different than the experimental time in all of the results presented due to the fact that the experiment time is real time. Also, the difference in writing the assembly language source file and Matlab or C language codes should be noted. Assembly language is different than the Matlab or C language codes, as previously explained in this chapter. During the assembly of the source codes, many variables are introduced into the control algorithms that might affect the value of the controller output. In spite
of all of these factors and other factors that are inherently found in the controller
design, such as the approximate data that are used to form the fuzzy model rule-base,
the controller performed satisfactorily and the results obtained here are comparable
and agreed well with the simulation results.

This concludes the development, simulation, and experimental verifications
of the two fuzzy controllers. It has been shown that the models developed in both
cases have theoretical foundations, as well as practical applications, in an important
branch of nonlinear control theory, which deals with engine management control. In
Chapter VII, the parameter selections and their effects on the performance of the
fuzzy logic controller are discussed.
CHAPTER VII

PARAMETER SELECTIONS AND TUNING GUIDELINES FOR THE FUZZY LOGIC CONTROLLER

Introduction

Fuzzy logic control is the actual process of mapping from a given input to an output using fuzzy logic. The process involves membership functions, fuzzy logic operator, If-Then rule base, and defuzzification. In the fuzzy logic controller brain, there are five parts of fuzzy inference process; (1) fuzzification of the input variables, (2) application of the fuzzy connective operator in the antecedent part of the If-Then rules, (3) implication from the antecedent to the consequent, (4) aggregation of the consequents across the rules, and (5) defuzzification to produce the controller crisp output. In the fuzzification process, the input’s degree of membership in the appropriate fuzzy sets via the membership function is determined. The input in most cases is crisp input, but fuzzy inputs can also be encountered. The range of discourse of the input limits the value of the input. The output of the fuzzification process is a fuzzy degree of membership in the interval of [0, 1]. This step is just like a table lookup or function evaluation. More discussion on fuzzification is stated in the sections that follow. Once the inputs have been fuzzified, then the degree to which each part of the antecedent has been satisfied for each rule becomes known. If the antecedent of a given rule has more than one part, the fuzzy operator or connective
operator is applied to obtain one value that represents the results of the antecedent for that rule. The value obtained is then applied to the output function. The input to the fuzzy connective operator is two or more membership values from the fuzzified input variables. The output is a single truth-value. Implication operators for connectives such as "AND" and "OR" are many and will be discussed in depth later in this chapter. Before applying the implication method, the rule's weight, if different than 1, must be taken into consideration. The rule weight is a number between 0 and 1 and it reflects how certain the rule is if applied to the system. The implication method is defined as the shaping of the consequent; usually it is a fuzzy set based on the antecedent, which is a crisp value. The input for the implication process is a single number given by the antecedent and the output is a fuzzy set. Implication occurs for each rule. Two often-used implication operators are the minimum operator and the product operator, although many other t-norms also can be used, which will be discussed in the section devoted to t-norms and t-conorms operators. The last step before defuzzification is called the aggregation process. Aggregation is when joining the parallel threads unifies the outputs of each rule. This is simply a matter of taking all the fuzzy sets that represent the output of each rule and combining them into a single fuzzy set. The input of the aggregation process is a list of truncated output functions returned by the implication process for each rule. The output of the aggregation process is one fuzzy set for each output variable (see the defuzzification section in Chapter II). The aggregation operators, such as the maximum operator, probabilistic or, and sum of each rule's output set, as well as other aggregation
operators, are also discussed. The last step in the fuzzy inference process is the defuzzification process. The input for the defuzzification process is a fuzzy set that was obtained from the aggregation process, and the output is a single crisp value. Many defuzzification methods are listed in Chapter II. In this chapter, the defuzzification methods are discussed in depth in terms of their effects on the controller performance. Three newly developed defuzzification methods are introduced and simulated/implemented in practical application.

The research presented in this chapter attempts to achieve many goals that are of interest in studying and analyzing the fuzzy controller. The first goal is to study the effects that certain choices of the inference operators have on the performance and stability of the fuzzy controller. In doing so, issues that are important at the initial design process, such as which parameter to choose at different stages of the fuzzy processing unit, can be understood and utilized, saving effort and time in the process. The type of fuzzy controllers that are suitable for certain types of applications can be correctly selected. For example, if the fuzzy control law can be or needs to be expressed analytically, then certain choices and combinations of the operators mentioned above should be selected carefully to yield the desired control law.

The second goal is to provide insight into the work that is needed to automate the design process. The thorough investigation presented in this study will aid future researchers or designers in understanding the necessity and importance of developing a fuzzy control software template, which takes into consideration the complexity, varieties, and selection process of the important design factors. This goal involves
active research in which two research branches that achieve the same goal are pursued. The goal can be satisfied either by building a complete and comprehensive fuzzy controller software design template, or by using some aspects of the already developed or potentially developed generalization models of the parameters that influence the controller performance and stability the most, to formulate an overall procedure for the design of the control algorithm. For example, the adaptive defuzzification method, such as the Semi Linear Defuzzification (SLIDE) method, can be used as the generalized model for the defuzzification process, while the normalized spline-base membership functions developed in Chapter IV can be used as the generalized model for the fuzzification process. Achieving this goal is important, because the design of the fuzzy logic control system, as it exists today, lacks definite sense of intuitive design. Thus, most of the design time is spent on the selection of the different parameters.

The third goal of this study is to develop a standardized way of diagnosing the fuzzy controller when it does not perform satisfactorily based on gained knowledge and experience. This goal in itself is important to achieve, because most of the time, at the initial testing stages of the controller, the performance is not what is expected and many redesign processes are needed to accomplish the task. Therefore, having a good idea of what causes certain performances and responses to behave differently than expected saves time and money.

The main goals of this study are addressed and discussed in the following sections. The branching goals that this study leads one to investigate further will be
the subject of future research elaborated on in Chapter VIII.

Fuzzy Logic Controller Parameters Selection

Parameter selection and tuning of the fuzzy logic controller play an important part of fuzzy logic controller performance and stability. Parameters such as fuzzification, connective operator, implication, aggregation, and defuzzification process can be selected from a pool of many parameters. In this chapter, the parameters that will be considered are limited, but the scope of the subject matter is limitless. Many methods and choices are available for each one of the fuzzy logic controller-processing units. The choice of the membership function affects the fuzzification process, the choice of connective operators affects the controller output and stability, and the choice of the fuzzification process influences the controller output and the processing speed and accuracy. Previous work in this area can be seen in Braae and Rutherford (1979), in which a study of the effects of the rule and the fuzzy set choices for the inputs and outputs universe of discourse was discussed. The effects of the choice of scaling factor and the input noise on the membership functions were also considered. Other important studies are referenced in the relevant sections of this study.

In the following sections, each choice of the parameter and its effects are discussed separately. To show the effects of choosing different parameters and their combinations, a case study that involves controlling the inverted pendulum is presented at the end of this chapter.
Membership Function Selections

The membership function choice influences the fuzzification value. Many shapes and methods can be utilized to construct membership functions, provided that certain criteria are met. Each membership function can be expressed mathematically by a linear piecewise function or nonlinear continuous function. If a function $f$ is a membership function, then $f$ can be defined on the range $[a, b]$ such that the function $f$ achieves its maximum for any element in the range of $f$ at a value of 1, i.e., if $x \in [a, b]$ then $\text{Max}\{f(x)\} \leq 1$, for at least one $x$ in $[a, b]$. If $\text{Max}\{f(x)\}$ for any one $x$ is equal to 1, then the membership function represented by $f$ is said to be a normal fuzzy membership function, as was discussed in Chapter II. It is also an important requirement that if $f$ is to be a fuzzy membership function, then the value of $f(x)$ for $x \in [a, b]$ is not equal to 0, which means that all the elements of the membership support have non-zero membership value; otherwise, the function is not a fuzzy membership function. It should be pointed out that the membership function for the elements on the border of the range can be zero, i.e., $f(a)$ and $f(b)$ can be zero. All of the membership function types that were considered in previous chapters satisfy these requirements. The question becomes which type or shape of the membership function should one use in the fuzzification process, or which type yields better controller performance? The answer to this question is a research topic in itself. Many previous works dealt with such issues and the conclusions reached were not that much different from one another, stating that the choice of the membership
function is application dependent. There are many factors that one should consider in choosing the membership function. Some of these factors involve the simplicity and the ability to express the membership function analytically. Other factors concern implementation and computational speed. Some membership function types restrict the ability of expressing the fuzzy control algorithm into a generalized analytical form, such as triangular or trapezoidal membership functions. But they are the most often used membership functions because of their simplicity and ease of implementation. Previous work in this regard can be seen in Pedrycz (1993), in which the author studied the theoretical foundation behind choosing triangular and trapezoidal membership functions. Figure 122 compares the effects on the fuzzified value for four types of membership functions: triangular, trapezoidal, Gaussian, and \( \pi \)-shape. The fuzzified value is different from one type to another, as can be seen by the arrows in Figure 122.

The effects on the controller performance can be apparent, if one simply considers the membership factors with all other parameters held constant; then the fuzzy controller performance can be seen to differ widely, as shown in Figure 123. The application of the fuzzy logic controller can dictate the type of the membership function to be used. For example, the Gaussian membership function can always be used in any type of fuzzy logic control, such as adaptive and neuro-fuzzy control schemes, in addition to other conventional fuzzy control. Usually, triangular membership functions are a good starting choice; they make the design and implementation process very simple and give a good indication of the controller
performance. Functional forms of membership functions are good choices in situations where the stability of the fuzzy logic controller needs to be carefully examined. If the fuzzy logic controller is judged to be stable for one type of membership function, it is in no way a guarantee of stability if other types of the membership function are used.

Figure 122. The Fuzzified Value for the Same Input With Different Membership Functions Shapes.

The stability of the fuzzy logic controller has been extensively investigated based on the ability to express the control law analytically, but no such standardized method to investigate stability for other forms of fuzzy control exists. Some authors investigated the stability of the designed fuzzy controller by eliminating rules and changing scaling factors and alike. This type of investigation does not prove the stability of the control system, since the number of membership function choices defined on the
the rule-base is just modifying the number of membership functions. Some plants can be controlled using as few as three fuzzy rules, and just increasing the numbers over three will slightly affect the performance. Thus, if in the initial design stage the number of rules is higher than necessary, then reducing the number of rules does not prove stability but can show that some of the rules might be redundant, or they may have no effects on the control system. Therefore, a membership functions template, with the ability to be incorporated into the control algorithm, should be a part of any control system design tool box, so that looping with the template becomes a part of an adaptive control loop that can be judged based on controller output criteria. The control algorithm should select the membership function type that achieves the controller output criteria. In this way, the control system design becomes more concentrated on the other important factors, thus reducing the time of the design and
increasing the controller adaptability. The effects of the membership function shape will be illustrated in practical application, using the inverted pendulum as a case study later in this chapter.

**Connectives and t-Norms and s-Norms**

In this section, the choice of the operator used for connectives such as "AND" and "OR" is studied and the effects on the fuzzy logic controller design, implementation, and performance are discussed. The connective operators have been studied in depth, on the basis of both mathematical foundation and control theory application. A comprehensive study was conducted by Misumoto (1989), who discussed many types and forms of t-norms and s-norms (also called t-conorms), as well as the generating function for both t-norms and t-conorms. Ruan (1993) studied the influence of fuzzy implication operators on the method-of-cases-inference rule. Ruan (1993) also conducted an in-depth mathematical discussion and extension of the triangular norm to n-argument. The t-norms are used for the connective "AND," while the t-conorms are used for the connective "OR." For example, if a fuzzy rule is expressed as:

If $x$ is $A$ "AND" $y$ is $B$ "OR" $z$ is $C$ the $u$ is $D$

and the membership functions for $x$, $y$, and $z$ are given by $\mu_A(x)$, $\mu_B(y)$, $\mu_C(z)$, respectively. Then the degree of certainty or the truth-value using t-norms and t-conorms operators can be expressed as:
where \( t \) and \( s \) represent any of the \( t \)-norms and \( s \)-norms (\( t \)-conorms) operators.

