Intelligent Control of Induction Motors

by

Fuzi Lftisi, B.Sc, M.Sc

A thesis submitted to the School of Graduate Studies
in partial fulfilment of the requirements for the degree of
Doctor of Philosophy

Faculty of Engineering & Applied Science
Memorial University of Newfoundland

[May 2019]
St. John's, Newfoundland and Labrador

Abstract

This thesis presents the development and implementation of an integral field oriented intelligent control for an induction motor (IM) drive using Fuzzy Logic Controller (FLC), and an Artificial Neural Network (ANN), employing a finite element controller and making use of a Proportional Integral (PI) adaptive controller as well. An analytical model of an induction motor drive has been developed. In order to prove the superiority of the proposed controller, the performance of this controller is compared with conventional PI-based IM drives. The performance of the proposed IM drive is investigated extensively at different operating conditions in simulation. The proposed adaptive PI-based speed controller's performance is found to be robust and it is a potential candidate for high performance industrial drive applications. The novel work focuses on using a Finite Element Controller map (FECM) to manipulate adaptive controllers for motor control drives.

A digital signal processing (DSP) board DS1104 and laboratory induction motor were used to implement the complete vector control scheme. The test results have been compared with simulated results at different dynamic operating conditions. The effectiveness of this control scheme has been evaluated, and it has been found to be more efficient than the conventional PI controller.

Acknowledgements

First, I would like to express my sincere gratitude to my supervisor, Dr. M. Azizur Rahman, for all his guidance, advice, support and encouragement throughout the PhD program. I am also grateful to him and to the Faculty of Engineering and Applied Science at Memorial University of Newfoundland for providing me with the necessary laboratory space, equipment, and technical assistance to complete my PhD research smoothly.

I would also like to thank the Ministry of Higher Education and Scientific Research of Libya for the financial support provided for me to pursue this PhD program.

Particular gratitude goes to Dr. Michael Hinchey for his support and for his willingness to devote so much of his time to providing his students with guidance.

I wish to thank MUN's School of Graduate Studies for providing me with a teaching assistantship in the course of carrying out my doctoral studies at the Faculty of Engineering and Applied Science.

I am also indebted to Dr. Glyn George, Xiaodong Liang and Dr. Siu O'Young, who are members of my supervisory committee. They have made invaluable suggestions about my work over the past years.

It is very important that I acknowledge the assistance I received from the Faculty, staff members, technicians and graduate students of the Faculty of Engineering and Applied Science of Memorial University of Newfoundland.

Thank you, as well, to Ginny Ryan and her team at MUN's Writing Centre for their assistance throughout the writing of my thesis.

Most importantly, I express my sincere appreciation to my wife, Nwara, my parents - Mr. Mohamed Lftisi and Mrs. Mabroka Kalil - my brothers and sisters as well as other relatives and friends, for their encouragement and support throughout the program.

Table of Contents

Abstract ..... **ii**

Acknowledgements ..... **iii**

Table of Contents ..... **v**

List of Tables ..... **xxi**

List of Symbols ..... **xxii**

List of Abbreviations ..... **xxx**

Chapter 1. Control of Motor Drives **1**

1.1 Introduction **1**

1.2 Literature Review **2**

1.3 Vector control based Drive **3**

1.4 PI, PID and Adaptive Controllers **4**

1.5 Adaptive and Intelligent Control of Induction Motor Drive **6**

1.6 Problem Identification **17**

1.7 Thesis Objectives **20**

1.8 Scope of Thesis Work **20**

1.9 Organisation of the Thesis **21**

Chapter 2. Introduction to Induction Machines **23**

2.1 Principle of Operation **23**

2.2 Classification of the Induction Machine **24**

2.3 Types of Induction Motors **24**

2.4 Dynamic Modelling of the Induction Motor **25**

2.5 Transformation of Induction Machine Quantities 30

2.5.1 Clarke Transformation 32

2.5.2 Inverse Clarke Transformation 33

2.6 Hysteresis Current Controller 35

2.7 Simulation of Induction Motor 39

2.8 Discussion of Simulation Results 43

2.9 Summary 44

Chapter 3. Speed Drives Control Methods for Induction Motors 45

3.1 Basic IM Drive Concepts 45

3.2 Stator Voltage Control Operation 46

3.3 Scalar Control Methods 48

3.4 Vector Control Schemes 49

3.5 Derivations of the Field Oriented Control (FOC): 51

3.6 Indirect FOC IM Drive 56

3.7 Optimal PI Parameters 58

3.8 Design and Analysis of Proportional Integral Controller 63

Chapter 4. Adaptive PI Controller with Gain Scheduling for Indirect Field Oriented Control of Induction Motor 68

4.1 Introduction 69

4.2 Gain Scheduled Controller Structure 73

4.3 Simulation Results and Discussion 77

4.3.1 Trajectory tracking performance 78

4.3.2 Effect of Load Disturbances 78

4.4 Conclusion 80

4.5 Grid Search Method in PI Controllers 81

4.5.1 Introduction 81

4.6 Self-Tuning Regulator Based Speed Controller for IM Drive Systems 83

4.7 Grid Search Method for an Self-Tuning PI Controller 84

4.8 Simulation Results 88

4.9 Summary 92

Chapter 5. A Fast Fuzzy Logic Controller Based on Vector Control for Three Phase Induction Motor Drives 93

5.1 Introduction 93

5.2 Fuzzy Logic Controller Structure 96

5.3 Control Strategy 97

5.4 Rule Evaluation 101

5.5 Aggregation and Defuzzification 103

5.6 Simulation Results 104

5.7 Experimental Results and Discussion 107

5.8 Conclusion 109

5.9 Extended Applications for the IM Motor Drive 110

5.10 Simulation Results 118

5.11 Experimental Results and Discussion 120

5.12 Fuzzy Sets Breakpoints Effect 122

5.13 Chapter Summary 131

Chapter 6. Vector Control of Induction Motor Using Neural Networks 132

6.1 Introduction 132

6.2 Artificial Neural Network Model 135

6.3 Artificial Neural Network Based Speed Controller for IM Drive 142

6.4 Vector Control of the IM with an Artificial Neural Network Controller 143

6.4.1 Artificial Neural Network Architectures 144

6.4.2 Simulation Results and Discussion 149

6.5 Experimental Results and Discussion 152

6.6 Conclusion 154

6.7 Training Processes and Properties of Learning 154

6.8 Training Algorithms for Adaptive Artificial Neural Network 155

6.8.1 Simulation Results Using Grid Search Method 161

6.9 Chapter Summary 166

Chapter 7. Finite Element Controller Map Method (FECMM) 167

7.1 Introduction 167

7.2 Nodes 168

7.3 Coordinate Systems 168

7.3.1 Global coordinates 169

7.3.2 Local coordinates 169

7.3.3 Natural coordinates 169

7.4 Shape Functions 170

7.5 Isoperimetric Element 171

7.6 Lagrange Interpolation Functions: Linear Rectangular Element 172

7.7 Finite Element Controller Map Method 175

7.8 Finite Element Maps 178

7.8.1 Nine Element Controller Map 179

7.9 Simulation Results and Discussion 181

7.9.1 Nine Element Results 181

7.10 Five Element Controller Map Method 185

7.10.1 Five Element with Four Nodes for each Element 185

7.10.2 Five Discrete Elements with Eight Nodes 189

7.11 Grid Search Optimization 194

7.12 Five Element Results with four nodes for each element 198

7.13 Five Element Results with Eight Nodes 201

7.14 Experimental Results and Discussion 204

7.15 Experimental Drive Configuration 206

7.16 Experimental Results and Discussion 208

7.17 Conclusion 213

7.18 Chapter Summary 214

Chapter 8. Comparative Simulation Results between Conventional PI, Fuzzy logic, Neural Network and Finite Element Controller Map Based IM Drive 215

Chapter 9. Conclusions, Contributions and Future Work 226

9.1 Conclusions 226

9.2 Contributions 230

9.3 Future Work 231

References 232

Appendix A Parameters of Induction Motors 254

Appendix B List of Technical Papers 255

Appendix C 257

C.1 Command Current 257

C.2 Current Controller 259

C.3 Voltage Source Inverter 259

C.4 Transformation to Stationary Frame 261

Appendix D 262

D.1 Fuzzy logic controller for four rules 262

D.2 Fuzzy logic controller with nine rules used instead of PI controller 264

Appendix E Neural Network Controller..... 266

Appendix F 267

F.1 Finite Element Controller Map for Nine Elements with Four Nodes..... 267

F.2 Finite Element Controller Map for five Elements with Four Nodes 268

F.3 Finite Element Controller for Five Elements with Eight Nodes..... 271

F.4 Finite Element Controller Map for Five Elements with Eight Nodes when the middle element takes a parabolic shape..... 275

List of Figures

Figure 2-1: Classification of Induction Machine 24

Figure 2-2: The dynamic equivalent circuit (d-q) of an induction machine, (a) direct axis circuit, (b) quadrature axis circuit 27

Figure 2-3: abc-dqo axis transformation 30

Figure 2-4: Clarke Transformation 32

Figure 2-5: Park Transformation 33

Figure 2-6: Fixed-band hysteresis current controller schemes 36

Figure 2-7: Hysteresis current controller waveforms 37

Figure 2-8: Simulated speed of the motor when it is powered by a voltage inverter 41

Figure 2-9: Simulated torque of the motor when it is powered by a voltage inverter 41

Figure 2-10: Simulated speed of the motor when it is directly powered by a voltage source supply41

Figure 2-11: Simulated torque of the motor when it is directly powered by a voltage source supply 41

Figure 2-12: Simulated one phase current when the motor is powered by a voltage inverter 42

Figure 2-13: Simulated flux of the motor when it is powered by a voltage inverter 42

Figure 2-14: Simulated phase current when the motor is directly powered by a voltage source supply 42

Figure 2-15: Simulated flux of the motor when it is directly powered by a voltage source supply42

Figure 3-1: Overview of available control strategies 47

Figure 3-2: Principle of field oriented control 52

Figure 3-3: Configuration of the block diagram of indirect field oriented control scheme 57

Figure 3-4: Block diagram of Speed Control Loop 58

Figure 3-5: Root locus plot of the open loop transfer function with the PI 62

Figure 3-6: Block diagram of a feedback control system 63

Figure 4-1: Speed tracking response for PI and PI gain-scheduling controller techniques without applying load. 79

Figure 4-2: Torques of the PI and PI gain-scheduling controller techniques corresponding to the speed command given in Fig. 4-1. 79

Figure 4-3: Stator Phase Current of the PI and PI gain-scheduling controller techniques corresponding to the speed command given in Fig. 4-1. 79

Figure 4-4: Speed tracking response for PI and PI gain-scheduling controller techniques with load torque (2 Nm). 79

Figure 4-5: Torques of the PI and PI gain-scheduling controller techniques corresponding to the speed command given in Fig. 4-2. 79

Figure 4-6: Stator Phase Current of the PI and PI gain-scheduling controller techniques corresponding to the speed command given in Fig. 4-2. 79

Figure 4-7: Schematic of the Self-Tuning control system 84

Figure 4-8: Integral Squared (ISE) speed error versus controller parameter grid 86

Figure 4-9: flowchart of the proposed methodology 88

Figure 4-10: Initial speed tracking response with PI controller with load torque (2 Nm) 90

Figure 4-11: Initial speed tracking response with PI controller with load torque (2.Nm) 90

Figure 4-12: Initial speed tracking response with PI controller with load torque (2 Nm 90

Figure 4-13: Final speed tracking response with PI controller with load torque (2. Nm) 90

Figure 4-14: Final speed tracking response with PI controller with load torque (2. Nm) 90

Figure 4-15: Final speed tracking response with PI controller with load torque (2 Nm) 90

Figure 4-16: Initial speed tracking response with PI controller with load torque (2 Nm) 91

Figure 4-17: Initial torque developed for conventional and self-tuning PI controller corresponding to the speed given in figure. 4-16 91

Figure 4-18: Initial one-phase stator current for conventional and self-tuning PI controller corresponding to the speed given in figure. 4-16 91

Figure 4-19: Final speed tracking response with PI controller with load torque (2 Nm) 91

Figure 4-20: Final torque developed for conventional and self-tuning PI controller corresponding to the speed given in figure. 4-16 91

Figure 4-21: Final one-phase stator current for conventional and self-tuning PI controller corresponding to the speed given in figure. 4-16 91

Figure 5-1: Membership Functions (MFs) for Input Error (E) 100

Figure 5-2: Membership Functions (MFs) for Input Integral Error (IE) 101

Figure 5-3: Crisp Output Membership Functions 103

Figure 5-4: Speed tracking response for PI & FLC techniques with rated load torque (2Nm) applied at 0.6 sec. 106

Figure 5-5: Developed motor torque for PI & FLC techniques with load torque (2Nm) applied at 0.6 sec 106

Figure 5-6: Stator phase currents for PI & FLC techniques with load torque (2. Nm) applied at 0.6 sec. 106

Figure 5-7: Speed tracking response for PI & FLC techniques with load torque (2Nm) applied at 0.6 sec. 106

Figure 5-8: Developed motor torque for PI & FLC techniques with Load torque (2Nm) applied at 0.6 sec. 106

Figure 5-9: Stator phase currents of IM for PI & FLC techniques with load torque (2Nm) applied at 0.6 sec. 106

Figure 5-10: Experimental speed response of the drive for the step changes of the command speed based on PI controller, (div=20 rad/sec) 107

Figure 5-11: Experimental speed response of the drive for the step changes of the command speed based on FLC controller, (div=20 rad/sec) 108

Figure 5-12: Membership Functions (MFs) for Input Error (E) 113

Figure 5-13: Membership Functions (MFs) for Input Integral Error (IE) 114

Figure 5-14: Crisp Output Membership Functions 116

Figure 5-15: Speed tracking responses at no load, 120 rad/sec 119

Figure 5-16: Speed tracking responses at load torque (2 Nm), 125 rad/sec 119

Figure 5-17: Torque developed at rated load, corresponding to the speed given in figure 5-16 119

Figure 5-18: Speed tracking responses at no load, 125 rad/sec 119

Figure 5-19: Speed tracking responses at load torque (2 Nm), 180 rad/sec 119

Figure 5-20: Torque developed at rated load, corresponding to the speed given in figure 5-19 119

Figure 5-21: Stator phase currents corresponding to the speed given in figure 5-16 120

Figure 5-22: Stator phase currents corresponding to the speed given in figure 5-19 120

Figure 5-23 Experimental speed response of the drive for the step changes of the command speed based on FLC,(div=20 rad/sec) 121

Figure 5-24: Experimental speed response of the drive for the step changes of the command speed based on FLC,(div=20 rad/sec) 121

Figure 5-25: Experimental speed response of the drive for the step changes of the command speed based on FLC,(div=20 rad/sec) 121

Figure 5-26: Experimental speed response of the drive for the step changes of the command speed based on PI controller,(div=20 rad/sec) 121

Figure 5-27: Experimental speed response of the drive for the step changes of the command speed based on PI controller,(div=20 rad/sec) 121

Figure 5-28: Experimental speed response of the drive for the step changes of the command speed based on PI controller,(div=20 rad/sec) 121

Figure 5-29: Impact of changing output fuzzy sets' breakpoints on speed performance 123

Figure 5-30: Impact of changing output fuzzy sets' breakpoints on speed performance 124

Figure 5-31: Impact of changing output fuzzy sets' breakpoints on torque performance 125

Figure 5-32: The resulting corresponding outputs controller signals from changing output fuzzy sets breakpoints 126

Figure 5-33: Output membership function when integral is set at 0% N and 100% P, while the error is changed 128

Figure 5-34: Output membership function when integral is set at 50% N and 50% P, while the error is changed 128

Figure 5-35: Output membership function when integral is set at 100% N and 0% P, while the error is changed 129

Figure 5-36: the relation between output fuzzy logic against the error 130

Figure 6-1: Multilayer feed-forward artificial neural network 135

Figure 6-2: Neuron structure of the example 136

Figure 6-3: Neuron network with a single input and output. 137

Figure 6-4: Sigmoid Activation Function 138

Figure 6-5: Neural network map 139

Figure 6-6: Global e^2 versus weight bowl 141

Figure 6-7: Illustration of vector control scheme using ANN 144

Figure 6-8: Artificial Neural network architectures 145

Figure 6-9: Sigmoidal functions 146

Figure 6-10: Sigmoid function and derivative of sigmoid function 147

Figure 6-11: Simulated response of the PI controller-based IM to step changes of command speed with applying load torque (2 Nm). 151

Figure 6-12: Simulated torque of the PI controller-based IM drive under the application of load torque (2 Nm). 151

Figure 6-13: Simulated currents of the PI controller-based IM drive under the application of load torque (2 Nm). 151

Figure 6-14: Simulated response of the ANN controller-based IM to step changes of command speed with applying load torque (2 Nm). 151

Figure 6-15: Simulated torque of the ANN controller-based IM drive under the application of load torque (2 Nm). 151

Figure 6-16: Simulated currents of the ANN controller-based IM drive under the application of load torque (2 N m). 151

Figure 6-17: Experimental speed response of the drive for the step changes of the command speed based on the PI controller, (div=10 rad/sec) 153

Figure 6-18: Experimental speed response of the drive for the step changes of the command speed based on the ANN controller, (div=10 rad/sec) 153

Figure 6-19: Adaptive diagram of adjusted weights 155

Figure 6-20: Flowchart of self-tuning weights using grid search optimization method 157

Figure 6-21: Flowchart illustrating the process of selecting optimum weights using GSM 158

Figure 6-22: Error Square (e^2) versus weights (W_i, W_j) bowl 160

Figure 6-23: Error Square (e^2) versus weights (W_i) 161

Figure 6-24: Initial speed response at 100 rad/sec without load 163

Figure 6-25: Initial speed response at 100 rad/sec for load (2 Nm) applied at 0.5 sec 163

Figure 6-26: Initial speed response at 100 rad/sec for load (2 Nm) applied at 1 sec. 163

Figure 6-27: Final speed response at 100 rad/sec without load 163

Figure 6-28: Final speed response at 100 rad/sec for load (2 Nm) applied at 0.5 sec 163

Figure 6-29: Final speed response at 100 rad/sec for load (2 Nm) applied at 1 sec. 163

Figure 6-30: Initial speed response at 115 rad/sec for load (2 Nm) applied at 1 sec 164

Figure 6-31: Initial speed response at 150 rad/sec for load (2 Nm) applied at 0.6 sec 164

Figure 6-32: Initial developed torque corresponding to the speed illustrated in figure (6.24) 164

Figure 6-33: Final speed response at 115 rad/sec for load (2 Nm) applied at 1 sec 164

Figure 6-34: Final speed response at 150 rad/sec for load (2 Nm) applied at 0.6 sec 164

Figure 6-35: Final developed torque response corresponding to the speed illustrated in figure (6.27) 164

Figure 6-36: Initial current response corresponding to the speed illustrated in figure (6.24). 165

Figure 6-37: Final current response corresponding to the speed illustrated in figure (6.27). 165

Figure 7-1: Construction of quadrilateral element 171

Figure 7-2Four-node quadrilateral element of Lagrange interpolation functions 172

Figure 7-3: Four-node quadrilateral element of Lagrange interpolation functions 172

Figure 7-4: Triangle used to determine binomial coefficients 174

Figure 7-5: Schematic diagram of vector control of induction motor 178

Figure 7-6: Output surface of proposed finite element controller based on 2-D PI shape function (speed controller) 179

Figure 7-7: Sets of elements interconnected with a set of specified nodes 180

Figure 7-8: Speed tracking response for FECM technique with load (2 Nm) applied at 1.04 sec.184

Figure 7-9: Developed torques of the FECM technique corresponding to the speed command given in Fig. 7-7 under the application of load (2 Nm) at 1.04 sec. 184

Figure 7-10: Stator Phase Current of the FECM technique corresponding to the speed command given in Fig. 7-7 under the application of load (2.0 Nm) at 1.04s sec. 184

Figure 7-11 : Speed tracking response for FECM and PI controller techniques with load (2 Nm) applied at 0.5 sec. 184

Figure 7-12: Developed torques by FECM and PI controllers corresponding to the speed command given in figure7-10 under the application of load (2 Nm) at 0.5 sec. 184

Figure 7-13: Stator Phase Currents of the FECM and PI controllers corresponding to the speed command given in figure 7-10 under the application of load at 0.5 sec. 184

Figure 7-14: Four Node configurations for five discrete elements 186

Figure 7-15 Element geometry and local coordinate system for eight node quadrilateral configration 189

Figure 7-16: Five Element for an eight node curve sides quadrilateral 192

Figure 7-17: Finite element controller map (FECM) with eight nodes. 193

Figure 7-18: Flowchart of adjusting nodes using grid search optimization method 196

Figure 7-19: Initial speed tracking response with FECMM without load 198

Figure 7-20:. Initial speed tracking response with FECMM for load (2 Nm) 198

Figure 7-21: Initial motor torque corresponding to the speed given in figure 7-18 198

Figure 7-22: Speed tracking response with FECMM controller without load 198

Figure 7-23: Speed tracking response with FECMM controller for load (2 Nm) 198

Figure 7-24: Developed motor torque corresponding to the speed given in figure 7-21. 198

Figure 7-25: Initial stator current corresponding to the speed given in figure 7-18 199

Figure 7-26: Stator current corresponding to the speed given in figure 7-21 199

Figure 7-27: Initial start of speed tracking response for PI & FECM controllers without load 201

Figure 7-28: Simulation response for speed tracking of PI & FECM controllers at the end of the time period without load 201

Figure 7-29: Initial start of speed tracking response for PI & FECM controllers with load torque (2 Nm) 201

Figure 7-30: response for speed tracking of PI & FECM controllers at the end of the time period with load torque (2 Nm) 201

Figure 7-31: Initial speed tracking response for PI & FECM controllers with load torque (1 Nm) of full rated load 202

Figure 7-32: Simulation response for speed tracking of PI & FECM controllers at the end of the time period with load torque (1 Nm) 202

Figure 7-33: Simulated torques of the PI & FECM controllers corresponding to the speed command given in figure 7-30 under the load torque (1 Nm) 202

Figure 7-34: Simulated currents of the PI & FECM controllers corresponding to the speed command given in figure 7-30 under the load torque (1 Nm) 202