The criteria that a \( t \)-norm should satisfy before it can be used as a connective operator is discussed in depth in a study conducted by Misumoto (1989) and can be stated mathematically as:

A function \( t : [0, 1] \times [0, 1] \rightarrow [0, 1] \) is a \( t \)-norm if and only if for any \( a, b \in [0, 1] \) the following holds:

1. \( t(a, 1) = a \)
2. \( t(c, d) = t(a, b) \)
3. \( t(a, b) = t(b, a) \)
4. \( t(a, t(b, c)) = t(a, b), t(b, c)) \)
5. \( t(0, a) = 0 \) \( (7.2) \)

If a \( t \)-norm, in addition to the above stated properties, is continuous and \( t(a, b) < a \), for all \( a \in (0, 1) \), then the \( t \)-norm is called an Archimedian \( t \)-norm. Every Archimedian \( t \)-norm is representable by a continuous and decreasing function \( f \) from \([0, 1] \) into \([0, \infty) \) with \( f(1) = 0 \) such that:

\[
t(a, b) = f^{-1}(f(a) + f(b)) \quad (7.3a)
\]

where \( f^{-1} \) is the inverse of \( f \) defined by:
The function $f$ is called the additive generator of $t$-norms.

On the other hand, the $s$-norm ($t$-conorm) denoted by $s$ should satisfy the following properties:

1. $s(a, 0) = a$
2. For $c < d$, $s(c, b) < s(d, b)$
3. $s(a, b) = s(b, a)$
4. $s(a, s(b, c)) = s(s(a, b), c)$
5. $s(1, a) = 1$ \hfill (7.4)

The $s$-conorm can be represented in a continuous and increasing function $g$ as:

$$s(a, b) = g^{-1}(g(a) + g(b)) \hfill (7.5a)$$

where the inverse of $g$ is given by:

$$g^{-1}(b) = \begin{cases} g^{-1}(b) & b \in [0, g(1)] \\ 1 & b \in [g(1), \infty) \end{cases} \hfill (7.5b)$$
Table 14
Common \( t \)-Norns and \( t \)-Conorms Operators

<table>
<thead>
<tr>
<th>t-norm and its generating function</th>
<th>t-conorm and its generating function</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Logical Min</strong></td>
<td><strong>Logical Sum</strong></td>
</tr>
<tr>
<td>( a \land b = \text{Min}(a, b) )</td>
<td>( a \lor b = \text{Max}(a, b) )</td>
</tr>
<tr>
<td>( f(a) = \begin{cases} 1 &amp; x = 0 \ 0 &amp; x &gt; 0 \end{cases} )</td>
<td>( g(a) = \begin{cases} 0 &amp; x = 0 \ 1 &amp; x &lt; 0 \end{cases} )</td>
</tr>
<tr>
<td><strong>Hamacher product</strong></td>
<td><strong>Hamacher sum</strong></td>
</tr>
<tr>
<td>( a \odot b = \frac{ab}{a + b - ab} )</td>
<td>( a \oslash b = \frac{a + b - 2ab}{1 - ab} )</td>
</tr>
<tr>
<td>( f(a) = \frac{1-a}{a} )</td>
<td>( g(a) = \frac{a}{1-a} )</td>
</tr>
<tr>
<td><strong>Einstein product</strong></td>
<td><strong>Einstein sum</strong></td>
</tr>
<tr>
<td>( a \oplus b = \frac{ab}{1 + (1-a)(1-b)} )</td>
<td>( a \oplus b = \frac{a + b}{1 + ab} )</td>
</tr>
<tr>
<td>( f(a) = \log\left[ \frac{2-a}{a} \right] )</td>
<td>( g(a) = \log\left[ \frac{1+a}{1-a} \right] )</td>
</tr>
<tr>
<td><strong>Bounded product</strong></td>
<td><strong>Bounded sum</strong></td>
</tr>
<tr>
<td>( a \cdot b = \text{Max}(0, a + b - 1) )</td>
<td>( a \prec \succ b = \text{Min}(1, a + b) )</td>
</tr>
<tr>
<td>( f(a) = 1 - a )</td>
<td>( g(a) = a )</td>
</tr>
<tr>
<td><strong>Drastic product</strong></td>
<td><strong>Drastic sum</strong></td>
</tr>
<tr>
<td>( \begin{cases} a &amp; b = 1 \ a &amp; a = 1 \ 0 &amp; a, b &lt; 1 \end{cases} )</td>
<td>( \begin{cases} a &amp; b = 0 \ b &amp; a = 0 \ 1 &amp; a, b &gt; 0 \end{cases} )</td>
</tr>
<tr>
<td>( f(a) = \begin{cases} 1 &amp; x &lt; 1 \ 0 &amp; x = 1 \end{cases} )</td>
<td>( g(a) = \begin{cases} 0 &amp; x &lt; 1 \ 1 &amp; x = 1 \end{cases} )</td>
</tr>
<tr>
<td><strong>Algebraic product</strong></td>
<td><strong>Algebraic sum</strong></td>
</tr>
<tr>
<td>( a \odot b = ab )</td>
<td>( a \oplus b = a + b - ab )</td>
</tr>
<tr>
<td>( f(a) = -\log a )</td>
<td>( g(a) = -\log(1-a) )</td>
</tr>
</tbody>
</table>
Some of the most commonly used $\mu$-norms and $\mu$-conorms operators are listed in Table 14. Many other $\mu$-norms and $\mu$-conorms can be seen in the paper published by Mizumoto (1989). A new $\mu$-norms and $\mu$-conorms operator can be generated if the additive generator function can be found. To construct the additive generators of the $\mu$-norms and $\mu$-conorms two methods are usually used (Misumoto, 1989). The two methods are as follows:

1. If $f(a)$ is an additive generator for a $\mu$-norm, then a new additive generator can be constructed by:

$$F(a) = 1 - f(1 - a) \quad (7.6)$$

So, for example, if the additive generator is given by $f(a) = (1 - a)^m$ then $F(a) = a^m$.

The same thing can be said about finding a new additive generator for $s$-norm or $\mu$-conorm, which in this case is given by:

$$G(a) = 1 - g(1 - a) \quad (7.7)$$

where $g$ is the generator for $\mu$-conorm.

2. For any additive generator $f(a)$ and $g(a)$, a new additive generator for the $\mu$-norm can be found using the relation $F(a) = f(g(a))$ and for the $\mu$-conorm using the relation $G(a) = g(f(a))$. For example, if $f(a) = (1 - a)^m$ and $g(a) = 1 - a$, then $F(a) = a^m$.

Figure 124 shows the effects of different $\mu$-norms on the controller performance. Other $\mu$-norms operators generated by a generator function mentioned
above will fall within the range shown in Figure 124.

Figure 124. Controller Output as a Function of \( t \)-Norms.

Defuzzification Method Effects on the Controller Performance

In fuzzy logic control systems, the defuzzification process involves the selection of one value as the output of the controller. Starting with a fuzzy subset \( C \) over the output universe of discourse \( U \) of the controller, the defuzzification step uses this fuzzy subset to select a representative element \( u \), as was discussed in Chapter II. Many defuzzification methods are in use nowadays. The most common defuzzification methods that have been used are the center of area (COA), which is also called the Centroid defuzzification method, and the Mean of Maxima (MOM) method. Other new methods have been introduced and many others can be constructed, some of which are discussed here. Yager and Filev (1993) introduced a parameterized family of defuzzification operators called the Semi Linear...
Defuzzification (SLIDE) method. The SLIDE defuzzification method was based on a simple transformation of the fuzzy output set of the controller, i.e., given the controller output universe of discourse $U$, another fuzzy set $V$ can be obtained from $U$ using the transformation given by:

$$
T_{\alpha, \beta} : U \rightarrow V
$$

$$
v_i = \begin{cases} 
  w_i & w_i \geq \alpha \\
  (1 - \beta)w_i & w_i < \alpha 
\end{cases}
$$

(7.8)

where $v_i$ is the membership function grade in $V$, derived from the $w_i$ using the transformation, and $\alpha, \beta$ are parameters of transformation such that $\alpha \in [0, M]$ and $\beta \in [0, 1]$. $M$ is defined by $M = \max_i \{w_i\}$ and $w_i$ are the membership function grade in $U$. The choice of the parameters $\alpha, \beta$ affects the defuzzified values obtained by the SLIDE method. For certain values of $\alpha, \beta$, the SLIDE method is reduced to the well-known methods such as COA and MOM, i.e., if $\alpha = 0$ and $\beta = 0$, then $v_i = w_i$, which is the COA defuzzified method. If $\alpha = \max_i \{w_i\}$ and $\beta = 1$, then the SLIDE method becomes the MOM method. This can be seen in Equation (7.9):

$$
x^* = \frac{(1 - \beta) \sum_{i \in L} w_i x_i + \sum_{i \in H} w_i x_i}{(1 - \beta) \sum_{i \in L} w_i + \sum_{i \in H} w_i}
$$

(7.9)

where $L = \{i/w_i < \alpha, i = 1, 2, \ldots n\}$ and $H = \{i/w_i \geq \alpha, i = 1, 2, \ldots n\}$.

Oliveira (1995) proposed a more general defuzzification method that also has
the COA and MOM defuzzification methods as special cases. The method was based on subsethood and, therefore, it is called the subsethood defuzzification (SD) method. The method uses the cardinality measure of the fuzzy universe of discourse and the mirror set of its support, which is stated mathematically by:

\[ x^* = \frac{M(n_X)}{M(X)} \quad (7.10a) \]

where \( n_X \) is the mirror set of the support of \( X \), and it is given by:

\[ n_X = \sum_{i=1}^{n} \frac{\mu_{n_X}(x_i)}{x_i} \quad (7.10b) \]

The function \( M(X) \) in Equation (7.10a) is given by:

\[ M(X) = \sum_{i=1}^{n} \mu_X(x_i) \quad (7.10c) \]

For two fuzzy sets \( A \) and \( B \), their intersection using any of the \( \cap \)-norms operators is given by:

\[ \mu_{A \cap B}(x) = \mu_A(x) \mu_B(x) \quad (7.10d) \]

Therefore, if \( A \) is a fuzzy set, then the defuzzified value of \( A \) can be expressed using the SD defuzzified method, as shown in Equation (7.11a):
\[ x^* = \frac{M_p(A \cap \mathbb{R})}{M_p(A)} = \frac{\sum_{i=1}^{n} (\mu_A(x_i) x_i)^p}{\sqrt{\sum_{i=1}^{n} \mu_A^p(x_i)}} \]  

(7.11a)

where

\[ M_p(A) = \sqrt[2p]{(\mu_A(x_1) + \ldots + \mu_A(x_n))} \]  

(7.11b)

Thus, if the t-norms is taken to be the product operator, then the SD method becomes the COA method; while, if instead of considering all the elements in the output fuzzy set A only strong \( \beta \)-cut of A at \( \beta = 1 \) is considered, then the SD becomes the MOM method. Saade (1996) used a defuzzified index in ranking to overcome the shortcomings of other defuzzification methods based on the Hurwicz criterion, thus reducing the probabilistic aspects of the developed method. These are some of the most recent studies that dealt with defuzzification methods, from which one can see the author's attempt to unify or standardize the defuzzification process based on some sort of mathematical foundation. This, however, has not been achieved yet, since each approach is applicable within the certain limitation of its based foundations. However, the defuzzification methods based on pure mathematical foundations have been extensively investigated and are close to being unified. On the other hand, the practical applications are still lagging behind the mathematical development of defuzzification methods, since most control engineers are still using more familiar methods that have proven to be functional and have
achieved good results. Figure 125 shows the effect of different defuzzification methods on the controller output. The defuzzification methods play an important part in the overall inference system. The crisp value obtained is the direct input value to the plant so it has a large effect on the stability and performance of the controller. As can be seen from Figure 125, the values of the output differ greatly from one method to another. It should be pointed out that some of the defuzzification methods are not suitable for certain applications and should be avoided.

![Diagram of defuzzification methods](image)

Figure 125. Effects of Different Defuzzification Methods on the Controller Output.

In Chapter II, a list of seven most often used defuzzification methods were discussed and stated, both mathematically and graphically. The newly developed methods, in one form or another, are constructed based on these methods with certain assumptions and modifications, since most newly developed methods have at least
one of their basic defuzzification methods as a special case. The criteria for choosing a certain method over another can be stated, taking into consideration many factors. Hellendoorn and Thomas (1993) have specified four criteria to use when choosing a defuzzification method. The first criterion is continuity. A small change in the input of a fuzzy process should not produce a large change in the output. The second criterion is the disambiguity, which means that a defuzzification method should always result in a unique value with no ambiguity. For example, the center of largest area method stated in Chapter II cannot satisfy this requirement, because when the largest membership functions have equal area, then there are two values for this defuzzification method. The third criterion is called plausibility. Plausibility means that the defuzzified value should lie approximately in the middle of the support region of the output membership function and should have the highest degree of membership in that support. For example, the COA method does not satisfy this criterion, because, although the defuzzified value lies in the middle of the support, it does not have the highest degree of membership. The fourth criterion is computational simplicity, which suggests that the more time-consuming a method is, the less value it should have in a computation system. For example, the MOM method is faster than the COA method. This criterion nowadays is less important than it used to be years ago because of the technological advancement in both hardware and software system design. Another criterion that can be added based on the recent studies in this regard is the adaptability criterion, which indicates the ability to incorporate the defuzzification method into an adaptive fuzzy control scheme such as
the SLIDE method. Above all, the defuzzification method should be assessed more in
terms of the controller performance than any other factor.

In the following sections, the overall effects of the parameter selection on a
fuzzy controller controlling a real plant are illustrated using the inverted pendulum as
a special case study. The study is conducted in order to assess the stability of the
plant under different combinations of parameter choices. The inverted pendulum
used in this illustrative case is shown in Figure 126. The inputs to the controller are
the vertical angle and its derivative. The dynamic equations governing the inverted
pendulum system are given by Equation (7.12):

\[
\begin{align*}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= \frac{g \sin x_1 - \frac{ml^2 \cos x_2 \sin x_1}{m_c + m}}{L} + \frac{\cos x_1}{L} \frac{m_c + m}{4} u \\
\end{align*}
\]

(7.12)

where \( x_1 = \theta, x_2 = \dot{\theta} \) and the values used for \( L, m_c, m, g \) are 0.5 \( \text{m} \), 1 \( \text{kg} \), 0.1 \( \text{kg} \), and 9.8 \( \text{m/s}^2 \), respectively. The fuzzy logic controller rule-base used in the simulations is
given in Table 15.

![Figure 126. Block Diagram of Proposed Simulation Model for the Inverted Pendulum
With a Fuzzy Logic Controller, Which Employs the Rules in Table 15.](image-url)
Table 15

Fuzzy Rules for the Inverted Pendulum Used for Simulation

<table>
<thead>
<tr>
<th>$\theta$</th>
<th>$\theta'$</th>
<th>NB</th>
<th>NS</th>
<th>Z</th>
<th>PS</th>
<th>PB</th>
</tr>
</thead>
<tbody>
<tr>
<td>NB</td>
<td>Z</td>
<td>NS</td>
<td>NM</td>
<td>NB</td>
<td>NB</td>
<td></td>
</tr>
<tr>
<td>NS</td>
<td>PS</td>
<td>Z</td>
<td>NS</td>
<td>NM</td>
<td>NB</td>
<td></td>
</tr>
<tr>
<td>Z</td>
<td>PM</td>
<td>PS</td>
<td>Z</td>
<td>NS</td>
<td>NM</td>
<td></td>
</tr>
<tr>
<td>PS</td>
<td>PB</td>
<td>PM</td>
<td>PS</td>
<td>Z</td>
<td>NS</td>
<td></td>
</tr>
<tr>
<td>PB</td>
<td>PB</td>
<td>PB</td>
<td>PM</td>
<td>PS</td>
<td>Z</td>
<td></td>
</tr>
</tbody>
</table>

The simulation is conducted for different situations and combinations of the fuzzy logic parameters. The simulation results for step input to the system are now presented for different sets of combinations.

Figure 127 shows the system response when triangular membership functions are used with the product for connective “AND,” product for implication, maximum for aggregation, and COA for defuzzification. The system appears to respond very well and attains zero steady state error with moderate overshoot.

Figure 128 shows the response when trapezoidal membership function are used. For this choice of membership function, the system response with the same implication, aggregation, defuzzification methods as used in Figure 127 deteriorated. The figure shows oscillatory response with a large steady state error.

Figure 129 shows a good response for ramp input when sigmoid membership functions are used. A short settling time and zero steady state error can be seen in the figure. The expression for the sigmoid membership function is stated in Chapter II.
Figure 127. System Response for Triangular Membership Functions, Product for Connective, Product for Implication, Maximum for Aggregation, and COA Defuzzification.

Figure 128. System Response for Trapezoidal Membership Function, Product for Connective, Product for Implication, Maximum for Aggregation, and COA for defuzzification.
Figure 129. Response when Sigmoid Membership Functions Are Used.

Figure 130 shows the response when, for each linguistic variable, a different membership function shape is used. For example, for three linguistic variables, negative big (NB), negative medium (NM), and negative small (NS), the linguistic variable NB can be represented by triangular membership function, NM by Gaussian membership function, and NS by Sigmoid membership function. The response of the system becomes unstable as can be seen in the figure. The membership functions chosen are bell-shape, Gaussian, and triangular, and they were distributed symmetrically around the linguistic value zero (Z).

The effects of choosing different defuzzification methods on the system are shown in Figure 131. From the figure, a difference in the response can be noticed in the amount of the overshoot and the magnitude of the steady state error.
Figure 130. Response of the System when Different Membership Functions (Bell-Shape, Gaussian, Triangular) are used.

It appears from Figure 131 that even though the subsethood defuzzification (SD) method appears to perform with minimal steady state error and a large overshoot, other methods share the same amount of overshoot but differ in the magnitude of the steady state error. This show that the newly developed defuzzification methods discussed earlier are good and are more practical methods, even though they have not been used in any practical application yet.

To show the defuzzification effect more clearly, Figures 132, 133, and 134 are simulated for three different defuzzification methods: Mean of Max (MOM), Center of Area (COA), and Center of Sums (COS).

Figure 132 shows the response when MOM is used. The system did not respond well to this method and it became unstable. Figure 133 shows the response when the COS is used; again, the system became unstable.
Figure 131. The Results of Simulation with Different Defuzzification Methods.

Figure 132. Response of the System when (MOM) Is Used.
Figure 133. Response of the System When COS Is Used.

Figure 134 shows the response when the COA is used. The figure shows a good response with a minimum overshoot, a zero steady state error, and a relatively short settling time.

Figure 134. Response of the System when the COA Method Is Used.
In Figure 135, the connective operators are compared while other parameters are fixed. The figure shows the Schwiezer and Sklar $t$-norms operators performed the best for this situation. The other $t$-norms operators yield different overshoot and steady state error, but not much difference between them can be pointed out. No conclusion can be drawn based on this result. The $t$-norms operator has a large effect on the controller performance, where, in my experience, the whole control surface can be modified simply by changing the $t$-norms operator. The availability of different $t$-norms and $t$-conorms makes the choice difficult. To overcome this difficulty, the $t$-norms, in addition to the membership functions, can be included in an adaptive scheme.

Figure 136 shows the comparison for the implication operators, while other parameters are fixed. Two most often used implication operators are included here; the others are omitted due to the similarities in their performance. The Min-operator

![Figure 135: The Simulation Results for Different t-Norms Operators.](image)
performs much better than the product operator and the other operators, with higher
overshoots but minimal steady state error, but no conclusion can be drawn on this
comparison since other operators in different applications perform much better than
the minimum operator. It may be advisable that the last thing to tune when tuning
fuzzy logic parameters is the implication operator if everything fails to change the
performance. It is difficult to separate the effects of the implication operator from the
other factors, since it is influenced by the choice of the membership functions for the
inputs and outputs and the scaling factors.

Three aggregation operators are compared, as shown in Figure 137. The three
aggregation operators performed in similar fashion, which indicates that the
aggregation operator has a little, if any effect on the controller performance in this
case.

Figure 136. The Implication Operators Comparison. Only Min and Product Operators
Are Shown.
Different combinations of the fuzzy inference parameters are then simulated, and the results are shown in Figures 138 through 142. In Figure 138, the combinations shown are as follows: "1"—combines the Minimum operator for connective "AND," Minimum for implication, Maximum for aggregation, and COA for defuzzification; "2"—combines a Minimum for connective "AND," Product for implication, Max for aggregation, and COA for defuzzification; and "3"—combines the same parameters as 1 but with the sum for the aggregation method. The figure shows that combinations 1 and 2 perform better than the other combinations.

Figure 139 shows a similar combination as Figure 138, except that the defuzzification method used for all combinations is the SD defuzzification method. In Figure 139, combination 1, i.e., Min for connective "AND," Min for implication, Max for aggregation, and SD for defuzzification, performed the best.
Figure 138. Combinations of Different Connectives, Implication, Aggregation, and Defuzzification Methods.

Figure 140 shows the controller performance with combinations as follows:

"1"—Min t-norms for connective “AND,” Min for implication, Max for aggregation, and MOM for defuzzification; "2"—same as 1 with Product for implication; and "3"—same as 1 with Sklar for connective and sum for aggregation. The figure shows that combinations 1 and 2 performed the same. Nothing can be concluded from this figure except that the Product and Min implication operators have little effect on the overall performance of the system, as was stated earlier. The initial response usually is very critical in determining the stability of the system. Therefore, combinations 1 and 2 performed best and should be selected if and only if the initial response is taken to be the critical factor in determining the system response.
Figure 139. Same Combination as in Figure 138 but With SD Defuzzification.

Figure 140. The Little Effect of Changing the Implication Operator while the Combination of Other Parameters Stayed the Same.
Figure 141 shows a set of combinations indicated by: “1”–the Sklar is used for the connective “AND,” Product for implication, Max for aggregation, and MOM for defuzzification; “2” represents the combination as in 1, except the Product for connective “AND,” and sum for aggregation are used; “3” is the same as 2 except the COA is used for defuzzification and Max for aggregation. The figure shows that combinations 1 and 2 performed the same, and combination 4 produced an unstable response. This shows that for certain combinations of these operators, one can have an unstable system. The reason the system becomes unstable may be due more to the defuzzification method than any other factor, since a small change in any other fuzzy controller’s parameters should not produce this large a change if the defuzzified method is a good one. This problem can shed light on the defuzzification process and give reasons for the desire to find a defuzzification method that can modify its parameters according to changes in other factors. This attempt was clear in the SLIDE defuzzification method, since the parameters $\alpha$ and $\beta$ can be made adaptive within the control algorithm. But to do so produces a computationally expensive, time consuming, and very complicated control algorithm.

Figure 142 shows three combinations; combination 1 consists of using Hamacher t-norm for connective “AND,” Product for implication, Max for aggregation, and SLIDE for defuzzification. Combination 2 consists of Sklar for t-norms, Product for implication, sum for aggregation, and Center Of Sum (COS) defuzzification. Combination 3 consists of the same implication and aggregation
Figure 141. Instability In Combination 4 With COA Defuzzification Method.

Figure 142. System Response for Three Combination of Parameters.

operators as combination 2 with MOM defuzzification, and Yager t-norms operator for connective "AND." The difference in responses between the three combinations is noticeable only for a short period of time, but as time goes on, the differences
becomes smaller. Since in most control problems the initial response is very critical in determining the stability of the system, combination 2 in the figure provides the best overall response.