Figure 7-35 Experimental setup for IM drive using DSP board DSS1104 207

Figure 7-36: Experimental speed response of the drive for step changes of command speed based on PI controller, (div=10 rad/sec) 209

Figure 7-37: Experimental speed response of the drive for the step changes of the command speed based on based on PI controller, (div=10 rad/sec) 209

Figure 7-38: Experimental speed response of the drive for step changes of command speed based on FECM controller for five element, (div=10 rad/sec) 209

Figure 7-39: Experimental speed response of the drive for the step changes of the command speed based on FECM controller for five element, (div=10 rad/sec) 209

Figure 7-40: Experimental speed response of the drive for the step changes of the command speed based on FECM controller for nine element, (div=20 rad/sec) 211

Figure 7-41: Experimental speed response of the drive for the step changes of the command speed based on FECM controller for nine element, (div=20 rad/sec) 212

Figure 8-1: Speed tracking responses noload, 180 rad/sec 217

Figure 8-2: Speed tracking responses at applying load 2 N.m, 180 rad/sec 217

Figure 8-3: Speed tracking responses at applying load 4 N.m, 180 rad/sec 218

Figure 8-4: Four-node quadrilateral element 222

Figure C-9-1Schematic diagram of indirect field oriented control 257

Figure C-1 (a): Current flux and torque components 232

Figure C-1 (b): Rotor flux angle 232

Figure C-2 Hysteresis current controller 233

Figure C-3 voltage source inverter 234

Figure D-1: Matlab function which generate the Fuzzy logic Program 236

Figure D-2: Matlab function which generate 9 rules Fuzzy logic Program 238

Figure F-2: Matlab function which generate Finite Element Controller for Five Elements with Eight Nodes 244

Figure F-3: Matlab function which generate Finite Element Controller Map for Five Elements with Eight Nodes when the middle element takes a parabolic shape 247

List of Tables

Table 2-1: Switching logic of VSI for phase ‘a’ 38

Table 4-1: Conventional and self-tuning PI controller parameters 89

Table 5-1: Fuzzy rules 111

Table 5-2: results obtained from obtained from each rules 117

Table 7-1: Control Signal at Nodes 180

Table 8-1: Rise time for conventional PI, fuzzy logic, neural network and finite element controllers 216

Table 8-2: Settling time for conventional PI, fuzzy logic, neural network and finite element controllers 219

Table 8-3: Root Mean Square Error for Fuzzy logic, neural network and Finite element controllers 220

List of Symbols

Symbol	Meaning
$a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8$	Binomial coefficients
abc	Three phase quantity
a	Nodal value of error (E)
B	Friction coefficient
b	Nodal value of integral error (I)
$b_1, b_2, b_3, b_4, b_5, b_6, b_7, b_8$	Binomial coefficients
c	Constant coefficient
E	Error
G_{QN}	Represents the width of negative fuzzy set
G_{QP}	Represents the width of positive fuzzy set
H_{QN}	Represents the moments arm of the centre area of negative fuzzy set
H_{QP}	Represents the moments arm of the canter area of positive fuzzy set
I_a, I_b, I_c	Three AC phases currents a, b and c
$i_{\alpha s} - i_{\beta s}$	Stator current in stationary reference frame (also known as α - β axes)
$d^e - q^e$	Synchronously rotating reference frame (or rotating frame) direct and quadrature axes

$d^s - q^s$	Stationary reference frame direct and quadrature axes (also known as α - β axes)
$i_{dr} - i_{qr}$	$d-q$ - Axis rotor current
$i_{ds} - i_{qs}$	$d-q$ - Axis stator current
$i_{ds}^* - i_{qs}^*$	$d-q$ - Axis stator current reference
$i_{ds}^e - i_{qs}^e$	$d^e - q^e$ - Axis stator current
$i_{ds}^s - i_{qs}^s$	$d^s - q^s$ - Axis stator current
i_a^*	Phase command current
i_s	Stator current
I	Input of neural network (chapter 6)
I	Horizontal direction (chapter 6)
I	Integral of error (chapter 7)
J	The moment of inertia
J	Integral of the speed error (chapter 6)
K	Iteration stability
$K_{p(\min)}$	Minimum proportional gain
$K_{p(\max)}$	Maximum proportional gain
$K_{i(\min)}$	Minimum integral gain
$K_{i(\max)}$	Maximum integral gain

K_t	Torque constant
L_{2-3}	Local coordinate line that passes through nodes 2 and 3
L_{3-4}	Local coordinate line that passes through nodes 3 and 4
L_{ls}	Stator leakage inductance
L_{lr}	Rotor leakage inductance
L_m	Mutual inductance
L_r	Rotor self-inductance.
L_s	Stator self-inductance
M	Output of the hidden layer neuron
ΔM	The change of M during training step
m	Nodal value of controller output(Q)
M_i	Shape function
M_p	Maximum overshoot
$M_1, M_2, M_3, M_4, M_5, M_6, M_7, M_8$	Shape functions
N	Negative fuzzy set
$N_1, N_2, N_3, N_4, N_5, N_6$	Logic signals for upper and down transistors of the inverter, (chapter 2)
O	Output of the neural network controller
O	Vertical direction
O/I	Output/input

O_N	Net speed
O_T	Target speed
p	Differential operator
P	Number of poles
q_n, q_s, q_e, q_p, q_w	North, South, East, Central and West grid nodal values
R_r	Rotor resistance
R_s	Stator resistance
S	Slip
$S_1, S_2, S_3, S_4, S_5, S_6$	Inverter switches
t_s	Settling time
T_e	Developed torque
T_L	Load torque
T^*	Reference torque
Q_i	Control nodal values
Q_o	Sum of the scaled shape
V_{ab}	Phase to phase voltage
V_{ao}, V_{bo}, V_{co}	Line to neutral (or phase voltage)
$v_{ds} - v_{qs}$	$d^e - q^e$ Axis stator voltage
$v_{dr} - v_{qr}$	$d^e - q^e$ Axis rotor voltage (referred to stator)

$v_{ds}^{e} - v_{qs}^{e}$	$d^{e} - q^{e}$ Axis stator voltage (referred to stator)
$v_{ds}^{s} - v_{qs}^{s}$	$d^{s} - q^{s}$ Axis stator voltage (referred to stator)
V_{dc}	Inverter DC voltage
V_{L}	Inverter voltage vector
$V_{T}, V_{t}, V_{B}, V_{b}$	Element edge value (chapter 7)
W_{I}	Input weight
$W_{bx},\ W_{by},\ W_{bz}$	Input bias weights of the neurons $X,\ Y,\ Z$
$W_{I(\max)}, W_{IU}$	Error weight grid upper limit
W_{IO}, W_{IL}	Error weight grid lower limit
W_{IB}	Input bias of the single neural network
W_{IM}	Input weight for single neuron
$W_{\mathrm{I(NEW)}}$	New value of error weight
$W_{\mathrm{I(OLD)}}$	Previous value of error weight
$W_{ix},\ W_{iy},\ W_{iz}$	Input weights of the neurons $X,\ Y,\ Z$
$W_{ie}, W_{ip}, \mathrm{W}_{iw}$	East, middle and west of the grid weights value
$W_{jx},\ W_{jy},\ W_{jz}$	Input weights of the neurons $X,\ Y,\ Z$
$W_{jn}, W_{jp}, \mathrm{W}_{js}$	North, middle and south of the grid weights value
$W_{\mathrm{J(max)}}$	Integral weight limit
$W_{J\,(\mathrm{NEW})}$	New value integral weight of grid

$W_{J(\mathrm{OLD})}$	Previous value integral weight of grid
W_{jO}	Integral initial or starting weight
W_O	Output weight
W_{ob}	Output bias of the neural network
W_{OM}	Output weight of the middle single neuron network
$W_{ox},\ W_{oy},\ W_{oz}$	Neuron output weights
$W_{O(\mathrm{NEW})}$	New value of the output weight
$W_{O(\mathrm{OLD})}$	Previous value of the output weight
$\Delta W_i, \Delta W_j$	Grid weights step size
ΔW_{o}	Output weights step size
X, Y, Z	Three neurons in the hidden layer
Z_a, Z_b, Z_c	Load impedance
$\alpha - \beta$	Axis of stationary reference frame (also known as α - β axes)
$\alpha - \beta - \varepsilon$	Local coordinates
ζ	Damping ratio
θ_e	Angle of synchronously rotating frame ($\omega_e t$)
θ_{sl}	Slip angle
θ_r	Rotor angle
$\mu_{c(k)}$	The geometric centre of the output fuzzy value

τ_r	Time constant
$\psi_{dr} - \psi_{qr}$	$d-q$ - Axis rotor flux linkage
$\psi_{ds} - \psi_{qs}$	$d-q$ - Axis stator flux linkage
$\psi_{ds}^{s} - \psi_{qs}^{s}$	$d^s - q^s$ - Axis stator flux linkage
ψ_r	Rotor flux linkage
ψ_r^*	Rotor flux command
$\vec{\psi}_r$	Vector rotor flux
ψ_s	Stator flux linkage
ω_e	Stator or line frequency (2πf) (rad/s)
ω_n	Natural frequency
ω_r	Rotor electrical speed (rad/s)
ω_r^*	Speed command
ω_{sl}	Slip speed
d_3, d_6	Output of adjustable finite element nodes
d_{3O}, d_{6O}	Initial guess of adjustable finite element nodes
$\Delta d_3, \Delta d_6$	Step size of adjustable finite element nodes
$d_{3(\max)}, d_{6(\max)}$	Saturation limits of adjustable finite element nodes
E_0, I_0	Centre values of the finite element
G, X, Y, Z	Coefficients in α equation

H,U,V,W	Coefficients in β equation

List of Abbreviations

A/D	Analog to digital
ac	Alternating current
AIC	Artificial Intelligent Controller
ANN	Artificial neural network
BE	Error breakpoint
BI	Integral break point
BLDC	Brushless DC
CSI	Current source inverter
d	Direct axis of rotating frame
D/A	Digital to analogue
DC	Direct current
DFOC	Direct field oriented control
div	Division
DMF	Degree of membership function
DSC	Direct self control
DSP	Digital signal processor
DTC	Direct torque control
E	Error
EMF	Electromotive force
FECM	Finite element controller map
FLC	Fuzzy logic controller

FOC	Field orientation control
FW	Flux weakening
GSM	Grid search method
GTO	Gate turn-off thyristor
hp	Horsepower
HPD	High performance drive
HPVSD	High performance variable speed drive
I	Integral of error
IDFOC	Indirect field oriented control
IGBT	Inverted gate bipolar transistor
IM	Induction motor
IPMSM	Interior permanent magnet synchronous motor
ISE	Integral square error
LCR	Linguistic control rule
LN	Large negative
LP	Large positive
MI	Machine intelligence
MRAC	Model reference adaptive control
N	Negative fuzzy set
N	Number of samples
N	The total number of rules
NA, NB, NC	Three binary logic variables of the three legs

NFC	Neuro-fuzzy logic controller
NN	Negative negative
NN	Neural network
NNC	Neural network controller
NP	Negative positive
p	Differential operator
P	Number of poles
P	Positive fuzzy set
PC	Personal computer
PDE	partial differential equations
PF	Pre-filter
PI	Proportional integral
PIAW	Proportional Integral with anti-windup
PID	Proportional integral derivative
PM	Permanent magnet
PMSM	Permanent magnet synchronous motor
PN	Positive negative
PP	Positive positive
PWM	Pulse width modulation
q	Quadrature axis of rotating frame
Q	Control signal
SMC	Sliding mode control
SN	Small negative

SP	Small positive
SRC	Silicon controlled rectifier
SVM	Space vector modulation
SVPWM	Space vector pulse width modulation
V/f	Volts per frequency
VSC	Variable structure control
VSI	Voltage source inverter
VFD	Variable Frequency Drive
2-D	Two-dimensional
3-D	Three-dimensional

Chapter 1.

Control of Motor Drives

1.1 Introduction

Induction Motors (IMs) have been increasingly used with variable speed drives (VSDs) due in large part to their solid construction, relative affordability, ease of maintenance, and reliable service. The controller plays a key role in VSD applications in order to enable the motor to adhere to reference trajectories without major deviations. High performance drive systems require a controller that can offer fast speed response and easily deal with uncertainties and disturbances. Fixed gain proportional-integral (PI) controllers, along with some adaptive controllers, have typically been used in industry despite their disadvantages in dealing with uncertainties.

To improve the control performance, intelligent algorithms have been incorporated in the controller design. Intelligent algorithms, including fuzzy logic (FL), neural network (NN) and neuro-fuzzy (NF) algorithms, attempt to replicate human logic, making them suitable for applications that have poorly defined mathematical models. The intelligent systems, which are also known as high performance drive systems (HPDS), are characterized by a high degree of adaptive behaviour, robustness, and efficiency, as well as by high torque to current ratio and high power to weight ratio [1-6].

Due to advantages of HPDS, intelligent control of electric motor drives has attracted the attention of electrical engineers in recent years. In the present research, a neural network has been devised for IM speed control. Indirect field orientation control is used

to decouple the torque and flux controls in the complete drive. In this way, the induction motor can be operated like a DC motor and high performance can be obtained, but current which flows through the induction motor is a time varying current.

1.2 Literature Review

Induction motors are widely used in various industrial sectors, the control of IMs remains a challenging issue as the dynamic system is non-linear and two of the state variables, rotor fluxes and currents, are not always measurable. The rotor resistance varies significantly caused by heating, which has a major effect on the system dynamics. Over the past few decades, in response to IM challenges faced by industry, researchers made concentrated efforts to develop a high-performance control theory for AC drives [3-7].

One task in the process of IM control system design involves two levels: first, choosing a control strategy, and second, designing a proper controller. IM control strategies can be broadly classified into two categories: scalar control and vector control. In the scalar control, only the magnitude and frequency (angular speed) of voltage, current, and flux linkage space vectors are controlled. The most common scalar technique is constant volts/frequency (V / f) control [1]. In the V / f control method, the magnitude of the stator voltage is adjusted in proportion to the frequency in order to maintain a constant stator flux in the IM. The V / f method consists of controlling the speed of the rotating magnetic field of the stator by changing the supply frequency. However, because the scalar control strategy is based on a steady-state principle, its transient performance is not optimized. A vector control is a more efficient IM control strategy [3] [63-65].

1.3 Vector control based Drive

Variable frequency drives (VFDs) provides significant advantages for the operation of induction motors. These advantages include reduced starting currents, variable speed control, and improved energy efficiency. VFDs mainly have two types of controllers, one is called a Direct Torque Controller (DTC), and the other one is vector control. Both are used to obtain high performance drive. The two types of controllers have advantages and disadvantages. For instance, in the direct torque con

troller, the torque is attained directly and independently to the flux linkage, through the selection of the optimal switching modes. Advantages of the DTC is the fast torque response, low inverter switching frequency, and low harmonic losses, in contrast, the main disadvantages are high torque and flux ripples and variable switching frequencies [60-61].

Secondly, vector control, also named field oriented control (FOC), is considered to be a variable frequency drive (VFD), which provides significant advantages for the operation of induction motors. These advantages include reduced starting currents, variable speed control, and improved energy efficiency.

Vector control is developed to obtain high performance operation of AC motor drives. FOC is used to control any application of AC motors either induction or synchronous that requires to operate precisely and smoothly over the full speed range, generate maximum torque at low speed, and have high dynamic performance including fast and accurate speed response as well as quick recovery speed from any disturbances, and possess insensitivity to parameter variations. By this method the induction motor can be controlled similar to a separately excited dc motor. In a vector control method, the three-

phase stator currents are converted to two orthogonal current components. One component defines the magnetic flux of the motor, the other defines the torque. The corresponding flux and current components references provided by the drive's speed controller is calculated by the control system of the drive. Typically proportional-integral (PI) controllers are used to keep the measured current components at their reference values. The pulse-width modulated variable-frequency drive generates gate signals, which will fire the transistor switching of the three-phase inverter to produce the actual voltages to the motor.

1.4 PI, PID and Adaptive Controllers

Commonly used industrial control drives for IMs include proportional-integral (PI) controllers, proportional-integral-derivative (PID) controllers, and adaptive versions of both PI and PID [9] [11] [30-34] [70-76] [90]. Due to their simplicity and ease of implementation in real-time experimental systems, conventional PI and PID controllers have been widely used as speed controllers. Moreover, Space Vector Pulse Width Modulation (SVPWM) technique that utilizes Digital Signal Processing (DSP) is also commonly used to control IM. SVPWM can prevent harmonic distortion in currents and output voltage, and enabled more efficient use of DC supply voltage than sinusoidal modulation techniques. However, the motor control scheme needed three PI controllers, and it was often difficult to identify any gains made by the controllers. The fixed gains were also impacted by parameter variations [10].

In [1], Talib *et al.* [11] used different means for measuring speed performance. Namely, they looked at the Indirect Field Oriented Control (IFOC) induction motor drive by

utilizing (1) PI control, (2) PI with anti-windup (PIAW), and (3) pre-filter (PF). Their aim was to compare the controller performance on the motor in relation to speed tracking and load rejection capability in speed operations designated as low-, medium- and high-rated. Conventional or linear PI controllers usually do not have output magnitude limiters, as these could damage the physical system because of their large output values. In this regard, introducing integrator limiters and saturation limiters could offer some protection to the system. However, saturation limiters can also cause errors such as system instability, sizeable overshoots, and slow settling time.

To prevent high overshoots and long settling times, PI controllers with anti-windup were introduced, following on the notion that inserting a pre-filter block can equalize the behaviour of the closed speed loop system in relation to the desired second order system. While it can get rid of the zero from the loop gain, this strategy still required three PI controllers: one for the outer speed control loop and two for the inner current loops. The motor performance is based on PI controller gains, so the controllers must be accurately tuned. However, any gains made must be gauged by time-consuming trial-and-error tuning techniques, such as the Ziegler-Nichols method and the first-order-plus-time-delay method. A certain degree of basic knowledge of process control is required to utilize these methods, even though they rarely provide the best performance control. Furthermore, the wind-up phenomenon leads to inconsistencies between the real plant input and the controller output.

Most published work in the literature concerns conventional PI- and PID-based speed controllers for IM drive performance in the normal range [62] [69-70]. The primary advantage of these types of controllers is their ease of implementation in real time.

However, they are highly sensitive to changes in parameters caused by sudden shifts in command speed, load, saturation, temperature, etc. Because of this inherent sensitivity, precise tuning of the controller parameters can be challenging, so these types of controllers are not optimal choices for high-performance applications.

The controllers are highly sensitive to uncertainties such as changes in parameters due to saturation, variations in temperature, sudden changes in command speed, and load disturbances. It is also difficult to precisely tune controller parameters for implementations on- and off-line, so these types of controllers are generally unsuited for high performance applications. In response to these limitations, researchers [12-15] are looking into developing adaptive control schemes for IM drive systems that enable the controller to adapt its parameters to system parameter variations and load disturbances. The ready availability of cost-effective digital signal processors (DSP) has enabled researchers to add these adaptive controllers to IM drives.

1.5 Adaptive and Intelligent Control of Induction Motor Drive

Rather than adopt a typical approach to speed-controlled drives that use PI-type controllers, industry can use a variable structure control (VSC) scheme to boost product speed and robustness through a sliding mode of control. The main objective is to limit the controlled variables to the tolerance band by inciting an immediate reaction to a disturbance. There are two main types of such control mechanisms – the relay type and the relay-linear type – while the adaptive control techniques themselves are usually characterized as having self-tuning control, model reference adaptive control (MRAC) [77-78] [80] [86] [118], sliding mode control (SMC) or variable structure control [8],

expert system control, fuzzy control [5] [22-23], neural control [24] [99-102] , or various combinations of these approaches.

The latest advances in electronic technologies and VSI techniques have resulted in the use of microprocessors in IM drive controls. This has led to computing power being applied to IM drive control and the implementation of complex control strategies, such as using the basic principles of machine intelligence (MI)-based controllers in high-performance drives (HPD). The core notion behind MI systems is to mimic human cognitive abilities, which makes them highly effective in complex applications and mathematical models that are poorly defined. Due to inherent adaptive and robust features, they are especially useful in high-performance IM drives.

There are three main types of controllers: neural controllers, fuzzy controllers, and a combination of the two called hybrid neuro-fuzzy controllers. As its name implies, this latter combination type includes the operating principles of both neural and fuzzy controllers and it is positioned to become the most significant approach to managing control systems in the future. Because this thesis focuses on the application of a variety of adaptive controllers (e.g., NN, Finite Element Controller Map [FECM] and the adaptive PI controller in the IM drive), the literature review follows the development of these control techniques, in particular.

Numerous researchers have published work on the development of high performance induction motor drives. Most of the control techniques that have thus far been developed feed the IM via either a voltage source or current source inverter. Plunkett and Plette [16-17] created a control scheme that utilized the stator current rather than inverter voltage control via the flux loop. The voltage-fed inverter drive was developed with the torque

and flux control at the outer-loop, and the hysteresis-band current control at the inner loop. Moreover, instead of applying a constant rated flux, to enhance the efficiency, the flux is programed as a function of torque.

In [20], Boussak used the model-reference adaptive system (MRAS) to estimate IM speed from measured terminal voltages and currents. Boussak's method proved to be more effective as well as less complicated when compared with similar approaches. Meanwhile, Rowan *et al.* [21] presented a method for online IM-drive tuning that makes use of the model reference adaptive control (MRAC). In their approach, they compared the reference model's output and that of either an adjustable or adaptive model. The comparison proceeds until any errors between the two models are eliminated. Although this type of system is characterized as robust, the model functions adequately only if the machine parameters are constant; if parameters vary, the control system decreases. .

Several other adaptive techniques address indirect field-oriented IM drives, but each has advantages and disadvantages, none of them have provided ideal all-round decoupling control combined with the maximum efficiency, nor have they offered a viable solution to detuning issues.

In the wake of Zadeh's [22-23] delineation of fuzzy logic control principles in 1965, several applications of fuzzy controllers in power electronics and drives have emerged [24] [27-29] [36-37] [66]. FLCs have some advantages compared to the conventional PID. These are: they have the ability to handle nonlinearity of arbitrary complexity; they do not need an accurate mathematical model; they are occasionally more robust against load disturbance, and system parameter variation than conventional controllers [143];

and finally, they are designed based on human logic through applying linguistic rules with an IF–THEN structure, which is the basis of human logic.