Guidelines for Fuzzy Logic Controller Design

Fuzzy logic controller design is still, in most cases, built on intuition and experience. Recent advancements in fuzzy logic opened the door for design process to become a more systematic and follows some guidelines and investigative methods, as is the case in classical control. The stability of a fuzzy controller was a major issue in the earlier stages of the development of fuzzy logic control. Several industries held back from using fuzzy logic control because of the difficulty in studying its stability. Recent researches into this area have given promising results and have been applied successfully in the investigation of fuzzy logic controllers. Other obstacles, such as the lack of a strong mathematical foundation and the ability to express the control law into a functional or analytical form, have also been solved, which has given the design engineer the confidence and ability to modify, redesign, and test the design control system using system analysis techniques, simulation, and laboratory verifications. It is no wonder nowadays that many industries are using fuzzy logic systems in control. On the horizon are many applications for which fuzzy control can be ideal, and the potential for fuzzy logic control in all aspects discussed in the introductory chapter are limitless. However, there are still questions that one needs to answer before considering using the fuzzy logic controller—questions such
as, should one use fuzzy logic control if classical control theory can be applied satisfactorily to a specific application? The answer to this question is subjective rather than objective. Many factors should be considered before answering such a question. Factors such as time, cost, performance, and hardware and software availability should be considered. In most authors' and experts' opinions, fuzzy logic should never be used if conventional methods can be applied successfully. This researcher differs with this assertion, believing that the fuzzy logic controller, if (a) designed based on analytical basis; (b) investigated for stability, controllability, observability, and optimality; and (c) tested both in simulation and the laboratory, performing satisfactorily, should be preferred over any conventional control techniques for the simple reason that the fuzzy logic controller is a very flexible control system that can be used with small modifications for a wide variety of applications, rather than specific applications as is the case in conventional control. Another question that might be asked is when should the fuzzy controller be applied? To answer this question, some facts about the control system that can benefit from fuzzy logic need to be stated. If the control system of the plant can be considered a black box with outputs available for measurement for wide range of changing inputs, and if the plant is controllable and observable with some information about the plant operation or control available, which can be formulated as a set of rules using all available tools, then an acceptable fuzzy control solution that satisfies the design specifications is possible. In general, fuzzy logic control should be avoided if conventional control theory yields satisfactory results based on a simple, solvable,
and adequate mathematical model of the plant. In Chapters II and III, many types and methods of designing fuzzy control systems were discussed. The design of a specific type of fuzzy control system is application dependent. For example, if only an input/output data set is available, then one can use the method outlined in Chapter III to design a fuzzy logic controller using different techniques. In situations where only a simple fuzzy controller is needed, then a one-level controller structure with no hierarchical rule structure can be enough. If the system is complex and needs a multilevel fuzzy control scheme, then a multilevel control structure with a few controllers at a level hierarchical rule structure is required. On the other hand, if an adaptive or supervisory fuzzy controller is needed, then usually a one-level controller structure with no hierarchical rule structure is good enough. A multilevel controller structure with a few controllers at a level with hierarchical rule structure can also be used in adaptive schemes. Fixed operational procedures that includes the approximate reasoning, fuzzification, and defuzzification should be used for all types of fuzzy control. Fixed input and output scaling factors are used for simple and complex/multilevel controllers, while in adaptive type controllers the scaling factors can be tuned and modified. Fixed input and output should be known for all types of fuzzy controllers, but in the case of adaptive fuzzy controllers, the inputs and outputs can be changed. The number of rules is very much dependent on the number of inputs, except in the case of adaptive fuzzy controllers, where the number of rules can be modified according to specific requirements. It should be stressed that the parameters, shape, and position of membership function in adaptive control can be
tuned and modified.

The structure of fuzzy logic controllers can be constructed using a hierarchical structure whenever there is any doubt in the stability of a fuzzy control system or in applications requiring high reliability. The procedure of choosing the inputs and outputs variables is no different from the procedure for conventional control methods.

**Scaling Factors**

In specifying the universe of discourse, one must first determine the applicable range for linguistic variables in the context of the overall system design. The range should be carefully selected. If the range chosen is too small, then regularly occurring data will be off the scale, which will impact the overall system performance. If the range of the input chosen is too large, then the number of membership functions should be increased to accommodate the range width; also, the membership function width and overlap are affected. Because of this, it is usually desirable and often necessary to scale or normalize the universe of discourse for the input and output variables. Normalization in other studies means applying the standard range of [-1, 1] for the inputs/outputs universes of discourse. In this study, however, the normalization reference means applying the standard range of [-1, 1] or [0, 1] for the inputs/outputs universes of discourse. An input scaling factor transforms a crisp input into a normalized input in order to keep its value within the universe of discourse. An output scaling factor provides a transformation of the defuzzified crisp output from the normalized universe of discourse of the controller output into an
actual physical output (see Chapter III for more discussion on the normalized universes of discourse and the scaling factors). If the choice of the scaling factor is not a good one, the actual operating area of the inputs will be transformed into a very narrow subset of the normalized universe and some input value will be saturated and that will cause the system to become unstable. The output-scaling factor has even greater effect on the fuzzy logic controller gain in closed loop systems and can influence the stability of the system. In the design processes, the priority of scaling is given to the output-scaling factor because of its high influence on stability and oscillation tendency of the system. The input scaling factor is given the second priority, since its influence is most noticeable on the basic sensitivity of the controller with respect to the optimal choice of the operating range of the input signals.

Knowledge acquired from conventional control methods, such as proportional plus derivative (PD), proportional plus integral (PI), and proportional plus integral plus derivative (PID), can be utilized to find the scaling factors of the fuzzy controller inputs and outputs.

Yager (1994) conducted a comparison between the parameters of the PID and the scaling factors of the fuzzy logic controller. The similarities between them are very apparent, as can be seen in Figure 143. The similarity between the factors is expressed as:

\[
K_I = (K_uK_z)
\]

\[
K_P = K_P(K_uK_e)
\]

\[
K_D = K_D(K_uK_{de})
\]
Figure 143. Similarities between FLC Scaling Factors and PID Parameters.

The following relations can be inferred from the comparison:

1. Increasing or decreasing the fuzzy logic controller output scaling factor causes an increase or decrease in both coefficients in the PD, PI, and PID controllers. For example, in the case of the PI controller, this defines the gain factor of the fuzzy controller.

2. Increasing or decreasing the input scaling factor $K_c$ results in an increase or decrease in $K_p$.

3. Increasing or decreasing the input scaling factor $K_e$ causes an increase or decrease in $K_d$.

The PID tuning techniques can be used to tune the scaling factors of the fuzzy logic controller. For example, if the controller response is steadily diverging, then decreasing the scaling factor of the input and output is necessary. If high overshoot and oscillatory response dominate the system response, then reducing the input and
output-scaling factors by different degrees can solve this problem. If the system requires a fast response time, then the scaling factor for the error must be decreased, while the error derivative scaling factor must be increased. For zero steady state error or a very strict requirement on the steady state error, the error scaling factor can be tuned up and down while reducing the derivative of the error scaling factor.

Another parameter that can be used along with the scaling factor is the overlap factor. Overlapping means simply the common area or elements of the two adjacent fuzzy linguistic values. Overlap plays an important factor in the fuzzy controller performance. Narrow membership functions, which result in smaller overlap also results in a faster response of the system. On the other hand, narrow membership functions produce large oscillation, overshoot, settling time, and low steady state error. The overlapping factor ($O_r$) can be assigned at the initial design process and can be adjusted later, if desired. Different methods have been developed to define the overlap factor. Reznik (1997) proposed the following for three types of membership functions. For triangular membership function expressed by:

$$
\mu(x) = 1 - \frac{|x - m|}{\sigma}
$$

(7.14a)

where $m$ and $\sigma$ are the membership parameter. The whole overlap ($O_r$) ratio is given by Equation (7.14b):
For exponential membership function (Gaussian), the $O_r$ ratio is expressed by:

$$O_r = \frac{1}{\frac{2}{\left(1 - \frac{m_2 - m_1}{2\sigma}\right)^2} - 1} \quad (7.14b)$$

For the quadratic membership function expressed by:

$$\mu(x) = \text{Max}(0, 1 - \left|\frac{x - m}{\sigma}\right|^2) \quad (7.16a)$$

The $O_r$ is given by:

$$O_r = \frac{2}{3} \frac{m_2 - m_1}{2\sigma} + \frac{1}{3} \left(\frac{m_2 - m_1}{2\sigma}\right)^3 \quad (7.16b)$$

In the parameterized membership functions discussed in Chapter IV, the parameters $\alpha$, $\beta$, and $\rho$ define the overlap, as was explained. In addition, in the normalized spline-base membership functions discussed in Chapter IV, the number of knots, the degree of the spline function, and the order of the normalized spline-based membership function plays an important factor in defining the overlapping ratio.

It should be mentioned that the above factors cited here work only for an
overlap ratio equal to 0.5, and for a well defined membership function, i.e., changing the membership function parameters $m$ and $\sigma$ in the studies this researcher conducted did not accurately yield the desired $O_r$ ratio that is different than 0.5. In fact this researcher found that, in the case of triangular membership function and with desired $O_r$ ratio of 0.5, the overlap ratio can be defined simply by the ratio of the parameters $m_1$ and $m_2$ only, i.e.,

$$O_r = \frac{m_1}{m_2}$$  \hfill (7.17)

with no need for the factors mentioned earlier. In this researcher’s experience on the topic of overlap, it is much easier to define the overlap ratio in the left and right (LR) parametric membership functions representation developed in Chapter IV than in any other membership functions form.

**Finding Membership Functions**

There are many different ways to find the membership functions for the inputs and outputs of a fuzzy control system. Some of these ways or methods were discussed in previous chapters. In this section, a list and a brief explanation for each method listed in previous chapters will be presented. The two approaches that used in the formulation of fuzzy control systems are the subjective approach and the objective approach. In the subjective approach, several methods for finding the membership function are usually used. The first method is based on the expert’s ability to generate
information through knowledge and understanding of the problem. The expert assigns the membership degree to each element of the universe of discourse. This method is sometimes referred to as an expert’s fuzzy system or intuition-based fuzzy system.

The second method in the subjective approach is similar to the previous one, except that in this method, instead of relying on only one expert’s knowledge and experience, many experts are involved, and their experience and knowledge is then processed all together via opinion poll or statistical analysis. This method is sometimes referred to as the rank-ordering method. A third subjective method is based on deduction. Deduction is based on available knowledge, such as natural laws, experts’ knowledge, and any other logically or theoretically based laws. It is usually performed by the entropy minimization principle, which clusters most optimally the parameters corresponding to the output classes. The input and output are then related by using the database to establish the relationship. This method is very useful and applicable for complex systems where the data are abundant and static. It is not much useful for data that are dynamic. The fourth subjective method is the induction method. Induction can be based on deriving membership degrees from a data set or facts that are known about the system.

In the objective approach, there are also many methods that can be used. Fuzzy statistics is based on statistical processing of the data available to derive the membership functions degrees. Knowledge from conventional control techniques and their application in the deriving of the membership function degrees based on an approximate analytical model or general information that can be derived by analyzing

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
the system is another method of an objective approach. This was the case applied to
the inverted pendulum case study presented in Chapter V, where both intuition and
knowledge learned from the classical control of the inverted pendulum were utilized
in deriving the membership functions and the rules. Other more advanced objective
methods can be used. Neural networks, for example, can be used to model the
membership functions or other parameters. This is often the case in what is called
neuro-fuzzy systems, as was discussed in Chapter V for the inverted pendulum
control problem. The parameters of the membership function can be selected and
modified using genetic algorithms. Genetic algorithms are based on the concept of
Darwin’s theory. Darwin’s theory basically stressed the fact that the existence of all
living things is based on the rule of “survival of the fittest.” All genetic algorithms
contain three basic operators: reproduction, crossover, and mutation. The initial
process start with a population of \( n \) strings for \( n \) parameters and of length \( L \) created at
random. Then the set of parameters is passed through a numerical model of the
problem space. The numerical model gives a solution based on the input set of
parameters. Based on the quality of this solution, a fitness value is then assigned to
the string representing the parameter. Each string is given a fitness value, and with
these fitness values, the three genetic operators are used to create a new generation of
strings, which is expected to perform better than the previous generations (higher
fitness values). The process continues until convergence is achieved within a
population. The reproduction operator determines which strings with better fitness
values receive corresponding better copies in the new generation. Crossover, on the
other hand, determines which strings are able to mix and match their desirable qualities in a random fashion. These two operators give the genetic algorithms most of their searching power. Mutation helps to increase the search power. For more information on genetic algorithms and neural network application in fuzzy logic, the reader is referred to Ross (1995), Passino et al. (1998), and Yen et al. (1999).

The fuzzy rules can be formed using (a) human opinions, interviews, observation, experience, and intuition; or (b) experimental results, measurement of the process inputs and outputs, and quantitative observation. Fuzzy statistics and data analysis tools can be used to generate the rules of the fuzzy control system. Fuzzy clustering, for example, can be used to derive the fuzzy rules, along with other parameters such as the membership functions.

The implementation of the fuzzy controller can be accomplished by using analog circuit processors, digital specialized processors, or digital general-purpose processors. For example, for a fuzzy controller with a small number of rules developed to be used in a high production rate such as consumer products, an analog circuit processor etc. can be used for implementation. For a complex fuzzy controller with a large number of rules developed for real-time application, a specialized digital processor is best suited for this kind of implementation.

**Testing and Evaluating the Fuzzy Logic Controller**

After the fuzzy logic controller is designed, some kind of verification must be conducted to evaluate its performance and assess if the control algorithm is the
optimal one that can be obtained. There are two ways to conduct verifications of any control system, fuzzy or conventional. The first one is verification by simulation. The second is experimental verification. The simulation should always be conducted first if possible. It is less expensive, timesaving, and flexible in the sense that during simulation, the controller's parameters can be modified or changed. The controller performance during simulation is compared to the design criteria or system's specifications. If the simulation results are as expected and the system specifications are met, then the next step is to conduct experimental verification. The simulation test is critical because it allows for testing of the extreme operating points of the system without causing the mechanical or physical damage that a real test, if failed, might cause. The extremes that are often tested during simulation are as follows: input values at the extreme ends of the universe of discourse, input values at the extreme ends of the individual membership function domains, and input values corresponding to membership function overlap. During simulation, attention must be given to the control surface of the fuzzy logic controller. The control surface usually gives a good indication of how the controller should perform. Conformity of the control surface to the design specifications is always required. The lack of conformity is usually due to one of the following factors:

1. The fuzzy operators, i.e., connective $\&$-norms or $\vee$-norms, and implication operators are chosen incorrectly.

2. The membership functions defined are not suitable or are incorrectly defined.
3. The rules specified in the rule-base have not been assigned correctly or there is more than one rule of the same antecedent part.

4. The universe of discourse for the inputs and outputs are incorrectly specified.

5. The number of membership functions specified for each input universe of discourse is either too small or too large.

If changes in the control surface, judged by the maximum rate of change of the inputs-output, are concentrated on the borders of all surface views, then the scaling factors need to be modified so that the rate of change is higher in the area of maximum surface changes.

**Tuning and Diagnosing the Fuzzy Logic Controller**

During the simulation process, if the controller does not perform up to expectations, then the controller needs to be diagnosed based on the simulation results. Through the experience gained during the work on this dissertation and from various read literatures, some guidelines that proven practical and applicable at any simulation stage have been developed. The purpose of developing these guidelines is to shorten the time spent analyzing the simulation results or experimental results and to provide a quick first-aid check on some parameters that can be modified or adjusted and, in some cases, can be changed without the hassle of going back and redesigning the controller from scratch. However, there is no guarantee that what is mentioned here will work on different design parameters or different simulation
platforms. The general idea, however, should hold for any fuzzy logic control system. In some instances the initial control law is defined in a wrong way or the rules are incorrectly formed. In such cases, these guidelines cannot change the facts and they may prove useless. Then, the designer should go back to the drawing board and design a new controller, taking into consideration the shortcoming of the faulty controller.

**System Response and Stability**

If the fuzzy logic controller does not provide stability to the system and the system response is judged to be unstable, then the following can be done to correct the problem:

1. Apply a different fuzzy system structure, such as the use of an adaptive or supervisory fuzzy logic controller instead of the more conventional fuzzy controller. The supervisory controller might degrade the system performance, but it provides the system with the needed stability.

2. Check the fuzzy system rule-base and inspect the rules for mislabeling, wrongly formulated rules, signs of the rules, redundancy of the rules, and conflict in the rules.

3. Tune and modify the scaling factors if the fuzzy logic controller is a scaled controller. A small change in the scaling factor can cause the controller output to saturate.
4. Check for the inference system operators, such as connective operators, implication and aggregation operators, and defuzzification methods. A combination of these operators might cause the system response to be unstable, as was shown earlier in this chapter.

5. If none of these steps change or modify the system response, then the whole system needs to be re-examined. The best method to use in the second design phase is to design a controller with as fewer rules as possible, but with not less than three rules; then simulation and modifications can be conducted at the same time. In this way, each new operator added, modified, or completely changed can be noticed more clearly and can be modified, corrected, or changed on the spot.

**Overshoot and Oscillatory Response**

If the response of the system has a large magnitude of overshoot with oscillatory response, then one can do the following:

1. Decrease the output denormalization factor, also called the de-scaling factor, in cases where the denormalization factor is used.

2. Use narrow membership functions around the response value expected, i.e., if the controller is a regulatory type controller then, the linguistic value around the zero value should be narrowed.

3. Change the membership function types. Some membership function types provide smooth response with less overshoot. An adaptive and generalized model of
expressing membership function as the normalized spline-base ones discussed in Chapter IV can also be used effectively to correct that.

4. Use a different $\ell$-norms operator. The Product operator instead of Max, for example, gives better stability and a smooth response.

5. In cases where a PD, PI, and PID type fuzzy logic controller are used, then utilize the procedure outlined in the earlier section to tune the fuzzy logic controller scaling factors.

Choosing different defuzzification methods can increase the speed of response of the system. MOM provides a faster response time but might yield oscillatory response with high overshoot; COA, with a wrong combination of fuzzy inference system parameters, might yield an unstable response. In a PID-type fuzzy logic controller, increasing the output-scaling factor of the PD-part and the deferential input-scaling factor can shorten the response speed.

In the case that the system steady state error is out of limit and unacceptable, then the following can be done to correct it:

1. Increase the concentration of the membership functions around the zero point by using either scaling factors or overlap factor.

2. In the case of a regulatory type fuzzy logic controller, reduce the width of the membership function for the zero class of the error signal.

3. Choose another defuzzification method, such as the center of area (COA) or the center of sum (COM), as well as the defuzzification methods discussed in this chapter. It is very helpful and time saving if one can develop algorithm that could be
executed and be able to pick up the best defuzzification method among all.

4. In scaled PID-type fuzzy logic controller, increase the scaling factor for an integral input. The scaling factor for the output PD-part should be decreased.

If the sensitivity of the fuzzy controller is insufficient, then the input scaling factors should be increased. In real-time implementation of the controller, some discrepancy can exist between the simulation results and experimental results. If the simulation and experimental results match closely or to any satisfactory degree, then the system design is complete and real-time application is considered. Sometimes the response speed when the controller is implemented is not satisfactory or performs slower than the simulation led to believe; then one might attempt changing the defuzzification methods and/or inference system. This might give a faster real-time implementation response. Another solution to this problem might be in the implementation processor type, i.e., an analog processor versus a specialized digital processor; or a high speed, more expensive digital processor versus a low speed, less expensive digital processor. If the controller can be modified and in doing so perform to the specification required, then modifying it and using less expensive digital or analog circuit processor in real-time implementation is a wise choice.

This chapter has dealt with the practical and design issues that a fuzzy logic control designer might face and deal with regularly. The results obtained from selecting certain fuzzy logic controller parameters highlight the effects of different parameters on the performance of the fuzzy logic controller. The effects of choosing certain $t$-norms and $t$-conorms operators, membership function shape, fuzzification,
defuzzification method, and composition and aggregation operators were shown. The choice of these parameters and methods plays an important factor in the controller performance and stability. It was shown that certain choices and combination of choices could lead to better performance or, in some cases, to an unstable control system. Based on the results obtained guidelines were developed, along with a diagnostic procedure to tune or change certain parameters to deal with the controller's unsatisfactory performance or shortcomings. The results also highlighted the need for the development and generalization of the fuzzy controller design processes. The results showed that, in certain situations and cases where the design of the fuzzy controller is conducted correctly, a wrong choice of any of the fuzzy system components could lead to confusion and a misleading conclusion. The analytical models of a fuzzy control law can be obtained only if certain assumptions regarding the membership function shape, the connective operator, the composition and aggregation operators, and the defuzzification method are assumed in advance. Nevertheless, when the development of the analytical model is not possible, then knowing the parameter's effect on the performance and stability of the controller is another way to ensure that the developed controller is stable and performs optimally. The study also showed that, if the model is developed carefully based on an adaptive scheme, then the best operators and parameters can automatically be selected. This is true because, in an adaptive scheme, one can incorporate different fuzzy inference operators and different defuzzification methods and produce the best combinations of factors and parameters that meet the designer specifications.
CHAPTER VIII

SUMMARY, CONCLUSIONS, AND SUGGESTIONS FOR FUTURE WORK

Summary

The work presented in this dissertation contains the theory behind fuzzy logic control in a general sense, and as it applies to control systems and design in particular. The present trends in the fuzzy logic research community are oriented toward the standardization of the fuzzy control design processes. In fuzzy control, however, most of the steps used to design and analyze conventional control systems can be utilized, provided that the control system is a simple one. In a highly nonlinear and complicated system, the design of the fuzzy control system requires a great deal of understanding and knowledge of the system, which sometimes is not available or is difficult to obtain. When not enough information is available, the plant can be experimentally simulated and the experimental data can be used to design the fuzzy control system.

The introduction and the background of fuzzy logic control was presented in Chapters II and III. Chapter II discussed the fuzzy control system in general and the subjects pertinent to the present study in particular. The chapter dealt with the mathematical basics and theoretical foundation of fuzzy logic as it pertains to control theory and applications. The variations of fuzzy system models and the differences
between the models were also discussed, in addition to the vocabulary and concepts of fuzzy logic.

Chapter III discussed various ways of designing and developing a fuzzy control system. In many situations, the only available information about the system is in the form of input/output data pairs. Different techniques and methods of using the experimental or statistical data to design a fuzzy control system were presented in the chapter. Knowledge gained from conventional control techniques, such as supervisory, adaptive, PD, PI, and PID control, was also discussed. The concept and the utilization of fuzzy clustering techniques, the normalization of the universe of discourse, and other important pertinent design issues were examined in-depth.

Chapters IV through VII constitute the core of this dissertation. These chapters were directed toward addressing different design, performance, stability, and applications issues of fuzzy logic control as they pertain to the control engineers in general and mechanical engineers in particular. Fuzzy logic control distinguishes itself from other control techniques in that it does not need a mathematical model of the controlled process. However, the lack of a mathematical model of the process makes simulating a fuzzy logic control system a formidable task. Simulation requires a mathematical model of the plant and, in most cases, either a mathematical model is not available, or it is a highly nonlinear and complex model. If the design of a fuzzy logic controller process is standardized, then the fuzzy control law can be expressed analytically, and similar analysis tools, such as those used for the conventional control methods, can be developed and used. The absence of systematic and
standardized methods of designing and analyzing fuzzy logic control systems limits and, in some cases, prevents most control engineers from applying fuzzy control in control applications. A great deal of information, data, and expert knowledge is required for a fuzzy-based control system to work properly, provided that the fuzzy inference engine operators and parameters are chosen and applied correctly.