Even though FOCs have their advantages, they also have their critics, particularly among the conventional PI controllers. However lacks of analysis methodologies as well as a lack of formal design are considered common weakness. This includes the rigors in obtaining stability and robustness [144-145]. Furthermore, the application of an FLC has encountered some disadvantages during hardware and software implementation due to high computational burden [93] [145-146]. To reduce the real time computational burden of an FLC, a method based on simple membership functions (MFs) and rules has been implemented in [5] [38] [93].

The Fuzzy logic (FL) has emerged as a complement to conventional strict methods. Design objectives, which are mathematically hard to express can be incorporated into FLC by linguistic rules. The FLC have been developed and can be split into two groups [120]. Improving the design and performance of the standard FLC was the one of concentration of the first group, While the second group of approaches combine the advantages of FLC and those of conventional adaptive techniques [28] [38] [120].

Most FLCs were initially designed by trial and error. Because the complexity of an FLC raising exponentially particularly when it is developed to control complex systems, it is troublesome to design and tune FLCs manually for most industrial problems like control of motors. That is why, the conventional nonlinear design method [114] was adopted in the fuzzy control area, such as fuzzy sliding control [75-76] [149], fuzzy gain scheduling [118], various forms of self-tuning and self-organizing FLCs [36], and adaptive fuzzy control [98] [148], in order to alleviate difficulties in constructing the fuzzy rule base and

improve the performance of the drive under severe perturbations of model parameters and operating conditions.

Furthermore, Bose [24] has proposed fuzzy control of vector-controlled induction motor. In his study, fuzzy control is used to achieve robustness against parameter variation and load torque disturbance effects. Various control look-up rule tables are used to improve the transient response and system settling time. The disadvantage of the scheme is the large number of fuzzy rules that require huge computational burden.

In the same vein, Masiala et al. [120] have reported Self-Tuning fuzzy Speed Control of an Indirect Field-Oriented Control Induction Motor Drive to improve and reduce the sensitivity of the drive to motor parameter changes and load disturbances. The controller has the ability to adjust its parameters online relative to the error between actual speed and model reference. On the application of fuzzy logic controller for inverted fed IM, Mir and Zinger [21] have used fuzzy logic to select switching states whose variables include: flux position, errors in flux magnitude and torque. The enhancement of system performance at low speeds is achieved with a fuzzy resistance estimator which eliminates the error due to the change in stator resistance. A change in stator resistance of the induction machines would lead to an error in the stator current. This error is utilised by the fuzzy resistance estimator to correct the stator resistance used by the controller to match the machine resistance.

The study on fuzzy logic application for intelligent control of a variable speed drive was made by Tang and Langya [27]. They developed a direct fuzzy logic controller and an adaptive fuzzy controller, based on model reference adaptive control for the doubly excited machine and converter system. Fuzzy logic was applied for the intelligent

control of a slip power recovery system. In comparison with the field orientation control, the intelligent control of the complex slip recovery system reduces costs and enhances robustness and desired performance. Ibaliden and Goureau [150] have experimented with the static and dynamic fuzzy controller for the speed control of an induction motor by comparing it to the conventional PI controller. They suggested the use of a dynamic controller consisting of two fuzzy controllers in order to enhance the performance of static fuzzy controller during parameter variations. Lai and Nakano [151] highlighted in their work, the use of a phase-locked loop IM speed drive incorporating a fuzzy logic controller. The relatively good speed regulation of the phase-locked loop techniques is combined with the advantages of fuzzy logic to obtain a robust, fast and precise control of induction motor speed. The results of their experiment demonstrated that the combination of a fuzzy logic controller and the phase-locked loop in IM drive could achieve precise speed control with fast response.

Heber et al [152] have used fuzzy logic scheme for field orientation control of IM to enhance speed control. The fuzzy logic design approach was used to overcome field orientation detuning caused by parameter uncertainties.

F. Barrero et al. [153]. This paper presents a new approach to indirect vector control of induction motors. Two types of nonlinear controllers are used, one is sliding mode type and the other one is PI-fuzzy logic-based. These two controllers are combined by means of an expert system based on Takagi–Sugeno fuzzy reasoning. The sliding-mode controller acts mainly in a transient state while the PI-like fuzzy controller acts in the steady state. The new structure embodies the advantages that both nonlinear controllers offer: sliding-mode controllers increasing system stability limits and PI-like fuzzy logic

based controllers reducing the chattering in permanent state. The scheme has been implemented and experimentally validated.

Liyong et al. [154] describes how to use a fuzzy logic controller (FLC) for indirect vector control induction motor (IM) drive. The fuzzy logic controller is employed in the outer speed loop. In this approach, two input variable are used, one represents the speed error and other represents the rate of change of the speed error. The output variable of the FLC controller is the torque. Though this work illustrates experimental results and shows comparisons between PI and FLC, it still lacks in results.

Likewise, Mishra et al. [155] compared Hysteresis Current Control (HCC) with Space Vector pulse modulation (SVPWM) to generate pulse width modulation (PWM) signals for voltage source inverter in vector controlled Induction Motor Drive. The dynamic performance and ripple content is analyzed using FLC based SVPWM and HCC IM drive and compared at different operating conditions. However, experimental tests are not provided, and its work only shows simulation results.

Zaky et al. [156] proposed a fuzzy logic controller (FLC) to alleviate the negative effects of motor parameter variation influence on indirect rotor field oriented control (IRFOC) without proposing a tuning method. The variation in stator resistance and load inertia is demonstrated. However, the critical parameters that essentially affect the vector orientation performance have not been investigated neither rotor nor magnetizing inductances were investigated.

Using a fuzzy logic controller based on indirect vector control, Rahman *et al*. [38] successfully developed a real-time application of the fuzzy logic controller for a high performance IM drive system. The researchers applied the fuzzy-logic speed controller

utilizing a digital signal processor board. This approach has been shown to be more robust and thus a prospective replacement for conventional PI controllers in high-performance industrial applications. However, this method is often constrained by over-simplification due to the assumption of the direct axis control current being zero. As a consequence, the motor draws higher than the rated current, resulting in a significant number of fuzzy rules and adding to the computational burden. Due to these unwanted side effects, the experimental implementation of this method was speed-limited. Furthermore, because the condition $i_d = 0$ is not a realistic constraint for an indirect motor controller at rated load conditions, there is clearly a gap in the research that points to the necessity of finding an approach that meshes with the presented scheme.

Many studies have attempted to use fuzzy logic principles along with other types of intelligent controllers (e.g., artificial neural networks [ANN]) to control IM drives at standard working conditions [37]. These are typically referred to as artificial intelligence (AI) controllers, as they utilize software programming that simulates human thought patterns in motor control [24-26]. However, there were primary issues surrounding the implementation of ANN control related to the heavy dependence on computerization. Most researchers use the Matlab ANN tool box to implement ANN control with many neurons. They only study control in simulation, and experimental results are often not provided for a system operated by ANN controller. This is due to the high computational burden required for these application. There is a connection between experimental results and designed controller. When the controller is simple, it is easy and does cost to be implemented. In response to this problem, subsequent studies were mostly theoretical in nature and based on simulations or low-speed experimental results. Such controllers

can be self-adaptive and do not require precise data on system parameters, even in cases where one parameter depends on another or on operating conditions. Furthermore, since intelligent controllers are created to be self-optimised and adaptive, they do not require information regarding system non-linearity.

More recent research projects have seen an increase in the application of ANN for IM drives as adaptive controllers by exploiting their non-linear input and output [4] [24-25] [99-102]. Tang and Xu [27-28] as well as Fonseca *et al*.[29], built a direct fuzzy logic controller and an adaptive fuzzy controller based on the model reference adaptive control for the doubly excited machine and converter system. They applied fuzzy logic to the intelligent control of a slip power recovery system. Compared to field orientation control, the intelligent control of a complex slip recovery system decreases costs while boosting robustness and performance levels.

Additionally, in order to achieve more improved performance and increased robustness, recently artificial neural networks is being used in designing such controllers [25-26] [98] [130-131]. Sathishkumar et al [157]. presents a neural network intelligent controller for a vector controlled induction motor drive. Comparison is made between a neural network and fuzzy logic. An evaluation of the drive performance is made when the system is subjected to disturbance effects. It is found that neural network based speed controller gives better tracking of speed of induction motor. A two-layer ANN has been trained off line to estimate the slip, the direct and the quadrature current commands that are to be used in implementing an indirect FOC.

Rahman and Hoque [130] have proposed an on-line adaptive ANN based controller for permanent magnet motor drive. Backpropagation learning algorithm is used in artificial

neural networks to calculate the gradient that is needed in the calculation of the weights. In this approach, two neural networks were used. One neural network's output gives the command signal while the other neural network's output gives the estimated signal. The weights and biases are calculated and updated according to the error resulting from the difference between the two output neural networks. Combined mixes of online and off-line training types were used. Vector control based on hysteresis current controller was used in this experiment. There were two weaknesses that appeared, the first weakness in the system was the controller since it was not able to push the speed of the motor above its speed limit. This along with the neural network combines with the vector control makes the system complex and not suitable for robust operation particularly during online training.

Online and offline neural models have their own advantages and disadvantages. Even an offline model can handle large data as computation time that not critical to their structure, it only robust to small variation, but fail to adapt to larger changes in the system. While, the online model quickly adapts to variation in the non-linear behavior of the system. The use of an offline training of NN to emulate the function of FOC has been proposed in [130] [158]. It shows that NN presents new solution to simplify the implementation of FOC. The input and output signal for training the NN are extracted from FOC of IM. The NN has been trained using a several condition to update their weight because of their limitation to larger changes of the system so that it can follow the speed trajectory specified by reference model.

Bohari et al. [159] proposed improvement performance from offline towards online neural network scheme for speed control of induction motor field oriented control based on load disturbance and parameter variation.

Likewise, Saxena et al. [160] designed an artificial neural network controller for a vector controlled induction motor drive. Back-propagation algorithm was used for training to update and obtain optimum weight and bias values. Also, the results were compared with conventional PI controller. However, while his work was good, it was limited only to the simulation results and lacks the experimental results.

Similarly**,** Ba-razzouk et *al*. [131] [161] an artificial neural based field oriented control of induction motors was presented. Learning algorithm is accomplished by the Levenberg-Marquart training algorithm which provided by the MATLAB Neural Networks toolbox. The second part of this paper presented a new scheme to estimate the rotor flux and the rotor time constant. This scheme involves only measurement of rotor mechanical speed and two stator voltages. Simulation results for this scheme are very attractive. However, experimental tests are not provided in [12].

Although much research work has been reported on the use of ANN and fuzzy logic for induction motor drives [53-76], one of the prominent works was described by Bose and Patel. They developed a combined algorithm, neuro fuzzy techniques to obtain a high-performance stator flux oriented speed sensorless for direct vector-controlled induction motor drive. The stator flux oriented drive starts from stand-still condition using stationary frame current model equations that do not require speed signal. The direct vector controlled drive synthesizes the stator flux vector signals from the machine

terminal voltages and currents. Some improvement is facilitated due to the use of this approach. Both simulation and experimental results are provided. On the other hand, this scheme has some drawbacks as it thoroughly relies on the look-up table and, to be practically carried out, it requires a powerful computational system to handle both on-line ANN and fuzzy logic processes at the same time.

1.6 Problem Identification

High Performance Drive (HPD) systems are widely used in robotics, machine tools, air conditioners, tractions, rolling mills etc. Among many specifications, there are three primary specifications for HPD systems to provide: (1) quick and accurate response times; (2) fast recovery to reference speed across a broad spectrum of potential sudden disturbances; and (3) relative immunity to variations in parameters. So far, existing HPD systems have experienced issues, which require further development of control algorithms and methods.

Compared to conventional motors, the most significant advantages of high performance induction motor drives are reliability, high torque to inertia ratio, high power to weight ratio, robustness, and high load ability. However, accurate mathematical model of IM should be considered nonlinearly due to the saturation effect involved with the practical application of IM drives.

. A main objective of this study is to build and implement an IM drive system for use in HPD applications. An indirect vector control scheme has been utilized which includes a speed controller as well as a current controller. This approach permits for the decoupling of the torque and flux, thus offering quicker and more transient responses and making control easier.

As discussed previously in the literature review, fixed-gain proportional integral (PI), proportional integral derivative (PID) controllers, model reference adaptive controllers (MRAC), variable structure controllers (VSC), and sliding mode controllers (SMC) require precise knowledge of system model parameters. Fixed-gain PI and PID controllers are especially sensitive to changes in parameters, along with load changes and similar disturbances. Thus, to handle a large number of potential variables and significant system nonlinearity, a suitable speed control system for IMs is needed.

Unlike the conventional PI, the adaptive controller changes parameter settings according to any differences between the desired and actual responses of the system. Specifically, an integrated squared difference is calculated and parameters are minimized to fit the adaptive controller, including handling non-linearities. The intelligent controllers of artificial neural network (ANN), finite element controller and adaptive PI controller are investigated in this study to gauge their potential for optimal IM speed control.

Some prior research on ANN applications for IM drives has already been done, but given the iterative nature of neural networks, the algorithms need to be trained either off- or on-line. In off-line training, ANN controllers require a significant amount of data in order to cover all operating conditions. Moreover, it is difficult for an off-line-trained ANN controller to deal with highly variable environments or system parameters. At the same time, online training algorithms require large amounts of computational overhead and thus restrict the frequency of the overall system. Therefore, to shorten the execution time of the ANN controller, a network structure with fixed weights has been developed. The main benefits of such a network structure are low execution time, high accuracy (compared to traditional ANN-based speed drives that are complex and labour-intensive,

particularly for on-line training control applications), and overall simplicity. The primary challenge here is that the values of the fixed weights rely on the mathematical models being accurate, which is not always possible.

In the basic PID controller, the parameters that can be adjusted are its three gains. Usually, the saturation limit is a controller parameter that is fixed. However, there is a limited amount of adjustment that can be done with just three parameters. A nonlinear polynomial type PID controller with more parameters can give a more complex map, and thus, a better fit to the desired response. Also, a piecewise PID controller can be developed where the gains change when PID inputs cross certain thresholds. In a situation when the gains are made a function of disturbance levels, the process is generally referred to as gain scheduling.

1.7 Thesis Objectives

This thesis presents several adaptive controllers design schemes for induction motors including a conventional PI Controller, a Fuzzy Logic (FL) PI Controller, an Artificial Neural Network (ANN) PI Controller, and a novel Finite Element PI Controller. Most works in the literature use MATLAB Toolboxes for Fuzzy Logic and Neural Network controllers. In this thesis, Matlab M code developed by the author is used to implement the controllers, this gave more control over the controller structure and parameters. It is found through this research that both FL and ANN controllers can be reduced to a standard PI controller. To make each controller self-tuning, only a few parameters in the controller map are allowed to be adjustable, which allows it to be easy implemented.

Few researchers develop controllers from basic concepts. No other researchers have used Finite Element controllers, which can have local character and could give robust control. The simple structure made the controllers easier to make adaptive.

1.8 Scope of Thesis Work

There were short and long term goals of the thesis work. The short term goal was to develop intelligent controllers that could mimic the conventional PI controller. This would prove that the intelligent controllers were developed properly. The structure of each controller was deliberately made simple to help make it adaptable. Coding was developed for each controller so that parameters would be accessible. Simulations and experiments showed that each controller could be made to mimic closely the

conventional PI controller. So they were developed properly. Since others mimic it, it is believed that they could also be made adaptive. The work to date shows that the intelligent controllers have been developed properly. The long term goal is to make the controller maps more complex so they can give performance beyond the conventional PI controller.

1.9 Organisation of the Thesis

This thesis consists of seven chapters:

In Chapter one, introduction and literature review of vector control techniques for IM drives, as well as the objectives of this work have been covered.

Chapter two introduces the indirect field orientation control (FOC) of VSI fed based induction motors. Following an introduction to principles and equations of the IM, the indirect FOC of the IM is presented and discussed. A current control method based on VSI is also presented.

Chapter three presents an adaptive PI based speed controller for the IM along with drive simulation results. In order to compare the adaptive PI based speed controller to the conventional PI controller, simulation results of an existing traditional PI controller-based IM drive are also presented.

Chapter four presents Adaptive PI Controller with Gain Scheduling, additionally, it is introduces Grid Search Method incorporated with PI controller.

Chapter five describes two simple designs of a Fuzzy Logic Controller (FLC), one using four rules and other one using nine. For comparison purposes, simulation results of a

traditional PI controller-based IM drive are also presented. Simulation results of the two designs are then discussed, and experimental implementation results are described.

Chapter six presents an ANN-based speed controller for the IM along with drive simulation results. For comparison purposes, simulation results of a traditional PI controller-based IM drive are also presented. Simulation results are then discussed, and experimental implementation results are described.

Chapter seven presents a novel Finite Element speed controller for the IM along with drive simulation results. For comparison purposes, simulation results of a traditional PI controller-based IM drive are also presented. Simulation results are then discussed, and experimental implementation results are described.

Chapter eight presents a comparison of the controllers based on rise time, settling time and rms error. The commands, disturbances and saturation levels were the same for each controller.

Chapter eight gives conclusions, contributions and recommendations for future work.

Chapter 2.

Introduction to Induction Machines

2.1 Principle of Operation

Induction machines have been considered the workhorse of modern industry. When three-phase balanced voltage has been applied to the stator windings of the machine, a rotating magnetic field will be established in the airgap. If the rotor is held stationary, a voltage will be induced in the rotor windings. The resulting driving force will have a constant magnitude, and it will rotate in the forward direction. The interaction of the stator rotating field and rotor induced field will create the driving force. The speed of the rotor will be lagging behind the stator synchronous field by a slip, *S*

The slip *S* is defined as

$$S = \frac{n_s - n_r}{n_s} \tag{2.1}$$

where, n_s is the synchronous speed of the stator rotating field and n_r is the rotor speed.

Synchronous speed is defined as

$$n_s = \frac{120\, f}{P} \tag{2.2}$$

where *f* is the frequency in hertz and *P* is the number of poles.

2.2 Classification of the Induction Machine

The general classification of the induction machine is given in figure 2-1

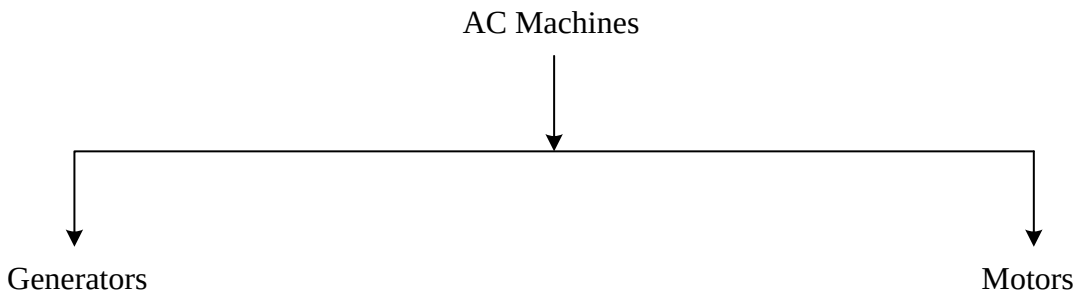


Figure 2-1: Classification of Induction Machine

In the case of a generator the slip is negative, while in the case of the induction motor (IM) the slip is positive.

2.3 Types of Induction Motors

- Wound - rotor type
- Squirrel - cage rotor type

The advantages of wound-rotor induction motors, when compared to squirrel-cage motors, are that they have access to rotor terminals, and therefore it is possible to insert additional resistance in the rotor circuit. Consequently, they have high starting torque and additionally they have a means of allowing variable speed by using external rotor resistance. The disadvantage is that it requires slip rings and a brush gear assembly. In

contrast, the advantages of the squirrel-cage induction motor are that it is a solid-rotor type, and it has many built-in features. There features are simplicity, ruggedness, reliability, low cost, and compactness [31-34]. The challenge of the induction machine is the method of control. The induction motor is a highly coupled and non-linear device, particularly when it's saturated. The dynamic models of the saturated induction motor and its applications in the area of speed control have been traditionally limited. The speed control of an IM through changing the number of poles has been known for a long time. However, this method is considered complex, expensive and therefore is not an efficient technique for high performance applications.

2.4 Dynamic Modelling of the Induction Motor

In this section, a dynamic model of the three-phase induction motor in synchronously rotating and stationary reference frames is described. This model is further used to develop reliable controllers for induction motor drives. The stator circuit equations can be modeled as follows [31-35] [40]:

$$v_{ds}^{s} = R_s i_{ds}^{s} + \frac{d}{dt}\psi_{ds}^{s} \tag{2.3}$$

$$v_{qs}^{s} = R_s i_{qs}^{s} + \frac{d}{dt}\psi_{qs}^{s} \tag{2.4}$$

where ψ_{ds}^{s} and ψ_{qs}^{s} are the d-axis and q-axis stator flux linkages, respectively. The equations (2.3) and (2.4) are further transposed into (d^e-q^e) frames as

$$v_{ds} = R_s i_{ds} + \frac{d}{dt}\psi_{ds} - \omega_e \psi_{qs} \tag{2.5}$$

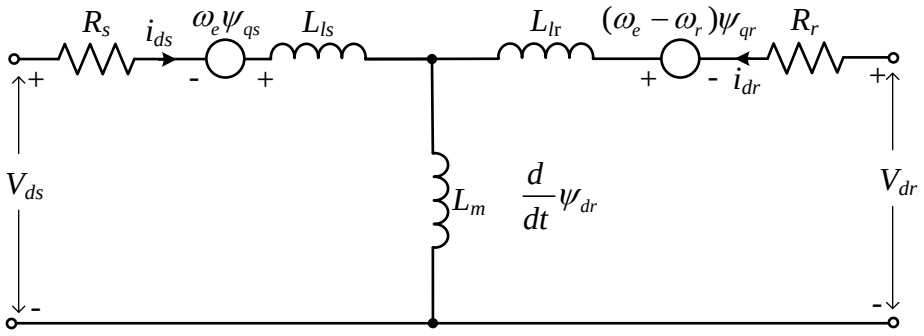
$$v_{qs} = R_s i_{qs} + \frac{d}{dt}\psi_{qs} + \omega_e \psi_{ds} \tag{2.6}$$

The superscript ‘e’ denotes the synchronous rotating frame parameters. It should be noted that when the synchronous speed is zero $\omega_e = 0$, the equations return to the stationary form. Because the rotor moves at speed $\omega_r = 0$, the $d-q$ axes are fixed on the rotor which rotates at a speed $\omega_e - \omega_r$, relative to the synchronously rotating frame. Therefore, in the $d^e - q^e$ frame, the rotor equations are modified:

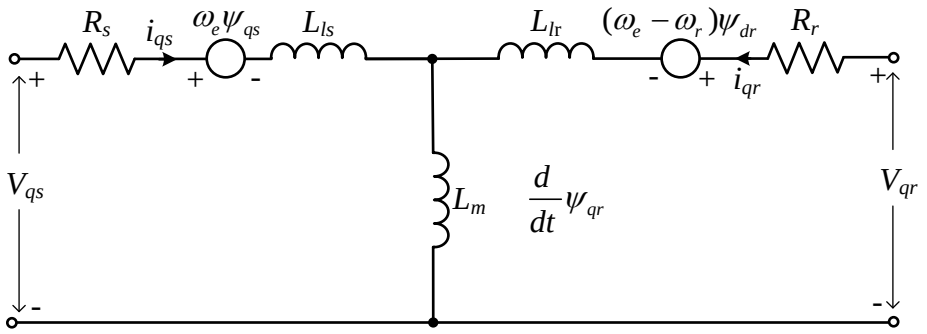
$$v_{dr} = R_r i_{dr} + p\psi_{dr} - (\omega_e - \omega_r)\psi_{qr} \tag{2.7}$$

$$v_{qr} = R_r i_{qr} + p\psi_{qr} + (\omega_e - \omega_r)\psi_{dr} \tag{2.8}$$

The flux linkages in terms of the currents can be written using an equivalent circuit shown in figure 2-2.



(a)



(b)

Figure 2-2: The dynamic equivalent circuit (d-q) of an induction machine, (a) direct axis circuit, (b) quadrature axis circuit

The associated equations characterizing this circuit are as follows:

$$\psi_{ds} = L_{ls} i_{ds} + L_m (i_{ds} + i_{dr}) \tag{2.9}$$

$$\psi_{qs} = L_{ls} i_{qs} + L_m (i_{qs} + i_{qr}) \tag{2.10}$$

$$\psi_s = \sqrt{(\psi_{ds})^2 + (\psi_{qs})^2} \tag{2.11}$$

$$\psi_{dr} = L_{lr} i_{dr} + L_m (i_{ds} + i_{dr}) \tag{2.12}$$

$$\psi_{qr} = L_{lr} i_{qr} + L_m (i_{qs} + i_{qr}) \tag{2.13}$$

$$\psi_r = \sqrt{(\psi_{dr})^2 + (\psi_{qr})^2} \tag{2.14}$$

Equations (2.15) and (2.16) define the stator and rotor self-inductances as the combination of leakage and magnetizing inductance terms. The magnetizing inductance is due to the effective air gap flux. The stator and rotor leakage inductances are caused by the stator and rotor leakage flux. This is leakage flux generated by the stator or rotor that fails to cross the air-gap.

$$L_s = L_{ls} + L_m \tag{2.15}$$

$$L_r = L_{lr} + L_m \tag{2.16}$$

Combining equations (2.9)-(2.16) and substituting into equations (2.5)-(2.8), the transient model in terms of v,i for a three-phase wye-connected induction machine is given in matrix form as [33] [36-39].

$$\begin{bmatrix} v_{ds}^e \\ v_{qs}^e \\ v_{dr}^e \\ v_{qr}^e \end{bmatrix} = \begin{bmatrix} -\omega_e L_s & R_s + pL_s & -\omega_e L_m & pL_m \\ R_s + pL_s & \omega_e L_s & pL_m & \omega_e L_m \\ -(\omega_e - \omega_r)L_m & pL_m & (\omega_e - \omega_r)L_r & R_r + pL_r \\ pL_m & (\omega_e - \omega_r)L_m & R_r + pL_r & (\omega_e - \omega_r)L_r \end{bmatrix} \begin{bmatrix} i_{ds}^e \\ i_{qs}^e \\ i_{dr}^e \\ i_{qr}^e \end{bmatrix} \tag{2.17}$$

$$\begin{bmatrix} v_{ds}^e \\ v_{qs}^e \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} -\omega_e L_s & R_s + pL_s & -\omega_e L_m & sL_m \\ R_s + pL_s & \omega_e L_s & sL_m & \omega_e L_m \\ -(\omega_e - \omega_r)L_m & sL_m & (\omega_e - \omega_r)L_r & R_r + pL_r \\ sL_m & (\omega_e - \omega_r)L_m & R_r + pL_r & (\omega_e - \omega_r)L_r \end{bmatrix} \begin{bmatrix} i_{ds}^e \\ i_{qs}^e \\ i_{dr}^e \\ i_{qr}^e \end{bmatrix} \quad (2.18)$$

During steady-state, all s-related terms in the above matrix become zero, and all variables in the synchronously rotating reference frame appear as DC quantities with sinusoidal excitation. Also note that for a squirrel cage induction motor $v_{qr} = 0$, $v_{dr} = 0$ as indicated in equation (2.18).

The dynamic model in a stationary frame can be derived simply by substituting $\omega_e = 0$ in equations (2.17) - (2.18).

Electrical rotor speed can be related to torque by the following expression

$$T_e = T_L + J\frac{d\omega_r}{dt} + B\omega_r \quad (2.19)$$

where T_L is the load torque, J is the moment of inertia, and B is friction.

Equations (2.20) and (2.21) define torque expressions produced by the induction motor. Equation (2.20) defines the torque as the cross product of rotor flux and stator current, (2.21) defines the torque as the cross product of stator flux and stator current.

$$T_e = \frac{3}{2}\left(\frac{P}{2}\right)\frac{L_m}{L_r}(\psi_{dr} i_{qs} - \psi_{qr} i_{ds}) \quad (2.20)$$

$$T_e = \frac{3}{2}\left(\frac{P}{2}\right)(\psi_{ds} i_{qs} - \psi_{qs} i_{ds}) \tag{2.21}$$

The equations detailed in this section describe the complete model of the electro-mechanical dynamics of an induction motor in a synchronously rotating frame.

2.5 Transformation of Induction Machine Quantities

Current, voltage and flux are usually described as three sinusoidal waves representing the three AC phases (a, b and c). Calculations in the abc phasor form are complex due to the sinusoidal phases and their variability in the time-frame.

That is why the a-b-c phases can be transformed into basic d-q-o form.

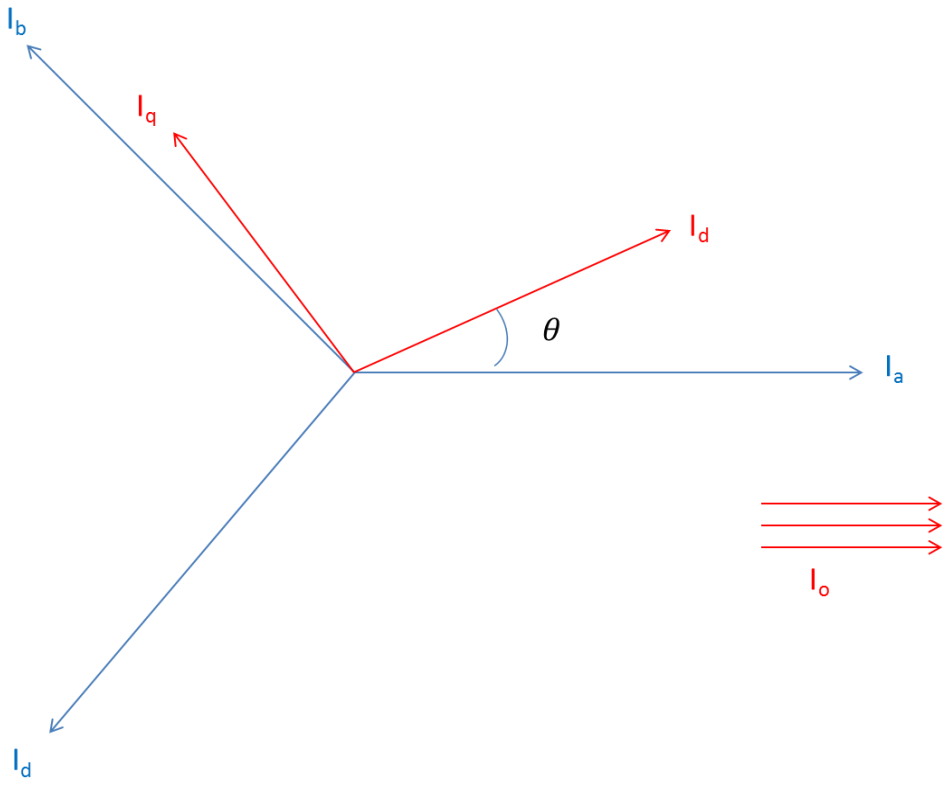


Figure 2-3: abc-dqo axis transformation

Figure 2.3 shows the basic abc-dqo axis transformation, where d is the direct axis, q is the quadrature axis and o is the neutral axis quantities, respectively.

Therefore, in order to reduce this complexity and the coupling effect, a change of variables is often required. It consists in transferring the IM equations to a quadrature rotating reference frame such that the mutual inductance is no longer time dependent. There are several methods for doing this. In this thesis, the well-known Park and Clarke transformations are used, modelled and implemented digitally. Using these transformations, many properties of an IM can be analyzed without complexities in the voltage, current and flux equations. Furthermore, Park and Clarke transformations make it possible and easy for control algorithms to be implemented on real-time DSPs [33] [40-44]. The following section illustrates how these transformations are performed for an IM.

2.5.1 Clarke Transformation

The Clarke transformation, developed by E. Clarke [33] [40], transforms stationary circuits to a stationary reference frame by converting balanced three phase quantities into balanced two phase orthogonal quantities in a stationary reference frame represented by α, β . Figure 2-4 demonstrate Clarke transformation

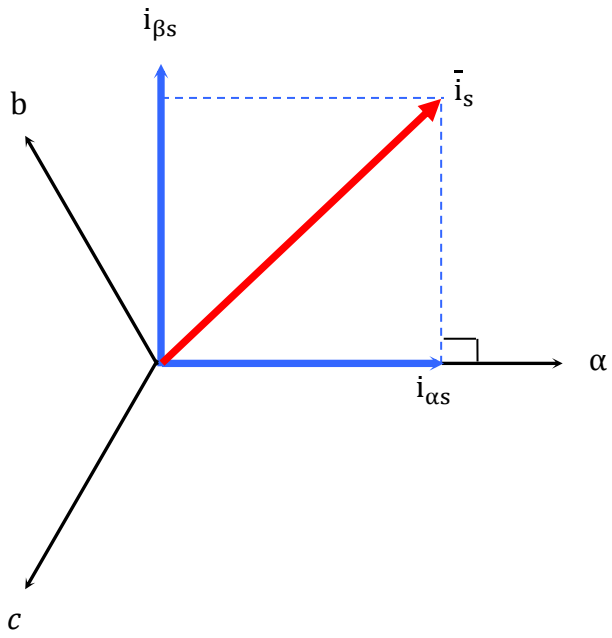


Figure 2-4: Clarke Transformation

$$\begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} = \frac{2}{3}\begin{bmatrix} 1 & -1/2 & -1/2 \\ 0 & \sqrt{3}/2 & -\sqrt{3}/2 \end{bmatrix}\begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} \tag{2.22}$$

2.5.2 Inverse Clarke Transformation

A two-axis orthogonal stationary reference frame can be transformed to a three-phase stationary reference frame using the inverse Clarke transformation. The inverse Clarke transformation is expressed by the following equations

$$\begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{1}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix} \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} \tag{2.23}$$

2.5.3 Park Transformation

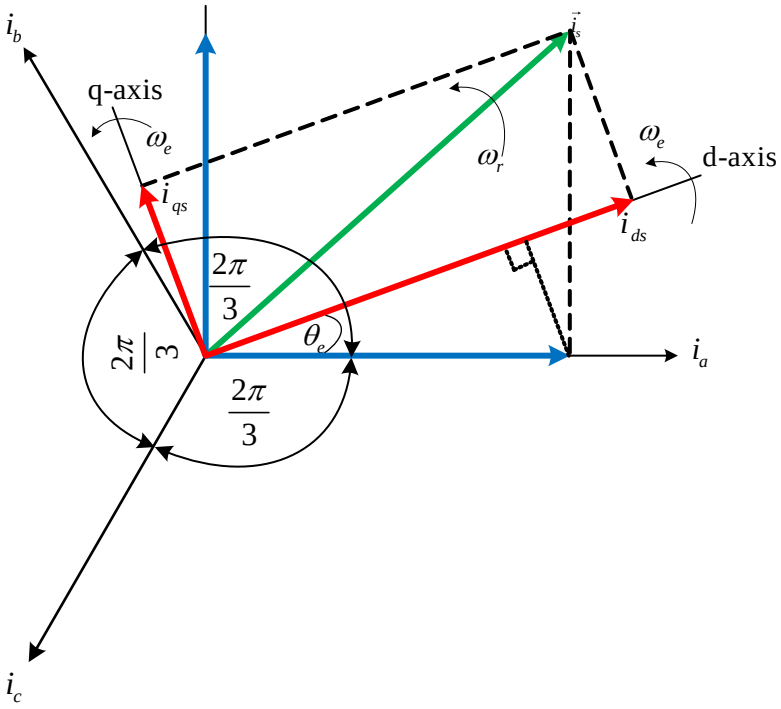


Figure 2-5: Park Transformation

Park's transformation converts balanced three-phase quantities into two phase quantities in synchronous machine analysis [33] [40-45].

Under balanced conditions, the current zero sequence component $i_{0,}$ is zero, and therefore it produces no flux at all. Under these conditions, we may write the d-q transformation as

$$\begin{pmatrix} I_a \\ I_b \\ I_c \end{pmatrix} = \sqrt{\frac{2}{3}} \begin{pmatrix} \cos(\theta_e) & \cos(\theta_e - 2\pi/3) & \cos(\theta_e + 2\pi/3) \\ -\sin(\theta_e) & -\sin(\theta_e - 2\pi/3) & -\sin(\theta_e + 2\pi/3) \\ 1/\sqrt{2} & 1/\sqrt{2} & 1/\sqrt{2} \end{pmatrix} \begin{pmatrix} I_d \\ I_q \\ I_0 \end{pmatrix} \qquad (2.24)$$

The Park transformation transforms the 2-axis orthogonal system quantities given in the stationary reference frame (α, β), into the rotating reference frame (d, q), as shown in figure 2-5. For the current vector, the relationship is expressed by the following equations:

$$\begin{aligned} i_{ds} &= i_{\alpha s}\cos\theta_e + i_{\beta s}\sin\theta_e \\ i_{qs} &= -i_{\alpha s}\sin\theta_e + i_{\beta s}\cos\theta_e \end{aligned} \qquad (2.25)$$

where θ_e is the rotation angle, i_{ds}, i_{qs} are rotating reference frame quantities, and $i_{\alpha s}$, $i_{\beta s}$ are orthogonal stationary reference frame quantities.

2.5.4 Inverse Park Transformation

The quantities in the rotating reference frame are transformed to the two-axis orthogonal stationary reference frame using the inverse Park transformation. The inverse Park transformation is given by the following equations:

$$\begin{aligned} i_{\alpha s} &= i_{ds}\cos\theta_e - i_{qs}\sin\theta_e \\ i_{\beta s} &= i_{ds}\sin\theta_e + i_{qs}\cos\theta_e \end{aligned} \tag{2.26}$$

2.6 Hysteresis Current Controller

For the hysteresis controller, when the actual phase currents are allowed to vary within the hysteresis band, this results in variations of the switching frequency of the inverter over the fundamental period. This is due to the sinusoidal nature of the command phase current which causes an irregular operation of the inverter with respect to time and an increase in switching losses which corresponds to some other types of current controllers. Such controllers remain in use due to their excellent dynamic response and simplicity.

Hysteresis controllers are divided into two categories, fixed-band and sinusoidal-band. For the sinusoidal-band controllers, the hysteresis band variation is sinusoidal over the fundamental period; as a result, the current has low harmonic content. However, this scheme produces very high switching frequency of the inverter close to zero crossings. On the other hand, in the fixed-band hysteresis controller, due to the switching frequency of the inverter being minimized, the harmonic content of the current is increased.

Considering that it is desired to maintain switching frequency of the inverter to a minimum, allowing the use of IGBT-based inverters producing accurate dynamic command current tracking, the fixed-band hysteresis controller has been selected for use in this work.

The operating principle of the fixed-band hysteresis controller is illustrated in figure 2-6. In the figure, N_1, N_3 and N_5 are the logic signals for the upper transistors of the inverter and N_4, N_6 and N_2 are the logic signals for the corresponding lower transistors of the inverter. Transistor T_1 is on when the logic signal N_1 is 1, and it is off when the logic signal N_1 is 0. Similarly, the other transistors follow logic signals.

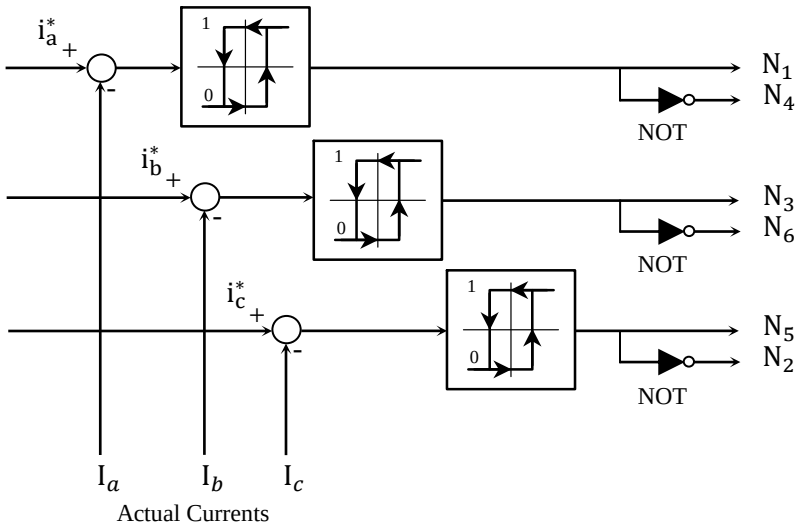


Figure 2-6: Fixed-band hysteresis current controller schemes

The operational logic of the fixed-band hysteresis controller is illustrated in figure 2-7. It can be described as follows:

$i_a^* > 0$, $N_4 = 0$: if $i_a > i_{up}$, then $N_1 = 0$, else if $i_a < i_{lo}$, then $N_1 = 1$.

$i_a^* < 0$, $N_1 = 0$: if $i_a > i_{up}$, then $N_4 = 1$, else if $i_a < i_{lo}$, then $N_4 = 0$.

where i_a^* is command current, i_a is actual phase 'a' current, $i_{up} = i_a^* + H$ is the upper band, $i_{lo} = i_a^* - H$ is the lower band, and H is the fixed hysteresis band.

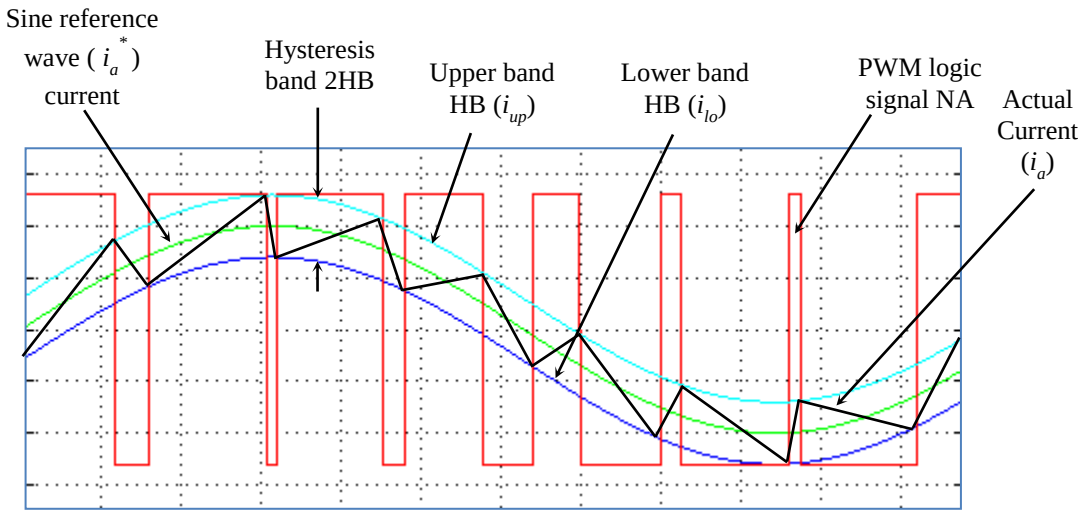


Figure 2-7: Hysteresis current controller waveforms

For example, in the case where the actual pulse current reaches the lower limit, the upper transistor of the half bridge for that particular phase is turned 'ON' and the lower transistor is turned 'OFF' (for instance, T_1 would be turned 'ON' and T_4 turned 'OFF' in

the case of phase 'a'). This switching action causes the phase current to increase positively in that phase. Consequently, when the phase current reaches the upper limit of the hysteresis band, the upper transistor is turned 'OFF', as the lower one also remains 'OFF'. As a result of its inductance, the current in the machine winding cannot go to zero instantaneously; therefore, as the upper transistor turns 'OFF', the freewheeling diode across the lower transistor begins to conduct, changing phase 'a' voltage from $+V_{dc}/2$ to $-V_{dc}/2$. The phase 'a' current is therefore forced to decrease negatively. The same principle is used when $i_a^* < 0$, and control of the transistors operates with a similar logic.

Table 2.2 summarizes the switching logic of the VSI for phase 'a'.