Chapter IV presented the first part of this study in which two methods of formulating a fuzzy control law were developed. The purpose of analytically developing the fuzzy control law is that the fuzzy logic control system can be analyzed and the control design processes can be outlined. One advantage of developing an analytical fuzzy based control law is that the analysis can be easily done using any of the well-known stability and performance analysis methods. Also, the standardization of certain aspects of the fuzzy inference engine operators can lead to the generalization and standardization of the design of fuzzy control systems.

In Chapter IV, two approaches of developing fuzzy analytical control law were presented:

1. In the first approach, a fuzzy control law for a special class of nonlinear control problems was derived based on using a characteristic function exemplified by left and right parameterization and utilizing certain assumptions and definitions. The concept of the membership function parameterization was then used to derive the fuzzy control law for regulatory-type control problems. With the same concept of membership function parameterization, a new fuzzy control law applicable to tracking and regulatory-type control systems was developed. Both of the developed
models were then utilized in a practical application of great importance to verify their mathematical formulation and to show their utility in real-life applications. The two models, the regulatory and the tracking model, were used at different stages to design an idle speed fuzzy control system for the spark ignition engines. The results obtained using the developed idle speed fuzzy logic controller showed that the developed models yielded the same response trend as other obtained responses using conventional control methods. The utilization of the fuzzy idle speed controller showed also that fuzzy control can be used effectively to control a highly nonlinear and complex system, as is the case for the idle speed control system.

2. The second approach presented in Chapter IV was based on using a normalized spline-base membership function to design a fuzzy logic controller. The spline mathematical functions provide the method that can be used to express the membership functions based on theoretical foundation. It is worth noting that there are as many as 10 or more types and shapes of membership functions, as discussed in Chapters II and VII. Using the spline-based membership function can reduce or even eliminate the need to choose from among them. This is because the spline-base membership function can be expressed in such a way that it can approximate any shape or type depending on the number of knots and the degree of the spline function chosen. The other important aspect of using one method to express the membership function is that the important part of the fuzzy inference system, which is, in this case, the fuzzification process, can be integrated easily into any control algorithm. Doing so will also help greatly in developing an analytical control law. This allows
the system properties, such as stability, robustness, and optimality, to be investigated. Investigation of these properties at the design stage is very desirable and, in some cases, necessary before the system is implemented in real-life applications. Investigating these properties will lead to important benefits, such as enormous savings in time and money. The normalized spline-base membership functions developed in Chapter IV were used in Chapter V to design a fuzzy controller for the inverted pendulum problem. The simulation results obtained using this approach verified the control schemes based on the developed model and showed that the model and the fuzzy controller developed based on the model are functional and are applicable to highly complex and nonlinear practical applications. The performances of the developed model matched very closely and, in some cases, outperformed other control methods, such as the neuro-fuzzy based control method.

Chapter V presented various practical and real-life applications (i.e., engine management) of fuzzy logic control. The applications are presented in different areas of automotive control systems:

A. Two approaches to designing fuzzy logic controllers for an idle speed for spark ignition engines with different engine operating conditions were presented. The simulation and experiment results obtained showed that the developed controllers performed as expected. The results obtained using the developed fuzzy control schemes were similar (Chapter V, Figure 48, page 200) to those obtained using conventional control approaches or other forms of fuzzy logic based control schemes.
B. The second important automotive-related application is the cruise control problem. The cruise controller was developed using an adaptive fuzzy control scheme. This control scheme is based on a modified form of a previously developed model by Qiao and Mizumoto (1996). The results obtained using this modified form of the adaptive PID-type fuzzy logic controller were comparable to results obtained using linearized conventional PID and LQR controllers. It was also shown that under certain conditions the developed controller outperformed the conventional PID and LQR controllers.

C. The third practical application that was presented in Chapter V dealt with the air/fuel ratio control. The air/fuel ratio control problem is currently a widespread and active research topic. The importance of air/fuel ratio control increases daily as more and more stringent emission regulations and fuel-efficient regulations are imposed. The developed air/fuel ratio controller performed as well as other control approaches discussed in recently published studies.

D. The fourth practical application considered in Chapter V was the inverted pendulum control problem. The purpose of this application is to show that the normalized spline-base membership functions are suitable for use in their general format in designing a fuzzy logic controller to control a real-life application, such as the inverted pendulum control problem. This problem is a special case of a more generalized and complex control problem, which deals with rocket balance, trajectory control, and many other similar applications.

In its entirety, the studies contained in Chapter V have shown through the
development of a variety of important, complex, and highly nonlinear practical applications that fuzzy logic control principles work well for highly nonlinear control systems.

Chapter VI presented the experimental results of the two types of idle speed fuzzy logic controllers developed in Chapter V. The experiments were conducted to test the performance of the developed fuzzy control schemes for the idle speed control problem. The results obtained in Chapter VI showed the same response trend as obtained simulation results shown in Chapter V. The small discrepancies between the experiment results and the simulation results are attributed to normal factors that constrain the experiment-working environment, as well as the differences in the simulation and the experimental software templates.

In Chapter VII, the results obtained from selecting certain fuzzy logic controller parameters highlight the effects of different parameters on the performance of the fuzzy logic controller. The effects of choosing certain $t$-norms and $t$-conorms operators, membership function shape, fuzzification, defuzzification method, and composition and aggregation operators were shown. The choice of these parameters and methods plays important roles in the controller performance and stability. The results of this study are:

a. It was shown that certain choices and combination of choices could lead to better performance or, in some cases, to an unstable control system.

b. Based on the results obtained by studying the effects of parameter selections of the fuzzy logic controller and the common and gained knowledge of conventional
control methods, guidelines were developed, along with a diagnostic procedure to tune or change certain parameters to deal with the controller's unsatisfactory performance or shortcomings.

c. The results obtained in Chapter VII also highlighted the need for the development and generalization of the fuzzy controller design processes. The results showed that, in certain situations and cases where the design of the fuzzy controller is conducted correctly, a wrong choice of any of the fuzzy system components could lead to confusion and a misleading conclusion. The generalization of the design process of the fuzzy logic control system is important if fuzzy logic control is to be applied and used in wider ranges of applications. As was stressed in earlier sections, the analytical models of a fuzzy control law can be obtained only if certain assumptions regarding the membership function shape, the connective operator, the composition and aggregation operators, and the defuzzification method are assumed in advance. Nevertheless, when the development of the analytical model is not possible, then knowing the parameter's effect on the performance and stability of the controller is another way to ensure that the developed controller is stable and performs optimally. The study also showed that, if the model is developed carefully based on an adaptive scheme, then the best operators and parameters can automatically be selected. This is true because, in an adaptive scheme, one can incorporate different fuzzy inference operators and different defuzzification methods and produce the best combinations of factors and parameters that meet the designer specifications.
Discussion

Throughout this dissertation, the points that have been stressed are:

1. First, under certain restrictions and assumptions, the fuzzy control system can be expressed analytically with sound theoretical foundations and can be used effectively in real-life applications. This point leads to the possibility of establishing a standard design procedure in a similar fashion to conventional control methods. The similarities and differences between the conventional approach and fuzzy approach are highlighted throughout the dissertation so that the exchanged benefits between the two are utilized. The dissertation also addressed the importance of and the need for developing some kind of design guidelines that can be used for any fuzzy logic control system design template or software.

2. The second point stressed is that the fuzzy logic control system can be applied successfully to complex nonlinear systems as well as simple linear systems. The knowledge needed to design such control systems can be minimized, provided that the fuzzy controller design processes are standardized. Standardization of the fuzzy logic controller design processes saves time and makes it possible for the designer to obtain the needed information about the system. An important and somewhat logical question often asked is, Why use fuzzy logic if other more theoretically based and mathematically verified control techniques can do the job? The answer to this question, in cases where other control methods can be used effectively, differs from one person to another depending on the background and the preferences of the
designer or engineer. However, in situations where the conventional control methods do not work well or cannot be applied, then the answer to that question is very clear: fuzzy logic should be used. One should keep in mind, however, that the fuzzy logic controller, even in situations where a conventional controller works extremely well, has one important advantage. The distinct advantage of fuzzy control over other control methods is that a developed fuzzy controller is not limited or restricted in its applications, that is, a controller designed conventionally is surely based on certain simplified underlying assumptions, either in the plant mathematical modeling or in the derived conventional control law. Therefore, it will work well only if those assumptions are met. This, however, is not the case for the fuzzy logic controller, since it works well in all operation conditions and under any variety of circumstances, provided that the needed minor modifications are accommodated and the extreme operation conditions are taken into consideration in the initial design process.

3. The third point stressed is that a fuzzy logic control system has great potential for being the control system of choice in many practical applications and operating environments. The recent advancements in the development of theoretical foundations, practical applications, system response analysis techniques, software, and hardware make the fuzzy control system design, in terms of time, simulation, implementation, and experimental verification, a very attractive alternative to the conventional control approach.
The variety of important topics and research areas presented in this dissertation can be divided into three fuzzy logic control related research areas and two very important areas of application.

i. In the first research area the design and development of analytical fuzzy logic control law based on certain mathematical expressions and fuzzy inference operators assumption was shown possible and practical.

ii. In the second area of research the utilization of the fuzzy logic control in areas of application that have not benefited from the rapid advancement and development of fuzzy logic theory was shown through the simulation and experimental results to have a great potential and promising future.

iii. In the third research area the effects that the choices and selections of the fuzzy logic controllers' parameters have on the performance and stability of the fuzzy logic controller were shown to be important and can not be neglected.

The areas of applications were divided according to their investigative focus and aspects. The two applications areas are:

1. In the first area of application, fuzzy logic control systems for the automotive-related applications, specifically in the area of idle speed control, cruise control, and air/fuel ratio control, were shown to have comparable performance to those developed based on conventional control methods.

2. In the second area of application, the need for understanding the development, design, analysis, and diagnosis of the developed fuzzy control systems was clearly demonstrated.
Conclusions

The studies presented in this dissertation and the results obtained indicate the following conclusions:

1. In this study, the principles that were used to derive the regulatory-type controller were used to develop a tracking-type fuzzy control law by incorporating certain modifications and a simple addition of simplifying assumptions. The developed model, if converted to 2-input1-output fuzzy control rules, is equally applicable to either a MISO or a MIMO control system. The number of 2-inputs-1-output rules into which a MISO or a MIMO control rule can be decomposed is calculated using a simple mathematical formula (Equation (2.30), Chapter II, Page 44), which was developed for that purpose in this dissertation.

2. The generalized method shown in this research of expressing a normalized spline-based membership function in a control algorithm is an important result that could be utilized along with other generalized models of the fuzzy inference engine operators to simplify the design process and to help analyze the controller for performance and stability. The results obtained using the normalized spline-base membership functions showed that this goal can be achieved and the dispute and confusion regarding which membership function type should be used is eliminated.

3. Three different fuzzy logic controllers for three important practical applications in the area of engine management and control were developed. The first fuzzy logic controller was used for the idle speed control problem. The idle speed controller was
developed using two different approaches, resulting in two different fuzzy logic control algorithms. Each of the two developed controllers was simulated and experimentally verified. The results in both simulations and experiments showed that the performance of the developed controllers is comparable to the performance of the mathematically based control algorithms that often use conventional control methods in different formats. Comparison with recently published and more traditional control methods showed that the performance trend obtained using the developed controller is consistent with those methods. In the second application, a cruise control controller was designed using a parameter adaptive Proportional plus Integral plus Derivative (PID)-type fuzzy controller. The results obtained using the fuzzy based controller showed that the developed control scheme performed as well as the PID or Linear Quadratic Regulator (LQR) conventional controllers, which are now used in cars that are equipped with cruise control. The results also showed that the control efforts using the developed controller are minimized, which means that less fuel is spent and the service life of the fuel injector is extended. In the third application, an air/fuel ratio fuzzy based controller was developed. The results in both the development and the simulation stages showed that, with a minimal amount of data and a great deal of intuitive knowledge of the air/fuel ratio control problem, a fuzzy based controller can be developed. This result is important because, when trying to develop a mathematical model for the conventional control algorithm, many assumptions and simplifications are made, which sometimes take the form of a model linearization near an operating region, and therefore, dictate a narrow operating range. The results
and the response trend obtained using the developed controller showed also that the
hassle of obtaining a mathematical model for a highly intricate and complex system
is not necessary.