Table 2-1: Switching logic of VSI for phase 'a'

$\mathbf{i_a^*}$	$\mathbf{i_a}$	$\mathbf{S_1}$	$\mathbf{S_4}$	$\mathbf{V_a}$
≥0	$i_a \leq (i_a^* - \Delta i)$	ON	OFF	$+V_{dc}/2$
≥0	$i_a \leq (i_a^* + \Delta i)$	OFF	OFF	$-V_{dc}/2$
<0	$i_a \geq (i_a^* + \Delta i)$	OFF	ON	$-V_{dc}/2$
<0	$i_a \geq (i_a^* - \Delta i)$	OFF	OFF	$+V_{dc}/2$

The operation of phases 'b' and 'c' would be similar. In this approach, the actual current wave is forced to track the command current wave within the hysteresis band.

2.7 Simulation of Induction Motor

The rapid advance of power electronics has allowed induction motors, the conventional solution for constant speed rotating power applications, to also be used efficiently in variable speed drive systems. In such cases, though, the motor must be fed by means of a static frequency inverter, rather than directly by the (sinusoidal) power line [46-53]. The utilization of squirrel cage induction motors with electronic inverters presents great advantages in costs and energy efficiency, compared with other industrial solutions for varying speed applications. However, the inverter affects the motor performance and might introduce disturbances into the main power line. Inverters using Pulse Width Modulation (PWM) switching use semiconductor devices to transform the DC power into controlled AC power. The PWM waveform is generated by comparing a reference signal and a carrier waveform. The PWM waveform controls the Insulated Gate Bipolar Transistor (IGBT) switches to generate the AC output. PWM switching is an efficient way to generate. However, all PWM methods inherently generate harmonics and noise originating in the high dv/dt and di/dt semiconductor switching transients. In order to reduce harmonics and switching noise, external filtering needs to be added [164] [40].

The increasing number of applications with induction motors fed by PWM inverters operating in variable speed duty thus requires a good understanding of the whole power system, as well as of the interactions among its parts with one another (power line – frequency inverter – induction motor – load).

The purpose of the study in this section is to compare the waveform characteristics of an induction motor model powered by a voltage inverter with one being powered directly by a three voltage supply [208V, 60Hz].

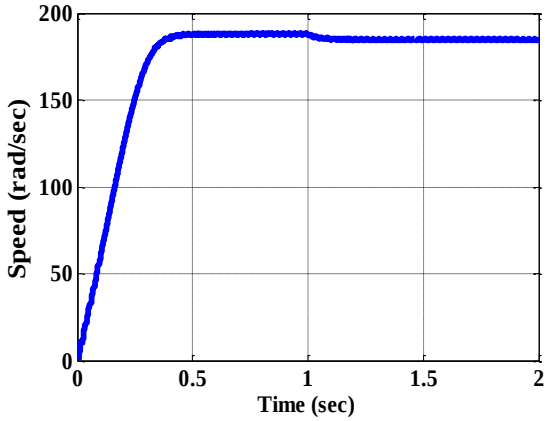


Figure 2-8: Simulated speed of the motor when it is powered by a voltage inverter

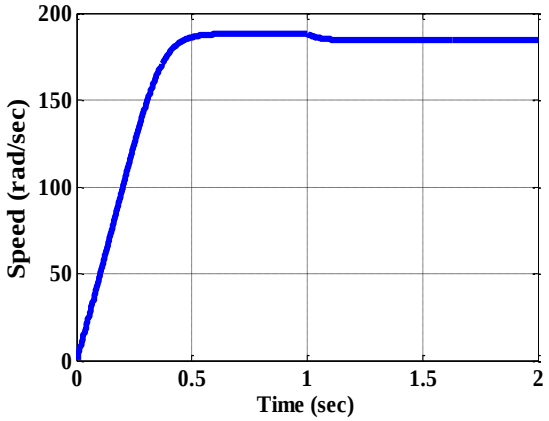


Figure 2-10: Simulated speed of the motor when it is directly powered by a voltage source supply

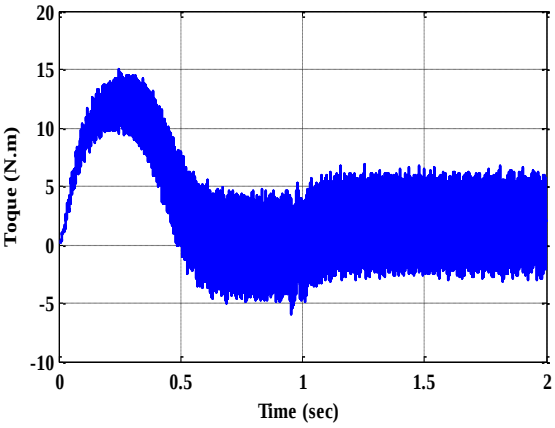


Figure 2-9: Simulated torque of the motor when it is powered by a voltage inverter

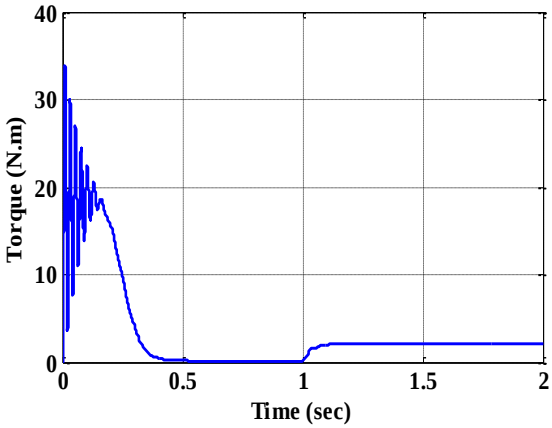


Figure 2-11: Simulated torque of the motor when it is directly powered by a voltage source supply

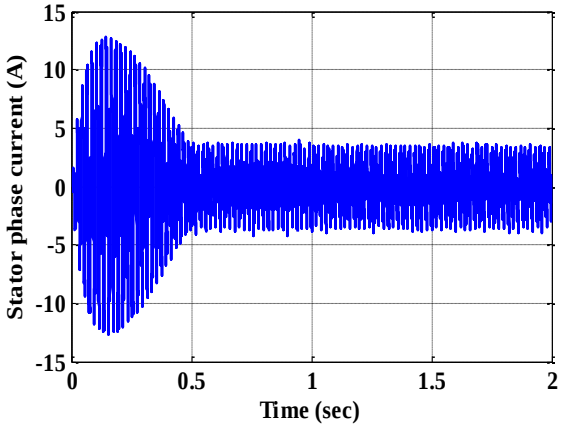


Figure 2-12: Simulated one phase current when the motor is powered by a voltage inverter

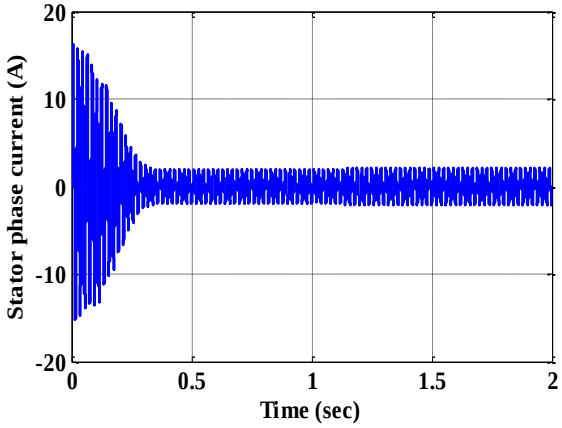


Figure 2-14: Simulated phase current when the motor is directly powered by a voltage source supply

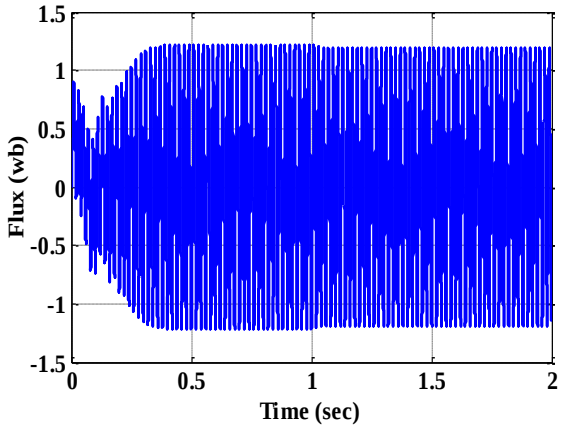


Figure 2-13: Simulated flux of the motor when it is powered by a voltage inverter

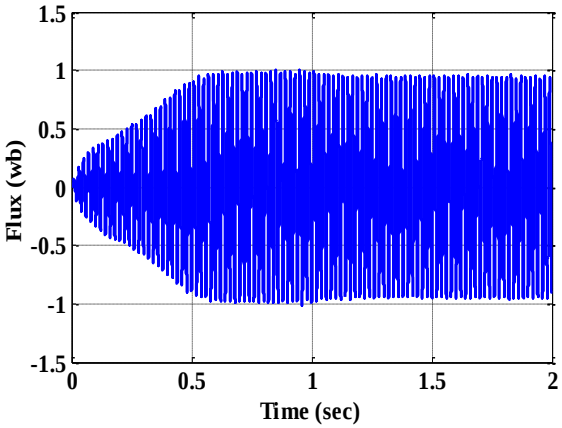


Figure 2-15: Simulated flux of the motor when it is directly powered by a voltage source supply

2.8 Discussion of Simulation Results

Initially, without load, the results obtained, as illustrated in figures 2-8, 2-10 show that the speed increases almost linearly and then reach a value close to the synchronous speed. These figures are both for actual motor speed obtained by simulations, one attained when the motor voltage is supplied though voltage inverter as illustrated in fig 2-8. While the other one is obtained when the motor is directly supplied by voltage supply as illustrate in fig. 2.10. Though their performance is very close, the motor which fed by voltage inverted has a low rise time than the other one.

During the transient state, the electromagnetic torque, as demonstrated in figures 2-9 and 2-11 exhibit oscillations, after which it stabilizes at a zero value (zero load). The current, as demonstrated in figures 2-12 and 2-14, shows successive oscillations at start-up. After the transient state these oscillations will decrease. Finally, the rotor flux, as illustrated in figures 2-13 and 2-15, is in sinusoidal form with almost constant amplitude.

When a load torque of (2 N.m at t = 1 sec.) is applied, the speed decreases, and the electromagnetic torque reaches its reference value to compensate for the oscillations with an almost instantaneous response before stabilizing to the nominal torque value. The rotor flux retains its shape with a slight decrease in its modulus and the stator current exhibits an increase in amplitude due to the increase in the load.

The figures of the two simulations of the machine supplied by the source supply and by the voltage inverter are almost identical to the oscillations which are distinguished in the figures of the machine fed by the inverter, due to the switching frequency of the switches.

2.9 Summary

This chapter has presented the development of the mathematical model of the induction motor using vector control, including Clark and Park transformations, through a PWM-controlled VSI utilizing a fixed-band hysteresis current controller. Further chapters will develop intelligent speed control methods for the IM drive to be used with the vector control scheme outlined here.

Chapter 3.

Speed Drives Control Methods for Induction Motors

3.1 Basic IM Drive Concepts

There are many methods of speed control of induction motors. Traditionally IMs were designed for constant-speed applications for the following reason. At constant supply voltage and frequency, based on the torque-speed characteristics of equation (3.1), the operating speed is very close to (less than 5% away from) the synchronous speed [54-55]

$$T_e \approx 3\left(\frac{T}{2}\right)\frac{R_r}{s\omega_s}\frac{V_s^2}{\left[\left(R_s+\frac{R_r}{s}\right)^2+\omega_s^2\left(L_{ls}+L_{lr}\right)^2\right]} \tag{3.1}$$

where T_e is the total developed electrical torque in a three phase motor , P is the number of poles, s is the slip. V_s is the stator supply voltage, R_s is the stator resistance per phase, R_r is the rotor resistance per phase, ωs is the synchronous speed in radian per second, L_{ls} is the stator leakage inductance and L_{lr} is the rotor leakage inductance.

If the load torque is increased, the speed drops by only a very small amount, making IMs very suitable for constant-drive systems. However, many industrial applications require variable speeds or a continuous variable range of speeds. With modern power electronics and Variable Frequency Drive (VFD) technologies it is possible to provide the necessary variable voltage and frequency that an IM requires for efficient and dynamic variable speed control. Modern power electronics, although more complex than those used for DC

drives, have not only made IMs suitable for many drive applications but also extended their applications and enabled users to take advantage of their low purchase and maintenance costs. The practical effect is the possibility for the IM to achieve a dynamic performance higher than that of a phase-controlled separately-excited DC drive. In order to understand how power electronics schemes are used to achieve such performances, it is important to analyse the fundamental concepts behind IM drives in general.

A careful analysis of equation (3.1) indicates that in general the speed and/or torque of an IM can be controlled by one of these methods, stator voltage, frequency, or voltage and frequency, depending on how the measured variables (current, voltage, and frequency) of the motor are manipulated in the controller.

3.2 Stator Voltage Control Operation

The torque equation as was demonstrated in (3.1) shows that the torque is proportional to the square of the supply voltage. Hence, a very simple method of controlling speed is to vary the supply voltage while maintaining constant supply frequency. This can be achieved through either a 3-phase autotransformer or a solid-state voltage controller.

The autotransformer method has the advantage of providing sinusoidal voltage for the IM, contrary to solid-state controllers. In large power applications, an input filter is required to reduce the harmonic currents flowing in the supply line if a solid-state controller is used. However, currently, solid-state approaches have become the most often used particularly with small squirrel-cage IMs. This is because they can be used as "Soft-Starters" for constant speed squirrel-cage IMs, where the starting voltage is applied

gradually to limit the stator inrush current. Variable frequency drives (VFD) adjust motor speed to match load fluctuations, reducing energy costs [51-60].

A solid-state voltage control consists of series-connected power switches (SRCs, GTOs, IGBTs, etc.) in the IM. The instant of applying voltage can be delayed by controlling the gating signals to the power switches. If the speed command is changed, the firing angles of the switches will change accordingly in order to generate a new voltage supply to the IM and thus a new operating speed.

Induction motors have to be driven by a Variable Frequency Drive (VFD) to be able to run at different speeds. Control methods for electric motors can be divided into two main categories, depending on what quantities they control. The scalar control algorithm controls only magnitudes, whereas the vector control algorithm controls both magnitude and angles. These two main methods can be further divided into a number of different methods depending on their functionality. An overview of different control methods can be seen in figure 3-1.[3] [61-62].

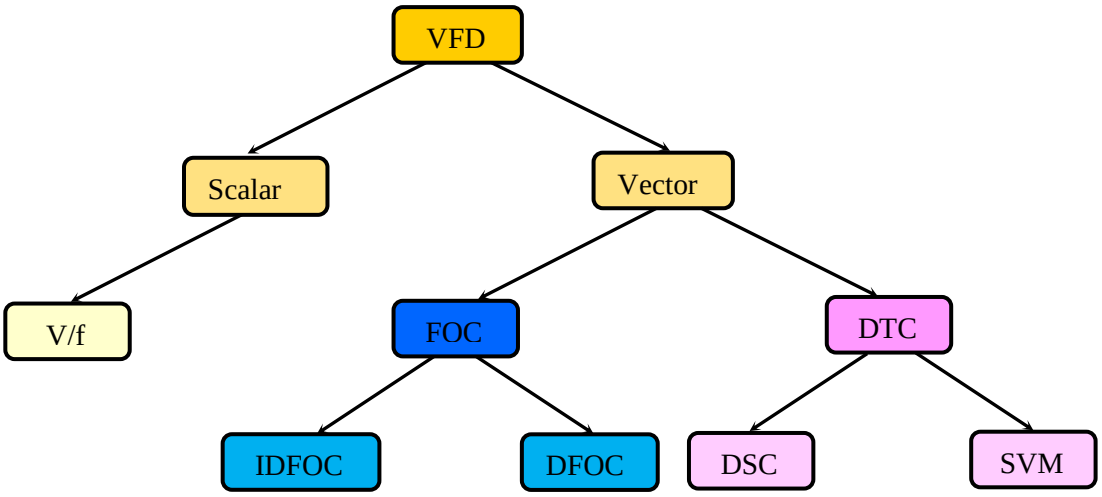


Figure 3-1: Overview of available control strategies

3.3 Scalar Control Methods

In order to overcome the limitations of voltage and frequency control methods, the scalar controller method is used to independently control the torque and speed; this is accomplished by varying the frequency and supply voltage to maintain constant air-gap flux. The key feature of this method is based upon an analysis of equation (3.1); thus, by maintaining constant air-gap flux at variable frequency, the stator voltage can be changed accordingly. This causes the control of IM drives to be more complex and sets them apart from DC drives, which require only voltage control.

A detailed study of these schemes is beyond the scope of this thesis, for further details refer to [51] [57] [63].

It is well-known that scalar methods provide good steady-state but poor dynamic responses, and that they only meet the requirements of industrial applications for which details of transient behaviours are not so important.

The poor dynamic responses obtained with scalar methods are due to the deviation of air-gap flux (in both phase and magnitude) caused by the inherent coupling effect of flux and torque. The coupling effect between the flux and torque in IMs makes their control system design very challenging. The foregoing problems can be solved by FOC techniques with real-time processors and an accurate IM model.

3.4 Vector Control Schemes

Traditional scalar control techniques have shortcomings due to the nonlinearity of the IM mathematical model and inherent coupling between direct and quadrature axis quantities. The consequences of this are inaccuracy and slow response, which are unacceptable in high performance drive applications [56-58]. In order to overcome this problem, direct torque control (DTC) or field-oriented control techniques, have been widely accepted for control of AC drives [59-62]. The vector control technique, which is based on employing a current controlled voltage source inverter (VSI), provides a method of variable speed control for the IM that has fast response and follows command speeds with precision and accuracy.

In the vector control technique for IM drives, the magnitude and phase angle of the phase currents are controlled to provide high precision control of the motor. This is an evolution of control techniques developed in the 1970s by Blaschke [162] and Hasse [163] for AC drives. However, implementation of these schemes was difficult due to technological limitations at that time. Currently, with the availability of advancing power electronics and microprocessor technologies, the practical implementation of the vector control scheme is no longer a problem.

The operating principle of vector control is based on elimination of the coupling between the direct (d) and quadrature (q) axes. This can be achieved by coordinate transformation, producing control very similar to that of a DC motor that is separately excited. Unlike DC machine control, both the magnitude and phase angle of the stator current need to be controlled in an AC machine. This is achieved by deploying a time-

varying vector that matches a sinusoidal flux wave moving in the air-gap of the machine, which explains the name “vector” control.

Vector control or the Field Oriented Control (FOC) of an induction machine is generally implemented to achieve a high-performance drive control. This is characterized by smooth rotation of the motor over the entire speed range. In vector control of induction motor drives, the three phase stator currents are decomposed into two components, the torque producing component and the flux producing component [62-64]. These two current components are orthogonal to each other. This allows independent control of flux and torque by the two orthogonal current components so that the operation becomes similar to that of a self-excited DC generator. To decompose the stator current, the position of the rotor flux must be known. There are two methods to obtain the rotor flux:

- Direct Field Oriented Control (DFOC): the rotor flux is either directly measured using sensing coils or estimated from the terminal measurements. Because it is not possible to directly sense the rotor flux, it is necessary to employ some type of computation to obtain the desired information from a directly sensed signal. However, it is not convenient to install a flux sensor inside the motor, and it is not practical for most industrial applications. Also, it will definitely decrease the reliability of the drive system [33] [51] [59]. A preferable approach is to use the flux observer to estimate or calculate the rotor flux to implement the field orientation control.
- IFOC (Indirect Field Oriented Control): the rotor flux position is calculated from the reference values and mechanical speed.

Indirect Field Oriented Control has been used in this research and described in detail in this chapter.