4. The stability and performance of a fuzzy logic controller were investigated by
utilizing a parametric study, in which the selection of the fuzzy controller’s
parameters and their influence on its performance and stability were observed and
analyzed. The results obtained in this study indicated that certain parameters or
combinations of parameters have a profound effect on the stability and performance
of the fuzzy controller. Different combinations of parameters were studied, and it was
observed that, although the controller itself was correctly designed, the performance
of the controller was dependent on the parameters chosen for the fuzzy inference
system. This is an important result, which shows that the unsatisfactory performance
and the instability of the fuzzy controller is not due solely to the rule-base and the
formulation of the fuzzy control algorithms, but it can be due to wrong choice or
combinations of wrong choices of the fuzzy inference parameters. Knowing this can
help greatly in reducing the time of the redesign process and the simulation studies.
The results of this study, in addition to general knowledge about conventional control
techniques, were then used to develop a diagnostic outline in which a designer, aided
by the simulation results of the developed controller, can diagnose the shortcomings
and unsatisfactory performance of the controller and help tune or suggest changes or
choices of different parameters.
5. Altogether, the design, development, application, and analysis of a fuzzy based control system were addressed, namely, tracking-type fuzzy control law, use of normalized spline-based membership functions, real-life application of fuzzy logic control, and parametric study in this research.

Future Studies and Projects

The work presented in this dissertation lays the groundwork for in-depth analytical studies and the development of a more generalized analytically based fuzzy control law. Future work could continue on the same path as the one presented in this dissertation. The future studies could be concentrated on the investigation of the spline-base membership functions. The choice of the membership functions for the inputs and outputs of the fuzzy controller should be investigated based on the generalization characteristic of the normalized spline-base membership function. The future goal in this regard should be directed toward obtaining an adaptive and selective fuzzification algorithm, so that the choice of the membership function shape and overlapping ratio is selected automatically to fit the system design criteria. Also, the control law that uses the $LR$ parameterization for expressing the membership functions should be modified from the form presented in this work. The modified form should include other choices of defuzzification methods, especially an adaptive type defuzzification method or a parametric defuzzification method such as the SLIDE method or other defuzzification methods discussed in Chapters II and VII. Attention should be given to the decomposition algorithm, which, if developed, will
help greatly in the development of structurally adjustable controller by extending the number of its inputs and outputs.

The software development for designing and simulating a fuzzy controller should also be the focus of any future studies as well.

Another future study could concentrate on the fuzzy controller performance and stability issues. The stability of the fuzzy controller can best be investigated and assessed if the control algorithm is expressed in analytical form. A stability criterion that can be applied to the fuzzy control algorithm should also be the focus of any future study. This could be achieved if in-depth study of the controller performance, relative to all other design factors, is conducted. Also, a more generalized diagnostic procedure based on different structures of the controller algorithm should be developed. The diagnostic procedure should serve as a quick reference list to aid the designer in the design, simulation, and performance evaluation processes.

The Development of a generic fuzzy controller design module for engine control unit (ECU) should be another future research to yield a control system that is:

(a) Flexible in its applications to different kinds of automotive models.

(b) More economical in terms of the time spent on the design, simulation, and implementation.

(c) Robust in terms of its ability to accommodate a wide range of operation conditions and external disturbances.

(d) Optimal in terms of its durability.
Appendix A

Experimental Implementations: Introduction, Assembly Language Programs, and Experimental Results Using Engine Simulator at Oakland University Testing Facilities
FUZZY LOGIC CONTROLLER EXPERIMENTAL VERIFICATION USING ASSEMBLY LANGUAGE PROGRAMMING

Introduction

The fuzzy controller consists of a set of fuzzy rules of the form if $X$ AND $Y$, then $Z$, where $X$, $Y$ and $Z$ are fuzzy sets. For example, a fuzzy rule for an air conditioner might be if temperature is WARM and change in temperature is ZERO, then motor speed is FAST. Note that WARM, ZERO, and FAST are fuzzy sets. After applying all of the fuzzy rules to a given set of input variables, the output (motor speed in this case) will, in general, belong to more than one fuzzy set with different weights. The weighted output fuzzy sets are added and then a centroid defuzzification process is used to obtain a single crisp output value.

![Functional diagram of a fuzzy controller](image)

Figure A. Functional diagram of a fuzzy controller.
Implementing a Fuzzy Controller

The fuzzy controller shown in Figure A consists of three parts: the fuzzification of inputs, the processing of rules, and the defuzzification of the output. Each of these parts is considered separately.

Fuzzification of inputs

For each crisp input $x_i$, a set of weights $w'_j$ are computed for each membership function (Haskell). In general, each input will have a different set and number of membership functions. For input number $i$ the weight $w'_j$ can be stored in a vector $\text{wi}[j], j = 0, NM_i - 1$ where $NM_i$ is the number of membership functions for input $i$.

Each value in the weight vector is a weight value between 0 and 1 given by the shape of a particular membership function. Typically for a given input, the weight vector will contain up to two non-zero entries in adjacent cells. For example, a trapezoidal membership function can be characterized by the four parameters $B_1$, $B_2$, $B_3$, and $B_4$ as shown in Figure B. The user must define the membership functions associated with each controller input. When the input is measured, the program will compute the weight vector associated with all of its membership functions.

Figure B. Characterizing a single membership function.
Processing the rules

For the idle speed controller, the two input fuzzy variables are $\theta$, the difference between the current RPM, the desired set RPM, and $\Delta \theta$. Here the symbol $\Delta \theta$ is used to indicate the approximation of the instantaneous rate of change of $\theta$.

There are two separate output control variables: the throttle control and the spark advance control. The Throttle is a signal, \{-2, -1, 0, 1, 2\}, sent to a stepper motor to control the throttle angle. The spark advance is a signal, \{-4, -2, 0, 2, 4\}, which represents the number of half-degrees by which the spark is to be advanced from dead center. Two separate fuzzy controllers for the throttle output and for the spark advance output are developed in later sections. In this section a typical output fuzzy variable is denoted by the symbol $v$.

Assume that the five membership functions shown in Figure C apply to both inputs as well as the output. The rules for a typical output signal can be represented by the fuzzy associate memory (FAM) table as shown in Table A that is analogous to a binary K-map used in digital logic.

![Figure C. Example of fuzzy membership functions.](image)

The columns in Table A correspond to the fuzzy sets for the input $\theta$ and the rows correspond to the fuzzy sets for the input $\Delta \theta$. The entries in the fuzzy FAM table
are the fuzzy sets for the output signal. Note that there are 25 rules shown in Table A.

For example, two of the rules that can be designed to keep the RPM equal to the set RPM can be expressed as:

\[
\text{if } \theta \text{ is } PM \text{ and } \Delta \theta \text{ is } Z \text{ then the output is } NM \\
\text{if } \theta \text{ is } PS \text{ and } \Delta \theta \text{ is } Z \text{ then the output is } NS
\]

The method of applying the fuzzy rules is illustrated in Figure D. The phrase \( \theta \) is \( PM \) means find the weight value from the \( PM \) membership function for that particular value of \( \theta \). The phrase \( \Delta \theta \) is \( Z \) means find the weight value from the \( Z \) membership function for that particular value of \( \Delta \theta \). The fuzzy connective \textit{and} means take the \textit{minimum} value of these two weight values (when minimum connective operator is used). The consequent phrase \textit{then the output is} \( NM \) means reduce the output membership function \( NM \) by multiplying the original membership function \( NM \) by the minimum weight value found from the antecedent phrase \textit{if} \( \theta \) is \( PM \) and \( \Delta \theta \) is \( Z \) as shown in Figure 5. Similarly, the rule \textit{if} \( \theta \) is \( PS \) and \( \Delta \theta \) is \( Z \) \textit{then the output is} \( NS \) will reduce the output membership function \( NS \).

![Figure D. Applying the fuzzy rules.](image)
Table A. Fuzzy Associate Memory (FAM) Table.

<table>
<thead>
<tr>
<th>$\Delta \theta$</th>
<th>$\theta$</th>
<th>NM</th>
<th>NS</th>
<th>Z</th>
<th>PS</th>
<th>PM</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\downarrow$</td>
<td>NM</td>
<td>PM</td>
<td>PM</td>
<td>PM</td>
<td>PS</td>
<td>Z</td>
</tr>
<tr>
<td>NS</td>
<td>PM</td>
<td>PM</td>
<td>PS</td>
<td>Z</td>
<td>NS</td>
<td>NS</td>
</tr>
<tr>
<td>Z</td>
<td>PM</td>
<td>PS</td>
<td>Z</td>
<td>NS</td>
<td>NM</td>
<td>NM</td>
</tr>
<tr>
<td>PS</td>
<td>PS</td>
<td>Z</td>
<td>NS</td>
<td>NM</td>
<td>NM</td>
<td>NM</td>
</tr>
<tr>
<td>PM</td>
<td>Z</td>
<td>NS</td>
<td>NM</td>
<td>NM</td>
<td>NM</td>
<td>NM</td>
</tr>
</tbody>
</table>

Output defuzzification

For a given pair of input values $\theta$ and $\Delta \theta$ the minimum weight for many of the rules will be zero meaning that that particular rule will not contribute to the output value. After applying all of the fuzzy rules using the method shown in Figure C, the crisp output value is taken to be the centroid value of the sum of all the weight-reduced output membership functions as suggested in Figure D for the rules:

- if $\theta$ is PM and $\Delta \theta$ is Z then the output is NM
- if $\theta$ is PS and $\Delta \theta$ is Z then the output is NS

If $L_i(y)$ denotes the original output membership function associated with rule $i$ and $y$ denotes the output universe of discourse (e.g., throttle signal in this case). After applying rule $i$ this membership function will be reduced to the value

$$m_i(y) = w_i L_i(y) \quad (a)$$

where $w_i$ is the minimum weight found by applying rule $i$. The sum of these reduced output membership functions over all rules is then given by:
where $N$ is the number of rules.

The crisp output value $v$ is then given by the centroid of $M(y)$ from Equation (c):

\[
v = \frac{\int yM(y)dy}{\int M(y)dy}
\]

Note that the centroid of membership function $L_i(y)$ is given by:

\[
c_i = \frac{\int yL_i(y)dy}{\int L_i(y)dy}
\]

The area of the membership function $L_i(y)$ can be expressed as:

\[
A_i = \int L_i(y)dy
\]

Substituting (d) into (c) yields

\[
\int yL_i(y)dy = c_i A_i
\]

Using Equations (a) and (b) the numerator of (c) can be written as:

\[
\int yM(y)dy = \int y\sum_{i=1}^{N} w_i L_i(y)dy = \sum_{i=1}^{N} \int yw_i L_i(y)dy = \sum_{i=1}^{N} w_i c_i A_i
\]

Similarly, using Equations (a) and (b) the denominator of (c) can be written as:
Substituting Equations (g) and (h) into (c) then the crisp output of the fuzzy controller can be expressed as:

\[ v = \frac{\sum_{i=1}^{N} w_i c_i A_i}{\sum_{i=1}^{N} w_i A_i} \]  \hspace{1cm} (i)

Equation (i) computes the output centroid from the centroid \((c_i)\) of the individual output membership functions. Note that in Equation (i) the summation is over all \(N\) rules. But the number of output membership functions, \(Q\), will, in general, be less than the number of rules, \(N\). This means that in the sums in Equation (i) there will be many terms that will have the same values of \(c_i\) and \(A_i\). For example, suppose that rules 2, 3, and 4 in the sum all have the output membership function \(L^k\) as the consequent. This means that in the sum \(w_2 c_2 A_2 + w_3 c_3 A_3 + w_4 c_4 A_4\) the values \(c_i\) and \(A_i\) are the same values \(c^k\) and \(A^k\) because they are just the centroid and area of the \(k^{th}\) output membership function. These three terms would then contribute the value \((w_2 + w_3 + w_4)c^k A^k = W^k c^k A^k\) to the sum, where \((w_2 + w_3 + w_4) = W^k\) is the sum of all weights from rules whose consequent is output membership function \(v\). This means that the equation for the output value, \(v\), given by (i) can be rewritten as:

\[ v = \frac{\sum_{k=1}^{Q} W^k c^k A^k}{\sum_{k=1}^{Q} W^k A^k} \]  \hspace{1cm} (j)
If the areas \( A^k \) of all output membership functions are equal, then Equation (j) reduces to Equation (k):

\[
v = \frac{\sum_{k=1}^{Q} W^k c^k}{\sum_{k=1}^{Q} W^k}
\]  
(k)

Equations (j) and (k) show that the output crisp value of a fuzzy controller can be computed by summing over only the number of output membership functions rather than over all fuzzy rules.