3.5 Derivations of the Field Oriented Control (FOC):

The basic principle of field oriented control is that if the d-axis of the arbitrary reference frame is aligned to any of the available flux vectors (ψ_s, ψ_r), the situation will become similar to that of separately excited DC motors. For the purpose of the field oriented control, the synchronously rotating reference frame has to be considered. Figure 3-2 illustrates the fundamental principle of vector control with the help of the phasor diagram. The stationary frame ($d^s - q^s$) axes are fixed on the stator. The ($d^r - q^r$) axes are fixed on the rotor and are rotating at the speed of the rotor (ω_r). The synchronously rotating reference frame ($d^e - q^e$) is rotating at synchronous speed (ω_e) and is ahead of the ($d^r - q^r$) axis by a positive slip angle θ_{sl} corresponding to slip frequency ω_{sl}. Since the rotor pole is directed along the d^e axis and ($\omega_e = \omega_r + \omega_{sl}$), it can be expressed as

$$\theta_e = \int \omega_e dt = \int (\omega_r + \omega_{sl}) dt = \theta_r + \theta_{sl} \tag{3.2}$$

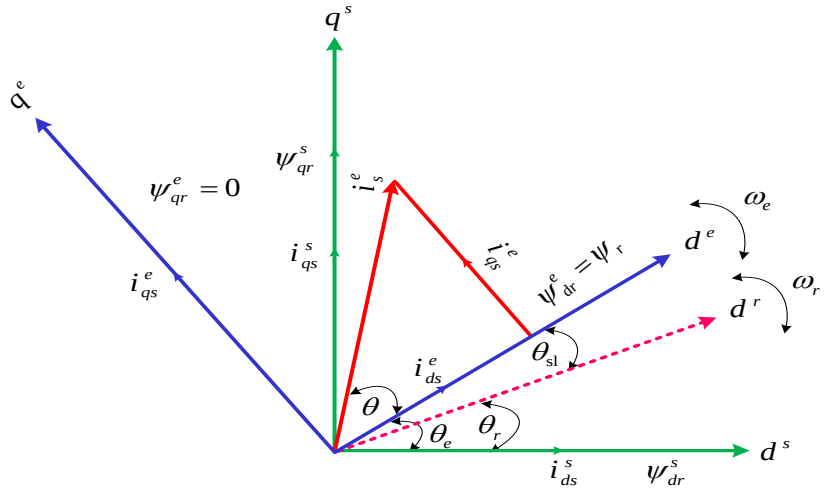


Figure 3-2: Principle of field oriented control

For the purpose of decoupling control, the mathematical modelling of the induction motor in the synchronously rotating reference frame is necessary. The corresponding stator and rotor voltage equations are given as [33]:

$$V^e_{ds} = R_s i^e_{ds} + \frac{d}{dt}\psi^e_{ds} - \omega_e \psi^e_{qs} \tag{3.3}$$

$$V^e_{qs} = R_s i^e_{qs} + \frac{d}{dt}\psi^e_{qs} + \omega_e \psi^e_{ds} \tag{3.4}$$

$$V^e_{dr} = R_r i^e_{dr} + \frac{d}{dt}\psi^e_{dr} - (\omega_e - \omega_r)\psi^e_{qr} \tag{3.5}$$

$$V^e_{qr} = R_r i^e_{qr} + \frac{d}{dt}\psi^e_{qr} + (\omega_e - \omega_r)\psi^e_{dr} \tag{3.6}$$

The torque equation can be written as

$$T_e = \frac{3}{2}\frac{P}{2} L_m (i^e_{qs} i^e_{dr} - i^e_{ds} i^e_{qr}) \tag{3.7}$$

The rotor flux linkage equation can be given as

$$\psi_{dr}^{e} = L_r i_{dr}^{e} + L_m i_{ds}^{e} \tag{3.8}$$

$$\psi_{qr}^{e} = L_r i_{qr}^{e} + L_m i_{qs}^{e} \tag{3.9}$$

From equation (3.8) and (3.9), the following currents are defined

$$i_{dr}^{e} = \frac{\psi_{dr}^{e} - L_m i_{ds}^{e}}{L_r} \tag{3.10}$$

$$i_{qr}^{e} = \frac{\psi_{qr}^{e} - L_m i_{qs}^{e}}{L_r} \tag{3.11}$$

Putting the values of i_{dr}^{e} and i_{qr}^{e} in equation (3.7), the expression of torque obtained is given as

$$T_e = \frac{3}{2}\frac{P}{2}\frac{L_m}{L_r}(i_{qs}^{e}\psi_{dr}^{e} - i_{ds}^{e}\psi_{qr}^{e}) \tag{3.12}$$

For the rotor field oriented control, the d^e axis of the synchronously rotating reference frame is aligned with the rotor field and the q-component of the rotor flux will be zero. Thus $\psi_{qr}^{e} = 0$ and $\psi_{dr}^{e} = \psi_r$. The torque equation is obtained as

$$T_e = \frac{3}{2}\frac{P}{2}\frac{L_m}{L_r}(i_{qs}^{e}\psi_{dr}^{e}) \tag{3.13}$$

Considering the d-axis rotor voltage equation (3.5),

$$V_{dr}^{e} = R_r i_{dr}^{e} + \frac{d}{dt}\psi_{dr}^{e} - \omega_{sl}\psi_{qr}^{e} \tag{3.14}$$

where $\omega_{sl} = (\omega_e - \omega_r)$

For the squirrel cage induction motor, the rotor bars are short circuited. Hence, the voltage across the d-axis rotor and the q-axis rotor is zero, i.e., $V_{dr}^{e} = V_{qr}^{e} = 0$. The q-axis component of the rotor flux is 0, i.e., $\psi_{qr}^{e} = 0$. Equation (3.5) can be written as

$$R_r i_{dr}^{e} + \frac{d}{dt}\psi_{dr}^{e} = 0 \tag{3.15}$$

Substituting $i_{dr}^{e} = \frac{\psi_{dr}^{e} - L_m i_{ds}^{e}}{L_r}$ in equation (3.15) and inserting the value of i_{dr}^{e} in equation (3.13), yield the following equations

$$R_r\left(\frac{\psi_{dr}^{e} - L_m i_{ds}^{e}}{L_r}\right) + \frac{d}{dt}\psi_{dr}^{e} = 0 \tag{3.16}$$

$$\frac{d}{dt}\psi_{dr}^{e} + \frac{1}{\tau_r}\psi_{dr}^{e} - \frac{L_m}{\tau_r} i_{ds}^{e} = 0 \tag{3.17}$$

where $\tau_r = \frac{L_r}{R_r} = \frac{L_m + L_{lr}}{R_r}$ is the rotor flux time constant.

Taking the Laplace transformation of equation (3.15), results in equation (3.18)

$$\psi_{dr}^{e} = \frac{L_m i_{ds}^{e}}{1 + \tau_r s} \tag{3.18}$$

Taking the other rotor voltage equation (3.6),

$$V_{qr}^{e} = R_r i_{qr}^{e} + \frac{d}{dt}\psi_{qr}^{e} + \omega_{sl}\psi_{dr}^{e} \tag{3.19}$$

For the squirrel cage induction motor, the rotor bars are short circuited. Hence the voltage across the d-axis rotor and the q-axis rotor is zero, i.e., $V_{dr}^{e} = V_{qr}^{e} = 0$. Further $\psi_{qr}^{e} = 0$ and $i_{qr}^{e} = \frac{\psi_{qr}^{e} - L_m i_{qs}^{e}}{L_r}$. Inserting the values of $V_{dr}^{e}, \psi_{qr}^{e}$ and i_{qr}^{e} in equation (3.17), the following equations are derived:

$$R_r \left(\frac{0 - L_m i_{qs}^{e}}{L_r} \right) = -\omega_{sl} \psi_{dr}^{e} \tag{3.20}$$

$$\omega_{sl} = \left(\frac{L_m}{\tau_r} \right) \frac{i_{qs}^{e}}{\psi_{dr}^{e}} \tag{3.21}$$

From equation (3.18), it is observed that the time constant (τ_r) of the rotor flux is very slow compared to the current. Therefore, the steady state value of the rotor flux ψ_{dr}^{e} can be obtained as

$$\psi_{dr}^{e} = L_m i_{ds}^{e} \tag{3.22}$$

Putting equation (3.22) in equation (3.21), the slip frequency is obtained as:

$$\omega_{sl} = \left(\frac{1}{\tau_r} \right) \frac{i_{qs}^{e}}{i_{ds}^{e}} \tag{3.23}$$

From equation (3.11), it can be concluded that the electromagnetic torque can be independently controlled by the q-axis component of the stator current, if the rotor flux is considered constant (i.e., at steady state). Similarly, the rotor flux can be independently controlled by the d-axis component of the stator current, at steady state. Thus, independent control of flux and torque by the two orthogonal current components has been achieved by this indirect field oriented control technique.

Given a desired value of the rotor flux $\psi_{dr}^{e\,*}$, the reference d-axis current component can be obtained as

$$\psi_{dr}^{e\,*} = \frac{L_m}{1+\tau_r s} i_{ds}^{e} \tag{3.24}$$

The desired value of the electromagnetic torque T_e^* is obtained from the PI controller output of the outer speed control loop, at a certain value of rotor flux. The reference value of the q-axis current component can be obtained as

$$T_e^* = \frac{3}{2}\frac{P}{2}\frac{L_m}{L_r}(i_{qs}^{e\,*}\psi_{dr}^{e\,*}) \tag{3.25}$$

Thus, the above analysis shows that the vector control strategy can provide the same performance as is achieved from a separately excited DC machine; this is achieved by formulating the stator current phasor, in the two-axis synchronously rotating reference frame, to have two components: flux producing current component and torque producing current component.

3.6 Indirect FOC IM Drive

Figure 3-3 shows the schematic diagram of the investigated IFOC IM drive. Two stator motor phase currents are measured. These measurements feed the Clarke transformation module. The outputs of this projection are designated as $i_{\alpha s}$ and $i_{\beta s}$. These two components of the current are the inputs of the Park transformation that provide the current in the d , q rotating reference frame. The induction motors require a rotor flux creation in order to operate. Thus, the flux reference must not be zero. The torque command which produces the current component i_{qs}^* could be the output of the

speed controller when a speed FOC is used. These reference current components i_{ds}^*, i_{qs}^* will be transformed to I_a^*, I_b^* and I_c^*, and then will be compared with the measured stator currents. The resulting errors are the inputs to the hysteresis bands. The outputs of this block are the signals that drive the inverter.

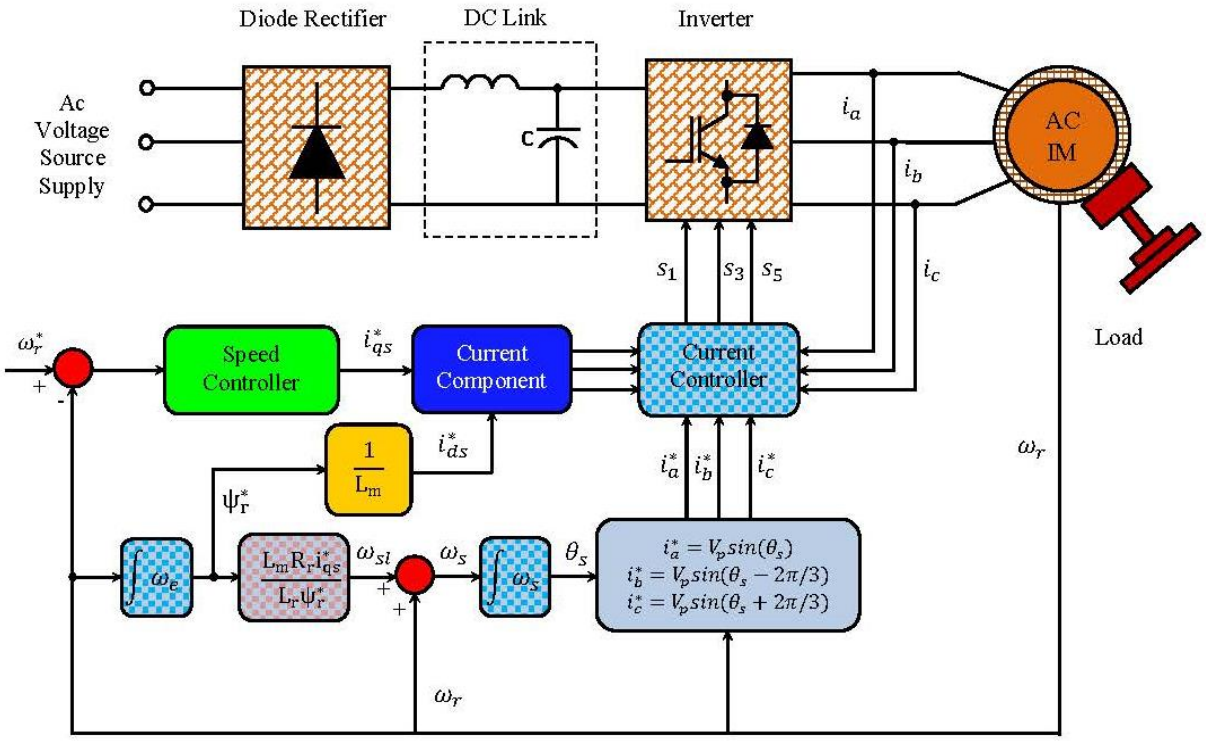


Figure 3-3: Configuration of the block diagram of indirect field oriented control scheme

3.7 Optimal PI Parameters

A PI controller is used to achieve speed control with the Indirect Field Oriented Control (IFOC) illustrated in figure 3-3.

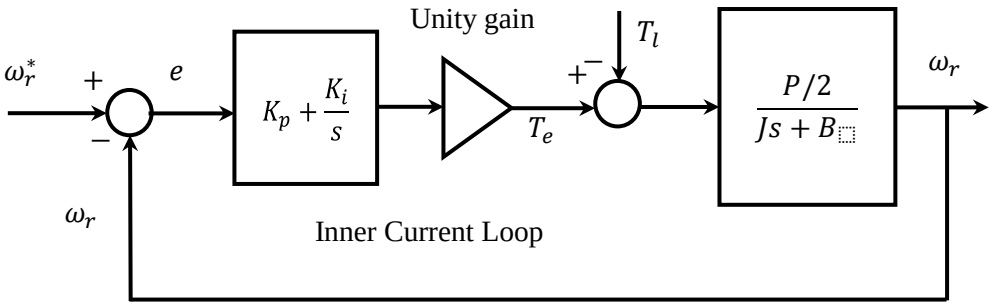


Figure 3-4: Block diagram of Speed Control Loop

The following mechanical equation governs the outer speed loop:

$$J\frac{d\omega_r}{dt} + B\omega_r = \frac{P}{2}\left(T_e - T_l\right) \tag{3.26}$$

where B and J denote the viscous friction coefficient and moment of inertia respectively, T_l is the external disturbance load, ω_r is the rotor mechanical speed in angular frequency, and P is the number of poles in the induction motor.

The electromagnetic torque can be expressed in a number of ways. Some of the commonly used expressions are

$$T_e = \begin{cases} \dfrac{3P}{2}(\psi_{ds} i_{qs} - \psi_{qs} i_{ds}) \\ \dfrac{3PL_m}{2}(i_{qs} i_{dr} - i_{ds} i_{qr}) \\ \dfrac{3PL_m}{2L_r}(\psi_{dr} i_{qs} - \psi_{qr} i_{ds}) \end{cases} \tag{3.27}$$

When the field-oriented control is applied by aligning the *d*-axis of the synchronous reference frame with the rotor flux vector $\vec{\psi}_r$ as illustrated in figure 3-2, the resultant *d*- and *q*-axis rotor flux components are $\psi_{qr} = 0,\ \psi_{dr} = \psi_r$

where ψ_r is the magnitude of $\vec{\psi}_r$ .Substituting rotor flux components into the equation of (3.25) yields

$$T_e = K_t \psi_{dr} i_{qs}^* = K_t \psi_r i_{qs}^* \tag{3.28}$$

the resulting mechanical equation can be described as

$$J\frac{\mathrm{d}\omega_\mathrm{r}}{\mathrm{dt}} + \mathrm{B}\omega_\mathrm{r} + \frac{\mathrm{P}}{2}T_l = \frac{\mathrm{P}}{2}K_t \psi_r i_{qs}^* \tag{3.29}$$

K_t is called the torque constant and is defined as

$$K_t = \frac{3\mathrm{P}}{2}\frac{\mathrm{L_m}}{\mathrm{L_r}} \tag{3.30}$$

The reference torque is the output of the speed controller. The reference torque is then transformed into a corresponding current component (i_{qs}^*) to be used by the inner current control loop. It should be noted, though, for the purpose of simplification, that the inner current control loop can be treated as a unity gain, while the outer speed loop is being designed as shown in figure 3-4. This is because the mechanical time constant is

very high compared to the electrical time constant [33] [40]. The current and rotor flux are stated as

$$i_{qs}^{*} = \frac{2L_r T_e}{3PL_m \psi_r^{*}} \tag{3.31}$$

$$\psi_r^{*} = L_m i_{ds}^{*} \tag{3.32}$$

The closed-loop transfer function with respect to the reference input is

$$\frac{\omega_r}{\omega_r^{*}} = \frac{K_i K_t}{Js^2 + \left(B + K_p K_t\right)s + K_i K_t} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{3.33}$$

where $\zeta = \dfrac{B + K_p K_t}{2\sqrt{JK_i K_t}}$ and $\omega_n = \sqrt{\dfrac{K_i K_t}{J}}$

Then, for a unit step input, the response is

$$\omega_r = \omega_r^{*} \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} = \frac{1}{s}\frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} = \frac{1}{s}\frac{\omega_n^2}{\left(s + p_1\right)\left(s + p_2\right)} \tag{3.34}$$

In order to obtain a fast response without overshoot, the system should be critically damped, i.e. $\zeta = 1$ and $p_1 = p_2 = \omega_n$. Then, the above equation becomes

$$\omega_r = \frac{1}{s}\frac{\omega_n^2}{(s + \omega_n)^2} = \frac{1}{s} - \frac{\omega_n}{\left(s + \omega_n\right)^2} - \frac{1}{\left(s + \omega_n\right)} \tag{3.35}$$

$$\begin{aligned} 1 &= \frac{B + K_p K_t}{2\sqrt{JK_i K_t}} \\ \omega_n &= \sqrt{\frac{K_i K_t}{J}} \end{aligned} \tag{3.36}$$

The transient response of the system is given by

$$\omega_r\left(t\right) = 1 - \omega_n t e^{-\omega_n t} - e^{-\omega_n t} \tag{3.37}$$

In order to attain the PI controller parameters which will affect the speed response, overshoot value, rise time, settling time, and load torque impact must be carefully adjusted according to the desired response. It must be noted that the stability and the dynamic performance of a closed loop system are directly related to the location of the closed-loop roots of the characteristic equation in the s-plane. However, the adjustment of PI gains cannot achieve all of these transient characteristics together. There is always a trade-off between them. Figure 3-2 shows the schematic diagram of the IM drive speed controller required to design a PI controller. The transfer function of the open loop system is described in equation (3.35) and has two poles at zero and $\frac{-B}{J}$, and one zero at $-\frac{K_i}{K_p}$:

$$G_{OL}\Big|_{T_l=0} = \frac{\mathrm{K_t K_p}\left(\mathrm{s}+\frac{\mathrm{K_i}}{\mathrm{K_p}}\right)}{s\left(\mathrm{Js}+\mathrm{B}\right)} \tag{3.38}$$

The root locus method for pole zero location displayed in figure 3-4 is used to design the optimal gains of the PI controller. The PI gain parameters can be calculated and implemented in the controller to determine the stability of the system, and achieve the desired response. The PI controller gains K_p , K_i are designed according to the desired performance characteristics of control systems, which are specified in terms of maximum overshoot M_p, $5\% \le M_p \le 10\%$, and settling time t_s, $0.5\ \mathrm{s} \le t_s \le 2$ sec. To obtain high dynamic response, it has been determined that the PI gains are $K_p = 3.2$, $K_i = 3.6$.

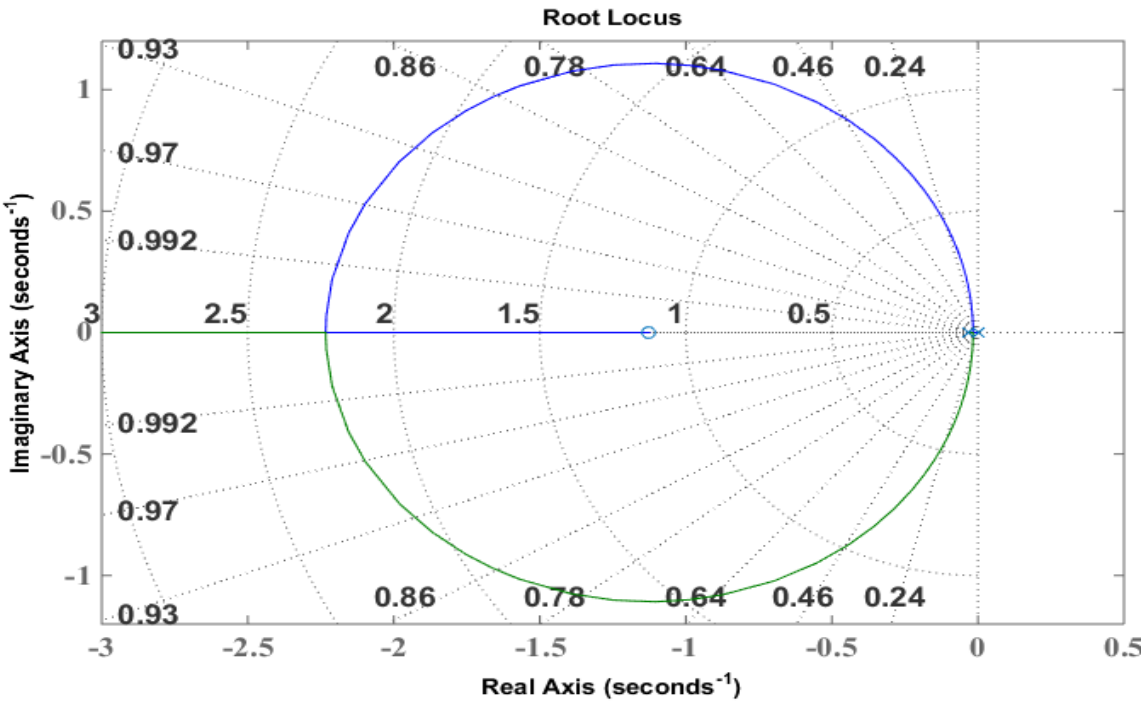


Figure 3-5: Root locus plot of the open loop transfer function with the PI

3.8 Design and Analysis of Proportional Integral Controller

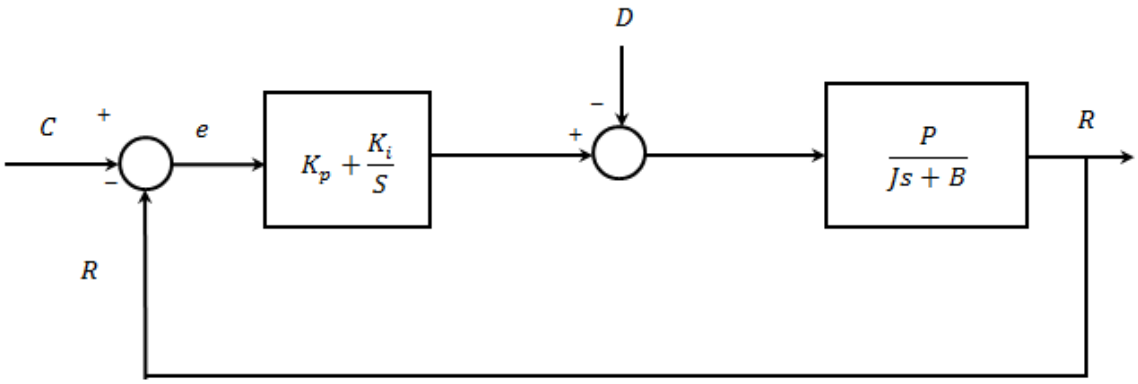


Figure 3-6: Block diagram of a feedback control system

For the command case, the transfer function G and feedback function H are given:

$$G = \left(\frac{K_p S + K_i}{S}\right)\left(\frac{P}{JS + B}\right) \tag{3.39}$$

$$H = 1 \tag{3.40}$$

The overall transfer function for the system is:

$$\frac{R}{C} = \frac{G}{1 + GH} \tag{3.41}$$

When the overall transfer function is applied, the characteristic transfer function for the whole system becomes:

$$= \frac{PK_p S + PK_i}{JS^2 + \left(B + PK_p\right)S + PK_i} \tag{3.42}$$

$$= \frac{\left(PK_p / J\right)S + PK_i / J}{S^2 + \left(B + PK_p / J\right)S + PK_i / J}$$

$$= \frac{\left(PK_p / J\right)S + PK_i / J}{\left(S - R_1\right)\left(S - R_2\right)}$$

where R_1, R_2 represents the characteristic roots

The characteristic equation is

$$JS^2 + \left(B + PK_p\right)S + PK_i = 0 \tag{3.43}$$

The output response for the unit step input

$$R = \frac{C_o}{S}\left(\frac{(PK_p / J)S + PK_i / J}{\left(S - R_1\right)\left(S - R_2\right)}\right) \tag{3.44}$$

This equation can be represented as a sum of simple fractions:

$$R = C_o\left(\frac{M_0}{S} + \frac{M_1}{\left(S - R_1\right)} + \frac{M_2}{\left(S - R_2\right)}\right) \tag{3.45}$$

Inverse Laplace Transformation gives

$$R = C_o\left(M_0 + M_1 e^{R_1 t} + M_2 e^{R_2 t}\right) \tag{3.46}$$

Partial Fraction Expansion gives

$$\frac{1}{S}\left[\frac{(PK_p/J)S+(PK_i/J)}{(S-R_1)(S-R_1)}\right]=\frac{M_0}{S}+\frac{M_1}{(S-R_1)}+\frac{M_2}{(S-R_2)} \tag{3.47}$$

$$\frac{M_0}{S}+\frac{M_1}{S-R_1}+\frac{M_2}{S-R_2}=\frac{1}{S}\left[\frac{(PK_p/J)S+(PK_i/J)}{(S-R_1)(S-R_1)}\right] \tag{3.48}$$

Partial fraction expansion is used to determine the unknown coefficients M_0, M_1, and M_2.

To find M_0, multiply both sides by s,

$$S\left(\frac{M_0}{S}\right)+S\left(\frac{M_1}{S-R_1}\right)+S\left(\frac{M_2}{S-R_2}\right)=\frac{S}{S}\left[\frac{(PK_p/J)S+(PK_i/J)}{(S-R_1)(S-R_1)}\right] \tag{3.49}$$

and then set $s=0$

$$M_0+S\left(\frac{M_1}{S-R_1}\right)+S\left(\frac{M_2}{S-R_2}\right)\Bigg|_{S=0}=\left(\frac{(PK_p/J)S+(PK_i/J)}{(S-R_1)(S-R_1)}\right)\Bigg|_{S=0} \tag{3.50}$$

$$M_0=\frac{PK_i}{JR_1R_2} \tag{3.51}$$

To find M1, multiply both sides by $S(S-R_1)$ and then set $S=R_1$

$$\begin{aligned}&S(S-R_1)\left(\frac{M_0}{S}\right)+S(S-R_1)\left(\frac{M_1}{S-R_1}\right)+S(S-R_1)\left(\frac{M_2}{S-R_2}\right)\Bigg|_{S=R_1}\\&=\frac{S(S-R_1)}{S}\left(\frac{(PK_p/J)S+(PK_i/J)}{(S-R_1)(S-R_2)}\right)\Bigg|_{S=R_1}\end{aligned} \tag{3.52}$$

$$M_1 = \frac{1}{R_1}\left(\frac{(PK_p/J)R_2 + (PK_i/J)}{(R_1 - R_2)}\right) \tag{3.53}$$

Likewise, for M2 multiply both sides by $S(S-R_2)$ and then set $S = R_2$

$$\begin{aligned} &S(S-R_2)\left(\frac{M_0}{S}\right) + S(S-R_2)\left(\frac{M_1}{S-R_1}\right) + S(S-R_2)\left(\frac{M_2}{S-R_2}\right)\Bigg|_{S=R_2} \\ &= \frac{S(S-R_2)}{S}\left(\frac{(PK_p/J)S + (PK_i/J)}{(S-R_1)(S-R_2)}\right)\Bigg|_{S=R_2} \end{aligned} \tag{3.54}$$

$$M_2 = \frac{1}{R_2}\left(\frac{(PK_p/J)R_1 + (PK_i/J)}{(R_2 - R_1)}\right) \tag{3.55}$$

For the disturbance case, the transfer function G is given:

$$G = \frac{P}{JS+B} \tag{3.56}$$

The overall transfer function is:

$$\frac{R}{D} = \frac{G}{1+GH} \tag{3.57}$$

The transfer function for the system is:

$$= \frac{(P/J)S}{JS^2 + \left((B + PK_p)/J\right)S + PK_i / J} \tag{3.58}$$

For a step disturbance

$$R = \frac{D_o}{S}\left(\frac{(P/J)S}{(S-R_1)(S-R_2)}\right) \tag{3.59}$$

This equation can be represented as a sum of simple fractions:

$$R = D_o\left(\frac{N_0}{S} + \frac{N_1}{(S-R_1)} + \frac{N_2}{(S-R_2)}\right) \tag{3.60}$$

Inverse Laplace Transformation gives

$$R = D_o\left(N_0 + N_1 e^{R_1 t} + N_2 e^{R_2 t}\right) \tag{3.61}$$

Partial Fraction Expansion gives

$$\frac{1}{S}\left[\frac{(P/J)S}{(S-R_1)(S-R_1)}\right] = \frac{N_0}{S} + \frac{N_1}{(S-R_1)} + \frac{N_2}{(S-R_2)} \tag{3.62}$$

$$N_0 = 0 \tag{3.63}$$

$$N_1 = \frac{P/J}{R_1 - R_2} \tag{3.64}$$

$$N_2 = \frac{P/J}{R_2 - R_1} \tag{3.65}$$

Chapter 4.

Adaptive PI Controller with Gain Scheduling for Indirect Field Oriented Control of Induction Motor

It is well known that a conventional proportional-integral (PI) controller is one of the most widely used in industry due to its simple control structure, ease of design, and low cost. However, the PI controller cannot yield a good control performance. Moreover, it suffers from slow response, large overshoots, and oscillations [11] [19] [65-73]. Because of the widespread use of PI control, it is highly desirable to have efficient manual and automatic methods for tuning the controllers. A gain-scheduled scheme for a PI controller, as it is simple to implement and provides good performance, has been proposed. In this approach, the PI gains are designed to be allowed to vary within a pre-determined range. This, therefore, alleviates the problems encountered by the conventional proportional-integral controller.

In this chapter, a gain-scheduling adaptive PI controller for field-oriented control of an IM drive is designed to eliminate the inherent problems of the conventional PI controller. The gains of the proposed control scheme are tuned such that the drive system exhibits satisfactory transient and steady-state responses under varying operating conditions.

4.1 Introduction

Intelligent motion control is a prerequisite both in general and industrial applications. In electrical drives, many different sensors and control algorithms are applied to control the speed of motors by utilizing a wide array of speed control methods. Industrial drives emerged from power electronics, with adjustable and variable speed drives offering significantly smoother operation and acceleration control, along with variable operating speeds and torque control. Induction motors (IMs) are gradually replacing DC motors due to improvements such as reliability, cost-effectiveness, and low maintenance requirements. However, controlling IMs is significantly more complicated than controlling DC motors, as induction motors are inherently nonlinear.

Of the solutions developed thus far to deal with these inherent challenges, the field-oriented control approach is the most promising. By splitting the stator currents into two orthogonal components, one in the direction of the flux linkage, representing d-axis current component, and the other perpendicular to the flux linkage, producing torque, where both components can be varied independently, the induction motor becomes a simple DC motor, under the assumption that the actual machine flux is kept constant and equal to some desired value. This is the principle of field-oriented control [33].

Over the years, several control algorithms have evolved for achieving high performance vector control of the IM. Among these, many are based on linear control techniques like proportional-integral or proportional integral-derivative compensation. However, the IM has a nonlinear model and exhibits coupled dynamics. As a result, the linear PI controller is inadequate in applications where good transient performance under all motor operating conditions is

desired. This has prompted the application of nonlinear control techniques for IM speed control such as model adaptive control systems [20] [74-77], sliding mode control [78-84], back-stepping control [83-85], etc.

Sliding Mode Controller is a nonlinear discontinuous robust control that alters the dynamics of a nonlinear system by applying a variable structure control signal.
This control signal, forces the system to reach, and then consequently remain on, a predefined surface (called the sliding surface). The dynamical performance of the system, when restricted to the surface is called sliding mode.
The advantage of using SMC is model parameters need not to be accurately known, only their bounds is needed to be known and, furthermore, the SMC controlled system is insensitive to the parameter variation of the system hence it is highly robust
Design involves two steps:
1) Selection of stable hyper plane (s) in the state/error space on which the motion should be restricted, called the Sliding Surface.
2) Synthesis of a control law which makes any arbitrary initial condition attracted towards Sliding Surface and remains thereon.
Sliding Mode Controller (SMC) is applicable to non-liner systems like IM. This provides a quick response, on account of the input changes, robustness against parameter variations, as well as the capability to recover from load disturbance. On the other hand, a limitation of this controller is the chattering caused when the signal function is used as a switching function. This problem is represented by high frequency oscillations which could cause the system to be unstable [79-80]. In the past years some other techniques, along with SMC, are including adaptive control and intelligent systems. Sometimes the use of these

techniques can increase the controller complexity [81]. As a means of decreasing the chattering effect, some works report using a low pass filter in the SMC output. A drawback to this method, however, is that it will result in sluggish torque response and a system exposed to inferior performance during the process of chattering reduction [81-84].

Although several nonlinear control techniques have been proposed and successfully implemented in research laboratories for speed/position control of electric motor drives, most of them are yet to be accepted by industry. It is beyond the scope of the thesis project to compare with all other methods.

Currently, the majority of the speed/position controllers in use by industry for electric motor drives are linear PI types [65-73] [86]. The popularity of these controllers is primarily due to their simplicity. Hence, given the clear preference of industry for a simple solution that is also easy to implement, this work proposes using a gain-scheduled scheme for tuning the gains of a conventional PI controller as a function of speed error. While a conventional fixed-gain proportional integral controller with its proportional and integral gains tuned properly for a specific operating condition does already offer a good level of transient response (as long as the operating point does not significantly deviate from its initial position), this kind of fixed-gain PI controller does not offer sufficient transient response for variable operating conditions. The main limitation is that most of the electromechanical drive systems exhibit non-linear dynamics, so that as the operating point varies over a wider range, it becomes difficult for a conventional fixed-gain PI controller to maintain a satisfactory level of transient performance.

The problem can be alleviated by re-tuning the proportional and integral gains of the controller for the new operating point. In principle, it is possible to design several linear PI controllers for different operating points over the complete operating range and switch from one controller to another as the operating point changes. This is difficult to achieve manually and hence is carried out in an automated manner by using the gain-scheduled scheme of adaptive control [65-66]. To implement the proposed gain scheduling approaches, it is first necessary to know the range over which the proportional and integral gains can oscillate under variable operating conditions. After the gain ranges are calculated, the proportional and integral gains are permitted to vary within the predetermined range according to both the operating point and the error signal. If the speed error is small, a small proportional gain is applied to boost the control effort in order to either accelerate or decelerate motor speed as quickly as possible. Similarly, when there is only a slight speed error, a large value of the integral gain is used to overcome the steady-state error. The performance of the proposed scheme is established on MATLAB/Simulink to test the IM drive system with a change in the step speed command under different operating conditions. The simulation results show improvements in transient as well as steady-state performances over the conventional fixed-gain PI controller.

The proposed gain scheduling adaptive PI controller is characterized by low computational time, ease of implementation, and suitability for practical applications. It boasts the following main characteristics compared to conventional fixed PI controllers:

- Accuracy in trajectory tracking.
- Robustness in the face of parameter variations and load torque disturbances.

Thus, the primary contributions of this study are as follows:

- The gain-scheduled PI speed controller provides enhanced performance and prevents overshooting and undershooting during changes of step-speed reference and load torque disturbances.
- The effectiveness of the proposed self-tuning PI controller is verified using computer simulations that investigate trajectory tracking accuracy and robustness properties against load torque disturbances and parameter variations.
- The proposed gain-scheduled PI controller is compared with conventional PI controllers and emerges with several notable advantages.

4.2 Gain Scheduled Controller Structure

Gain scheduling is an approach of controlling nonlinear system. It uses a set of linear controllers. For a different operating point of the system, each controller provides satisfactory control. To implement gain-scheduled controller, the gain is automatically adjusted according to scheduling variables which define the present operating point. For instance, theses variables include time, system states like orientation or velocity or external operating conditions.

For designing gain scheduled control systems, usually a small set of operating points are selected. At each point, a linear controller is designed properly.

During various operating point, the system changes between the different linear controllers, depending on the present values of the scheduling variable.

Gain scheduling is more appropriate when the scheduling variables are vary slowly compared to the control bandwidth, like ambient temperature of a chemical reaction or the speed of a cruising aircraft, motor rotor position.

Gain scheduling is challenging when the scheduling variables have fast-varying states of the system. Because there is no guarantee of local linear performance near operating points of global performance in nonlinear systems, extensive simulation-based validation is required.

Gain-scheduled design involves

- An *operating range* is required to be determined as a set of range that allow the values of the relevant system parameters to remain during operation.

A *gain schedule*, which includes the formulas or data tables that yields the appropriate controller gains for given values of the scheduling variables. See [165] for an overview of gain scheduling and its challenges.

The main advantage of the method proposed in this chapter is that the gains are allowed to vary over a given range under varying operating conditions. Since the proportional term (K_p) improves overshoots and rise time response while the integral (K_i) term reduces steady state error, when a large speed error occurs, a high value of proportional gain is required for better control performance. Likewise, when a small speed error occurs, a high value of integral gain is required to minimize steady state

error. The output of the gain-scheduled PI controller is the reference torque, which can be expressed as

$$T^{*} = K_p e(t) + K_i(t) \int e(t)\,dt \tag{4.1}$$

where $e(t) = \omega_r^{*}(t) - \omega_r(t)$, $K_p(t)$ is the proportional gain, and $K_i(t)$ is the integral gain. These gains are functions of the speed error $e(t)$. The gain $K_p(t)$ is expressed as a function of the speed error as follows:

$$K_p(t) = K_{p(max)} - \left(K_{p(max)} - K_{p(min)}\right) e^{-[ke(t)]} \tag{4.2}$$

where k is a constant that decides the rate at which $K_p(t)$ varies between maximum and minimum values of the proportional gain. A large proportional gain, $K_{p(max)}$, is used to speed up the transient response when speed error $e(t)$ is large, and when error $e(t)$ becomes small, a minimum proportional gain $K_{p(min)}$ is used to eliminate overshoots and oscillations. The integral gain $K_i(t)$ is expressed as a function of the speed error signal $e(t)$ as

$$K_i(t) = K_{i(max)} e^{-[ke(t)]} \tag{4.3}$$

The integral gain $K_i(t)$ in equation (4.3) varies in the range of

$$0 \le K_{i(t)} \le K_{i(max)} \tag{4.4}$$

It can be noted from equation (4.2) that the exponential term approaches zero when the error $e(t)$ is large. Therefore, the proposed gain $K_p(t)$ is expressed as $K_{p(max)}$.

In the same way, the exponential term approaches one as the error $e(t)$ gets smaller, and consequently, $K_p(t)$ is expressed as $K_{p(min)}$.

The Ziegler-Nichols tuning method can be utilized to determine the gains of the proposed gain-scheduling PI controller. The value of $K_{p(min)}$ is determined to be equal to 0.45 K_p, while the value of $K_{p(max)}$ is determined to be equal to twice K_p. The value of $K_{i(max)}$ is determined to be small. The value of the gain K is determined to be approximately 0.1.

If the speed error is found to be large, then a large proportional gain is needed to attain better control performance. Therefore, the motor speed should accelerate or decelerate to the command speed as quickly as possible. Conversely, if the speed error is found to be small, then a large integral gain is required to overcome the steady-state error.

Some advantages of the proposed gain-scheduling PI scheme include a reduction in computational burden as well as a simple and inexpensive implementation cost.

In the initial transition period, in order to accelerate or decelerate motor speed to the desired reference as quickly as possible, a large control signal is necessary. To create this large control signal, the gain-scheduling PI controller has to generate a large proportional gain $K_p(t)$, as expressed in the previous equation (4.2), while the integral gain $K_i(t)$ is kept constant at its minimum value. Once the motor speed attains command speed, the speed error signal becomes smaller. Then the proportional gain $K_p(t)$ reaches its minimum value. In a comparable manner, during the steady-state period, to

overcome the steady-state error, the integral gain $K_i(t)$ is increased to its maximum value. As a result, as ensured in equations (4.2) and (4.3), the proportional gain $K_p(t)$ and the integral gain $K_i(t)$ vary as a function of speed error $e(t)$. In this way, the performance of both steady-state and transient operations are enhanced.

4.3 Simulation Results and Discussion

In order to demonstrate the effectiveness of the proposed PI gain-scheduling controller based on IFOC of the IM drive, simulations are carried out under a variety of operating conditions. The ratings and parameters for the tested induction motor are listed in the appendix. Several tests were conducted in MATLAB/SIMULINK to evaluate the performance of the proposed method to ensure that it can operate in a wide range of conditions. The speed responses were observed under different operating conditions such as a sudden change in command speed, step change in load, etc. The results are presented in this section.

A comparison between the conversion PI controller and the gain scheduling PI controller is then evaluated under sudden change in command speed, and load disturbances

4.3.1 Trajectory tracking performance

To determine trajectory tracking performance, IM drive system tests are conducted under varying speed controllers. The simulated speed responses by conventional fixed PI and gain-scheduling PI controllers during no load are demonstrated in figure 4-1. The speed response of the IM drive for load torque step change is demonstrated in figure 4-4, while the corresponding response of the motor stator phase current is demonstrated in figure 4-3. It is clear from this test that both conventional and self-tuning PI are capable of precisely tracking desired command and they show almost the same response. Further tests are applied and when the step change command is increased to 125 rad/sec. as demonstrated in figure 4-4, throughout this test both techniques follow the tracking speed command but gain scheduling PI is smoother than fixed PI and requires less rise time to reach the command speed.

4.3.2 Effect of Load Disturbances

The robustness of the proposed approach is evaluated under sudden load impact. Figure 4-4 illustrates the response attained when the rated torque is applied to the IM. It can be seen that the proposed approach has the capability to reject load disturbance quickly, with only a slight dip in the motor speed, unlike the conventional PI controller, which shows fluctuation during rising time.

The results clearly demonstrate that with regard to overshoots, settling time, and rise time the proposed approach provides superior efficiency and is better than the conventional PI controller.

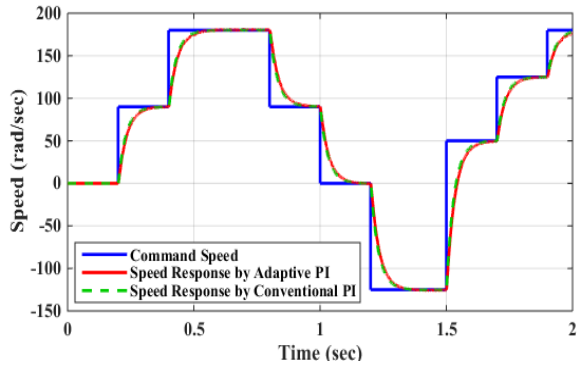


Figure 4-1: Speed tracking response for PI and PI gain-scheduling controller techniques without applying load.

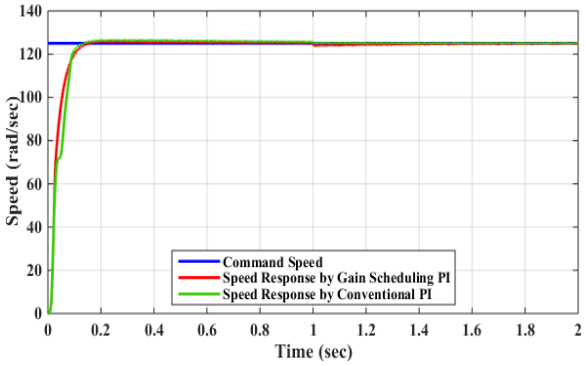


Figure 4-4: Speed tracking response for PI and PI gain-scheduling controller techniques with load torque (2 Nm).

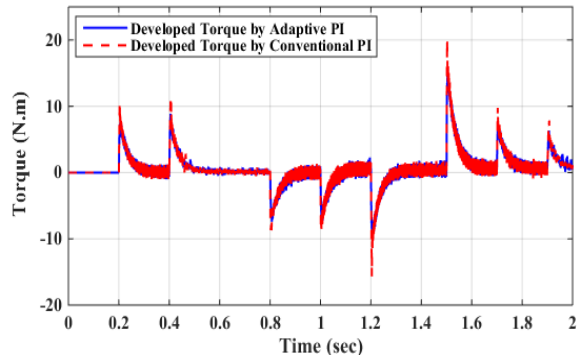


Figure 4-2: Torques of the PI and PI gain-scheduling controller techniques corresponding to the speed command given in Fig. 4-1.

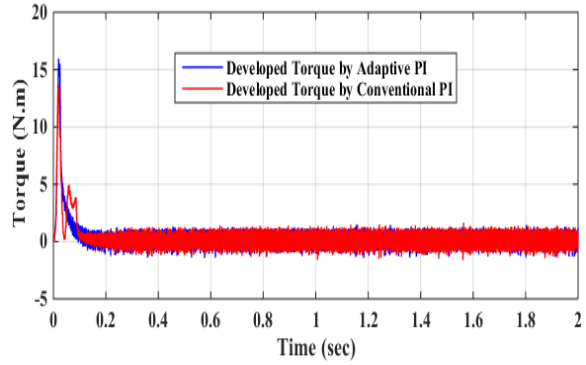


Figure 4-5: Torques of the PI and PI gain-scheduling controller techniques corresponding to the speed command given in Fig. 4-2.

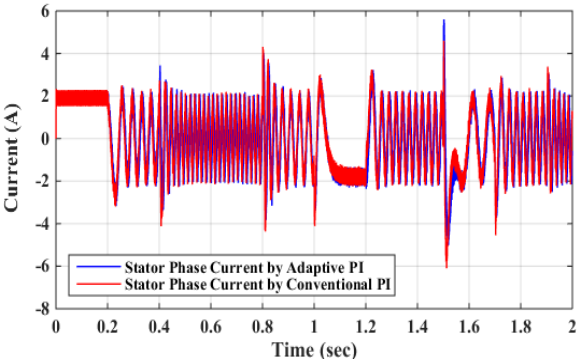


Figure 4-3: Stator Phase Current of the PI and PI gain-scheduling controller techniques corresponding to the speed command given in Fig. 4-1.