**DSP Assembly Language Implementation**

In this section the implementation of the fuzzy controller described in the early sections of this report for the idle speed controller using DSP assembly language is given. The DSP/HC11 program implements the complete fuzzy controller. The first part of the program defines the input membership functions for fuzzy rules for controlling the spark advance and the throttle position. The arrays \( OUT1 \) (\( OUT2 \)) and \( CENT1 \) (\( CENT2 \)) correspond to the arrays \( W^k \) and \( A^k \) in Equation (k). The constant values \( NM_1, NS_1, Z_1, PS_1, \) and \( PM_1 \) are the membership function names for the fuzzy sets of input \( x1 \). Similar names are defined for input \( x2 \). These will be the indexes into the weight arrays \( X1_{\text{WT}}, X2_{\text{WT}}, X3_{\text{WT}} \) and \( X4_{\text{WT}} \).

The parameters associated with the input membership functions are defined in first in the program. The constants \( B1, B2, B3, \) and \( B4 \) are indexes into the membership function data structures and define the membership function. These
parameters correspond to $B_1$, $B_2$, $B_3$, and $B_4$ shown in Figure C. The membership function parameters for input $x_1$ start at address $X1_MF$ and those for inputs $x_2$, $x_3$ and $x_4$ start at addresses $X2_MF$, $X3_MF$ and $X4_MF$, respectively. Two lookup tables at addresses $LOOKUP1$ and $LOOKUP2$ are used to compute the weight value for the linear ramp values between $B_1$ and $B_2$ and between $B_3$ and $B_4$. These weight values are computed in the specially written subroutine called $GETWT$.

The rules for the spark advance begin at address $RULES1$ and the rules for the throttle position begin at address $RULES2$. These rules are of the form:

\[
FDB \; X1_{\text{WT}}+NM_1 \quad ;\; \text{if } X1 \text{ is } NM_1 \\
FDB \; X3_{\text{WT}}+NM_2 \quad ;\; \text{and } X3 \text{ is } NM_2 \\
FDB \; \text{OUT1+PM}_0 \quad ;\; \text{Then OUT1 is PM}_0
\]

Note that the first two of these three double bytes contain the addresses of the weight vector elements for the two inputs corresponding to this particular rule. The last double byte contains the address of the $OUT1$ array element to which the minimum of the two input weight array values is to be added. This process is carried out by the subroutines $FIRERULES1$ and $FIRERULES2$.

The subroutine $GETINPT$ will fill the weight arrays for all inputs by calling the subroutine $FILLWT$ for each input. The subroutines $GETX1$ and $GETX2$ called by the subroutine $GETINPT$ compute the RPM error and change in RPM error as the two inputs.

The subroutines $FINDOUT1$ and $FINDOUT2$ compute the crisp output values using Equation (k). The complete fuzzy controller will continually call subroutines $GETINPT$, $FIRERULES1$, $FINDOUT1$, $FIRERULES2$, and $FINDOUT2$. 
EXPERIMENTAL RESULTS

The membership functions shape for the experimental verification after a considerable trial and error are then specified for the inputs and the output. The four parameters $B_1, B_2, B_3,$ and $B_4$ values for the membership function as discussed earlier and shown in Figure A are obtained and are shown in Tables 1 through 4.

Table 1. RPM Error

<table>
<thead>
<tr>
<th>B1</th>
<th>B2</th>
<th>B3</th>
<th>B4</th>
</tr>
</thead>
<tbody>
<tr>
<td>-5520</td>
<td>-5500</td>
<td>-600</td>
<td>-580</td>
</tr>
<tr>
<td>-590</td>
<td>-490</td>
<td>-290</td>
<td>-190</td>
</tr>
<tr>
<td>-280</td>
<td>-260</td>
<td>260</td>
<td>280</td>
</tr>
<tr>
<td>190</td>
<td>290</td>
<td>490</td>
<td>590</td>
</tr>
<tr>
<td>580</td>
<td>600</td>
<td>16000</td>
<td>16020</td>
</tr>
</tbody>
</table>

Table 2. Rate of Change of RPM Error (Throttle)

<table>
<thead>
<tr>
<th>B1</th>
<th>B2</th>
<th>B3</th>
<th>B4</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1970</td>
<td>-1950</td>
<td>-1280</td>
<td>-1260</td>
</tr>
<tr>
<td>-1270</td>
<td>-1250</td>
<td>-830</td>
<td>-810</td>
</tr>
<tr>
<td>-820</td>
<td>-800</td>
<td>800</td>
<td>820</td>
</tr>
<tr>
<td>810</td>
<td>830</td>
<td>1250</td>
<td>1270</td>
</tr>
<tr>
<td>1260</td>
<td>1280</td>
<td>1950</td>
<td>1970</td>
</tr>
</tbody>
</table>
Table 3. RPM Error (Spark Advance)

<table>
<thead>
<tr>
<th>B1</th>
<th>B2</th>
<th>B3</th>
<th>B4</th>
</tr>
</thead>
<tbody>
<tr>
<td>-5020</td>
<td>-5000</td>
<td>-335</td>
<td>-315</td>
</tr>
<tr>
<td>-320</td>
<td>-300</td>
<td>-190</td>
<td>-170</td>
</tr>
<tr>
<td>-180</td>
<td>-160</td>
<td>160</td>
<td>180</td>
</tr>
<tr>
<td>170</td>
<td>190</td>
<td>300</td>
<td>320</td>
</tr>
<tr>
<td>315</td>
<td>335</td>
<td>16000</td>
<td>16020</td>
</tr>
</tbody>
</table>

Table 4. Rate of Change of RPM Error (Spark Advance)

<table>
<thead>
<tr>
<th>B1</th>
<th>B2</th>
<th>B3</th>
<th>B4</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1970</td>
<td>-1950</td>
<td>-1080</td>
<td>-1060</td>
</tr>
<tr>
<td>-1070</td>
<td>-1050</td>
<td>-230</td>
<td>-210</td>
</tr>
<tr>
<td>-220</td>
<td>-200</td>
<td>200</td>
<td>220</td>
</tr>
<tr>
<td>210</td>
<td>230</td>
<td>1050</td>
<td>1070</td>
</tr>
<tr>
<td>1060</td>
<td>1080</td>
<td>1950</td>
<td>1970</td>
</tr>
</tbody>
</table>

The rules used for the Throttle and spark advance are shown in Tables 5 and 6.

Table 5. Fuzzy Rules for Throttle Angle (θ)

<table>
<thead>
<tr>
<th>θ</th>
<th>NM</th>
<th>NS</th>
<th>Z</th>
<th>PS</th>
<th>PM</th>
</tr>
</thead>
<tbody>
<tr>
<td>NM</td>
<td>PM</td>
<td>PS</td>
<td>Z</td>
<td>Z</td>
<td>Z</td>
</tr>
<tr>
<td>NS</td>
<td>PS</td>
<td>PS</td>
<td>Z</td>
<td>Z</td>
<td>Z</td>
</tr>
<tr>
<td>Δθ</td>
<td>Z</td>
<td>PS</td>
<td>Z</td>
<td>Z</td>
<td>Z</td>
</tr>
<tr>
<td>PS</td>
<td>Z</td>
<td>Z</td>
<td>Z</td>
<td>NS</td>
<td>NS</td>
</tr>
<tr>
<td>PM</td>
<td>Z</td>
<td>Z</td>
<td>Z</td>
<td>NS</td>
<td>NM</td>
</tr>
</tbody>
</table>

Note. θ = RPM error and Δθ = Rate of change of RPM error.
Table 6. Fuzzy Rules For Spark Advance (α)

<table>
<thead>
<tr>
<th></th>
<th>NM</th>
<th>NS</th>
<th>Z</th>
<th>PS</th>
<th>PM</th>
</tr>
</thead>
<tbody>
<tr>
<td>NM</td>
<td>PM</td>
<td>PS</td>
<td>PS</td>
<td>Z</td>
<td>Z</td>
</tr>
<tr>
<td>NS</td>
<td>PS</td>
<td>PS</td>
<td>Z</td>
<td>Z</td>
<td>Z</td>
</tr>
<tr>
<td>Δα</td>
<td>Z</td>
<td>PS</td>
<td>Z</td>
<td>Z</td>
<td>NS</td>
</tr>
<tr>
<td>PS</td>
<td>Z</td>
<td>Z</td>
<td>Z</td>
<td>NS</td>
<td>NS</td>
</tr>
<tr>
<td>PM</td>
<td>Z</td>
<td>Z</td>
<td>NS</td>
<td>NS</td>
<td>NM</td>
</tr>
</tbody>
</table>

Note. α = RPM error and Δα = Rate of change of RPM error

A number of experiments were conducted using this fuzzy controller. The variation of RPM during a cold start is shown in Figure E. The increase in RPM at about 16 seconds is when the lights were turned on.

Figure E. RPM Variation for Cold Start as a Function of Time.
Conclusions and Recommendations

The fuzzy controller described and implemented in Chapter VI provided good idle speed controller for spark ignition engines. As described in the Chapters II and III fuzzy controllers have the advantage of being model free and can approximate any real continuous function to arbitrary accuracy. However, it proved to be a time-consuming task to iteratively tune the membership functions and rules of the fuzzy controller for optimum performance. This was particularly true inasmuch as any change required the burning of a new PROM to insert in the system. This problem would be overcome by implementing the following two changes:

1. Provide a mechanism for the real-time modification of membership functions while the fuzzy controller is actually controlling the engine. In this way the membership functions can be tuned while observing the operation of the engine. This can be accomplished by implementing a kernel of the type described in a recent papers in which a C++ program running on a PC can interactively and incrementally download executable code to an embedded system and communicate to the system while the system is running.

2. Develop an algorithm that can automatically search for the optimized membership function parameters while the engine is running. In this way the tuning of the membership functions could be carried out automatically, without the need for human intervention. This could be done by using a learning algorithm.

These two tasks will be undertaken in future research on fuzzy control.
Appendix B

Experimental Setup for Engine Control Using Chryslers' Engine Simulator at Oakland University Testing Facilities
Chrysler Engine Simulator Used for Experimental Verification At Oakland University Testing Facilities. Engine Control Testing Stand Connecting to the Engine Control Load Box and the Computer (a), Expanded View of the Control Testing Stand (b), and Engine Control Load Box Expanded View with PC Connection (c).
Appendix C

Spline-Based Membership Functions Degrees and Resulting Shapes
The Resulting Membership Function Shapes and Types as A Function of the Spline Degree and The Ability of Normalized Spline Base To Approximate other Shapes.
BIBLIOGRAPHY


Mencik Z., and Blumberg, P. N. Representation of Engine Data by Multi-Variate Least Squares Regression, *SAE Paper 780288*.


Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.


Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.