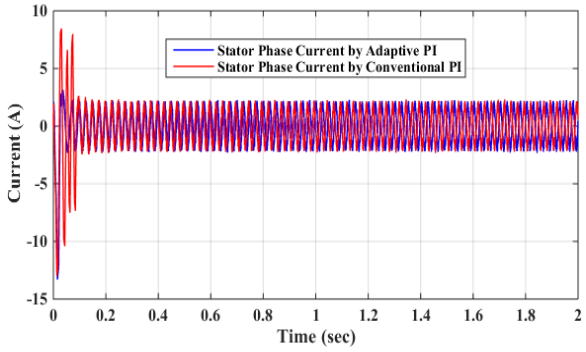


Figure 4-6: Stator Phase Current of the PI and PI gain-scheduling controller techniques corresponding to the speed command given in Fig. 4-2.

4.4 Conclusion

A PI gain-scheduling controller for an induction motor drive based on field oriented control has been presented in this section. In addition, the performance of the proposed approach has been compared with the conventional PI controller, and has been found to be superior. Simulation results confirm the success of this approach. Further, the approach is simple and easily implemented, and represents a significant reduction in computational time.

4.5 Grid Search Method in PI Controllers

4.5.1 Introduction

Two core abilities must first be established if optimal electrical drive performance is to be ensured. The first of these abilities is steady state and dynamic tracking in order to set point changes, and the second is the rejection of load disturbance [65-67] [72]. Traditional controllers for this type of drive are developed by referencing the linearization at the operational point. However, these controllers are most effective when there is only a small load change and the system is kept close to the equilibrium point.

To achieve the most efficient performance, various approaches have been developed and used to control induction motor speed. One of these is the proportional-integral (PI) controller, which enjoys wide usage due mainly to its simplicity and ease of implementation. While new control approaches such as the Artificial Intelligent Controller (AIC) are being applied to motor drives (which include expert system, fuzzy logic, neural networks and genetic algorithms that are particularly effective), [86-101] these systems are inherently complicated. Furthermore, because the PI controller is the most common for IM drives in industrial applications, it is a challenge to find a more suitable way to adjust its gains. However, without tuning, the PI controller is slow to adapt to changes in speed, load disturbances, and variations in parameters. The usual means to address these challenges is to manually tune the gains through observation of the system output. Additionally, its gains are fixed and therefore it has difficulty adapting to a wide range according to the induction motor parameters. Consequently, as the

system is typically subject to sudden unpredictable changes, the robustness and performance of the PI will degrade in certain industrial control applications.

The most common tuning method for the PID in industrial applications is the Ziegler-Nichols method. This approach does not require an accurate system model, and its control parameters are defined by the ultimate gain and period, which are obtained by observing the oscillation of the proportional closed loop controller [68-70]. One key advantage of the Ziegler-Nichols method is its capability to reject disturbances. However, this method also creates certain challenges. For instance, since a procedure of trial and error has to be performed, the process is time-consuming. Additionally, the Ziegler-Nichols method forces the system into a marginally stable condition, which in turn can cause an unstable operation or dangerous situation, its step response overshoots significantly, and it requires a high control signal for satisfactory system performance. To determine PI controller gains, another tuning approach utilizes system frequency response, based on particular phase and gain margins, along with crossover frequency [72-73] [102-107].

All of the above methods are considered model-based strategies. However, it is a model's accuracy along with the conditions assumed concerning actual operating conditions that determine the tuning method. Therefore, to alleviate the problems inherent in PI controllers in a dynamically changing system such as an IM drive, researchers have used many techniques to make PI controllers more adaptive. Some have proposed the use of self-tuning PI controllers [37] [67] [73], while others have proposed the use of optimization techniques in order to automatically attain optimum PI gains.

For example, earlier in this chapter the use of a gain scheduling method instead a fixed PI controller was described, but for successfully implementing such a method, it is necessary to know the range over which the proportional and integral gains could vary for varying operating conditions. Another approach described in this chapter is to couple a genetic algorithm to design a PI controller to obtain the optimum PI gain [98-101]. Optimum gain generally converges to a response having minimum error. However, in this study it is proposed to use the Grid Search Method, along with the integrated squared error, which is a performance index, to determine PI controller gains and find the value of error in the iteration.

4.6 Self-Tuning Regulator Based Speed Controller for IM Drive Systems

The proposed PI Self-Tuning controller compares the response of an Self-Tuning model with a desired response and the difference between them is used to update the controller mechanism of the adjustable model as illustrated in figure 4-7. In this work, a simulation is used to generate a desired response, though an operating experience can also be used to generate this response. Reference speed is the response of the reference model. The controller mechanism employs the function of the Integral Square Error (ISE) and steepest descent method for finding the local minimum of a function which presumes that the gradient of the function can be computed.

The function of the ISE can be defined as

$$ISE = \sum_{k=1}^{N} e(k)^2 \tag{4.5}$$

where N is the number of samples (time instants).

Initially, the Self-Tuning gains K_p and K_i are set to arbitrary non-optimal values. However, if the gains are set to a value smaller than the optimal ones, the system shows sluggish response and requires a long time to converge to the steady-state. On the other hand, if the gains are set to values greater than the optimal ones, the system becomes underdamped and experiences large oscillations which sometimes create system instability. Thus, optimal values must be found between these two.

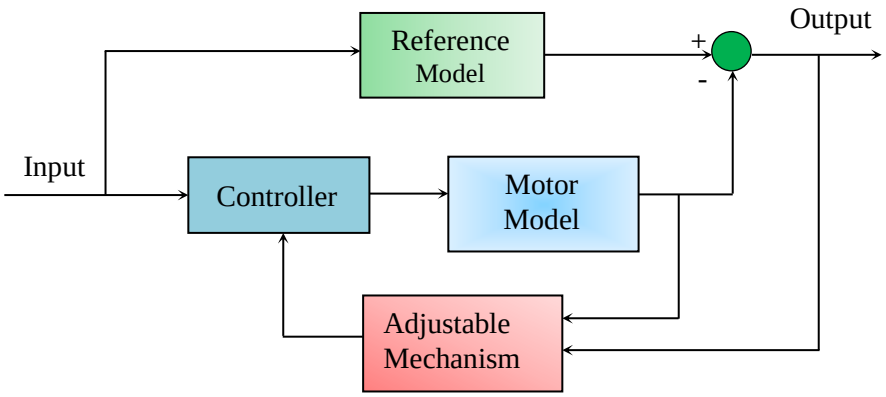


Figure 4-7: Schematic of the Self-Tuning control system

4.7 Grid Search Method for an Self-Tuning PI Controller

With a classic PI controller applied to control an induction motor, when all the model parameters of the IM, external load torque, disturbance, and other operating conditions are well known, high performance can be obtained. When these conditions or system parameters are not known or are uncertain, the fixed PI controller cannot guarantee the optimum performance. If a speed tracking problem occurs where the required speed is a function of time, the optimum values of the controller parameters cannot be easily found. In such cases, a technique to continuously optimize the PI

controller gains is preferable. Some of these techniques, however, are very difficult to implement in a complex system, and they impose extended controller computational times.

In this work, one of the optimization techniques is used to optimize the PI gains. The approach requires use of a reference model run by any type of satisfactory controller, and the output of this model will be considered as the target to be achieved. The approach also requires another adjustable model controlled by the PI controller, and its gains are determined by the grid search optimum method. The objective in using the grid search method is to continuously adapt the optimum gain parameters of the PI controller. The objective function is the squared speed error between the reference model and the adjustable model. The basic principle of the grid search is as follows: select the size of the search area defined by the upper and lower bounds of each of the independent variables, set the step length for each of the independent variables, and search the parameter according to the step length. Thus a 3-d suitable plane grid is formed over the area of interest and the objective function is evaluated at each node of the grid. After the computation of the objective function values at all the nodes of the grid (each intersection node on the grid corresponds to a set of independent variables), it is possible to interpolate and find a minimum between the grid lines, as depicted in figure 4-8.

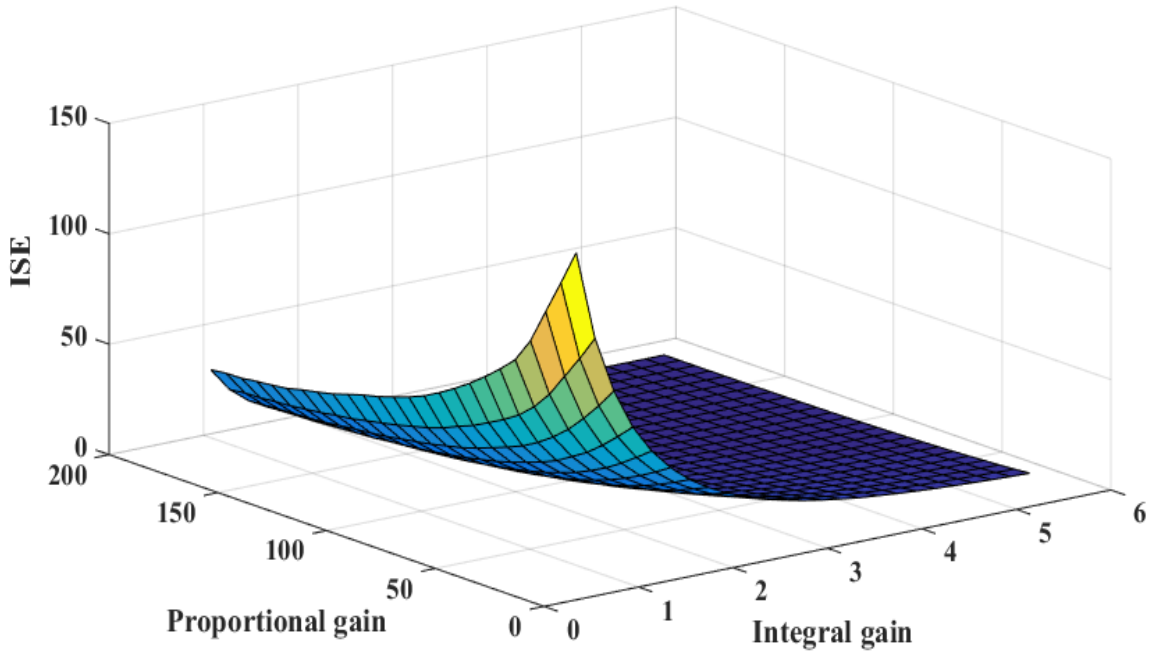


Figure 4-8: Integral Squared (ISE) speed error versus controller parameter grid

This study uses the grid search method to determine the optimum values of the proportional and integral gain parameters for different types of speed control and to continually adapt these parameters automatically in real-time. To minimize learning time and serve as a starting point, initial values are given to the proportional gain and integral gain corresponding to the minimum value of the objective function over the grid, and then step size is set to 4 and 10 for the proportional and integral gains, respectively. Figure 4-9 illustrates the order of procedures that take place during this particular approach.

The algorithm is relatively simple to program compared to other optimization algorithms and also has the advantages of fast convergence and no requirement for

accurate knowledge of the system model. However, it is inefficient in that it requires extended computational time.

The performance of the proposed grid search method to optimize a proportional-integral controller with field-oriented control is simulated and compared with that of the conventional fixed gain proportional-integral controller under different speed commands, repetitive operations, parameter variations, and load disturbances. The computer simulation and experimental results are given to demonstrate the effectiveness of the proposed PI learning controller, showing its superiority to the conventional fixed PI controller.

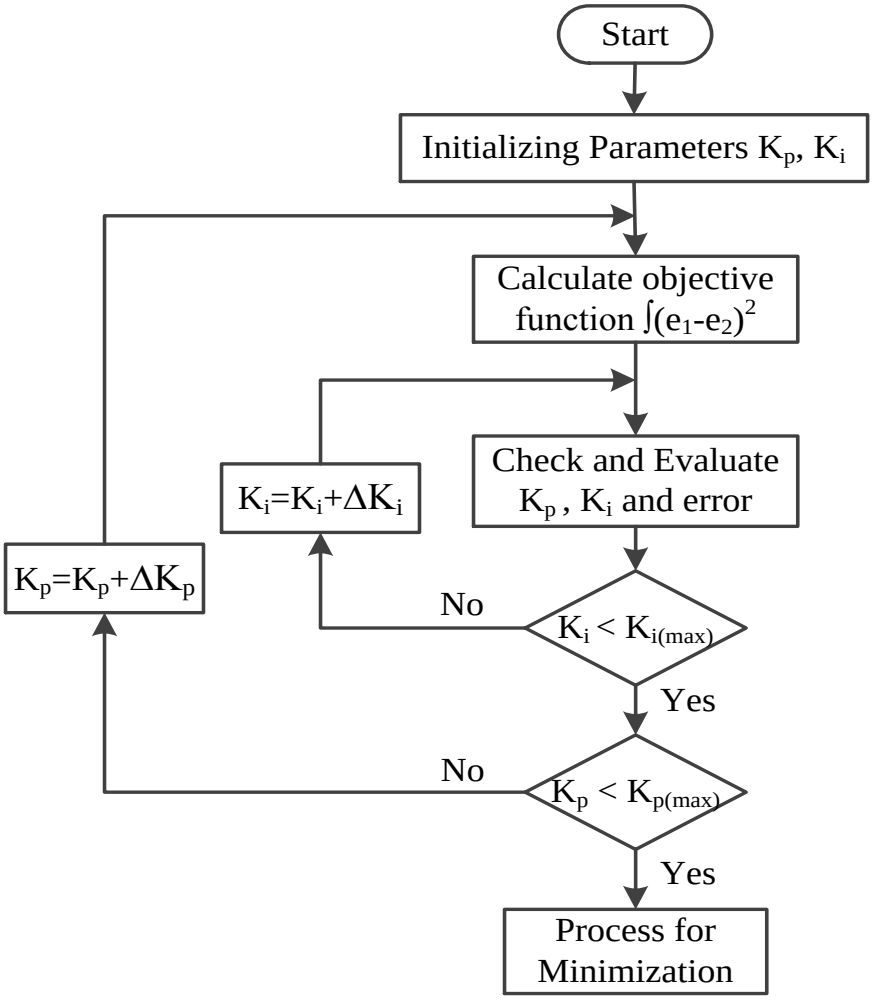


Figure 4-9: flowchart of the proposed methodology

4.8 Simulation Results

In order to evaluate the proposed algorithm, a series of simulation tests were carried out on an indirect vector controlled induction motor drive using both a conventional tuned PI controller and a PI controller using the grid search method to find and automatically adapt the optimum controller gains. The speed and current responses of the induction motor were observed under different operating conditions, such as sudden change in reference speed and step change in load.

Figures 4-10, 4-11, 4-12, and 4-16 show the speed of the proposed technique in the initial stage and have overshoot in these four cases. This represents the response of the algorithm at initial start-up, using semi-arbitrary gains and before the controller has had time to execute its self-tuning process. Performance then improves gradually before reaching optimization, as illustrated in figures 4-13, 4-14, 4-15 and 4-19, where motor speed is shown to converge to the desired speed. It is also clear from the figures that the algorithm has high precision in tracking the command speed and is able to recover quickly from load disturbances.

Figure 4-18 shows that a small disturbance of stator current occurred at initial startup of the motor from standstill. At step change the current response also fluctuated significantly but recovered within a short time, and at the sudden load disturbance there was also current fluctuation. After the self-tuning process these disturbances were eliminated, as shown in figure 4-21. Along with these figures, figure 4-20 and 4-21 show that after the self-tuning process, the controller reached a performance standard at least equal to that of an optimally tuned fixed gain PI controller. The GSM (Grid Search Method) based controller located the optimum proportional and integral gains corresponding to the minimum objective function over the grid for the self-tuning PI controller, as indicated in Table 4.1.

Table 4-1: Conventional and self-tuning PI controller parameters

Conventional PI	$k_p = 3.2$	$k_i = 3.6$
Grid Search Method	$k_p = 3.3$	$k_i = 3.4$

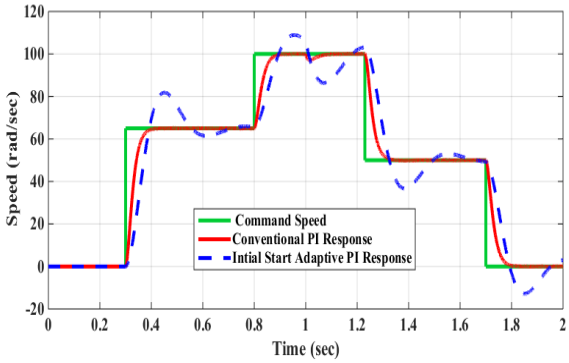


Figure 4-10: Initial speed tracking response with PI controller with load torque (2 Nm)

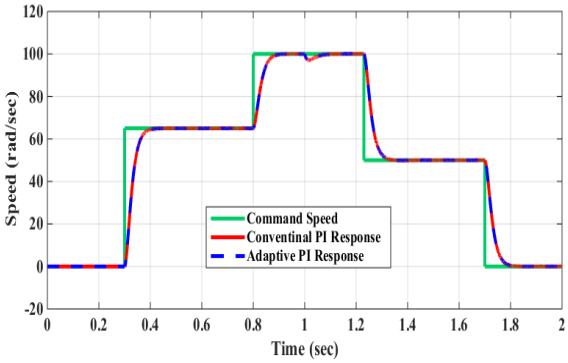


Figure 4-13: Final speed tracking response with PI controller with load torque (2. Nm)

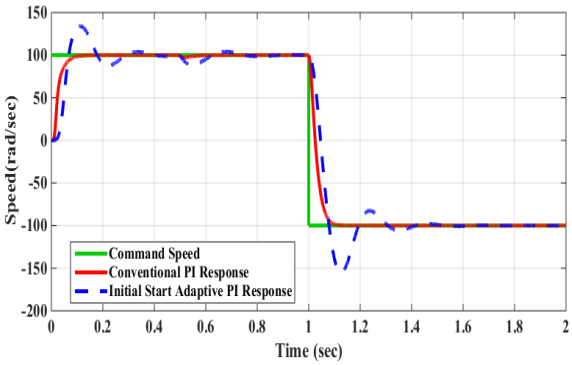


Figure 4-11: Initial speed tracking response with PI controller with load torque (2.Nm)

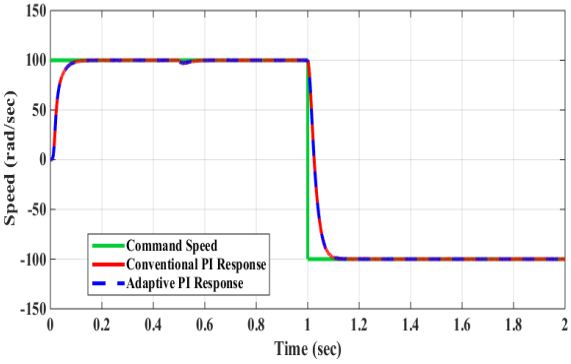


Figure 4-14: Final speed tracking response with PI controller with load torque (2. Nm)

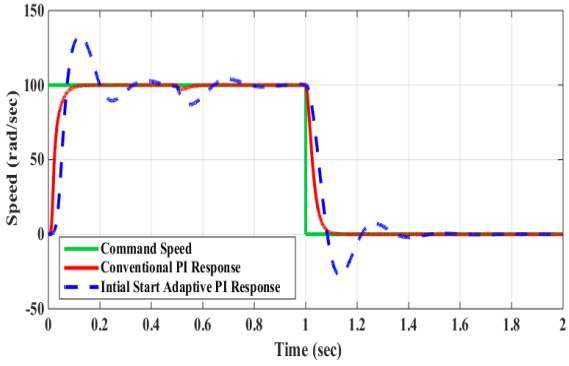


Figure 4-12: Initial speed tracking response with PI controller with load torque (2 Nm

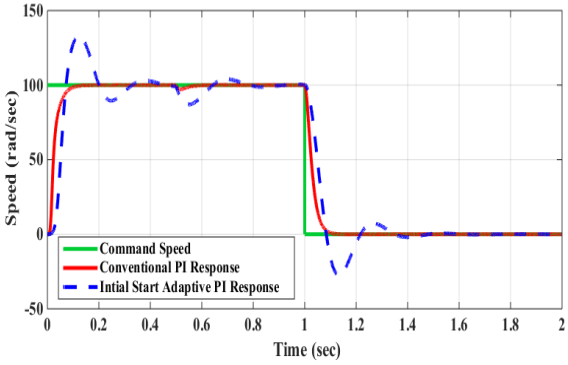


Figure 4-15: Final speed tracking response with PI controller with load torque (2 Nm)

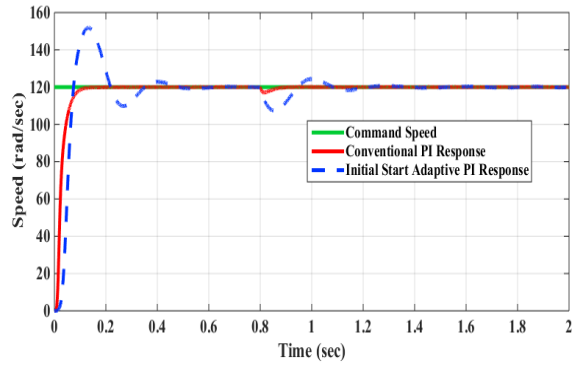


Figure 4-16: Initial speed tracking response with PI controller with load torque (2 Nm)

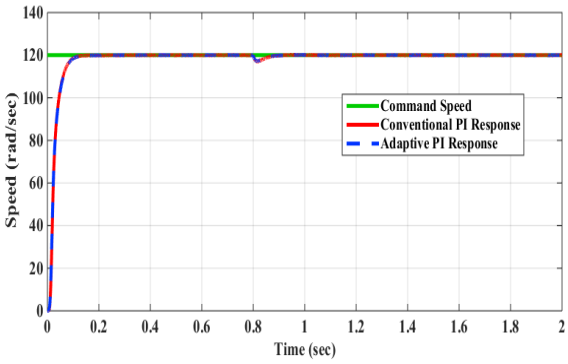


Figure 4-19: Final speed tracking response with PI controller with load torque (2 Nm)

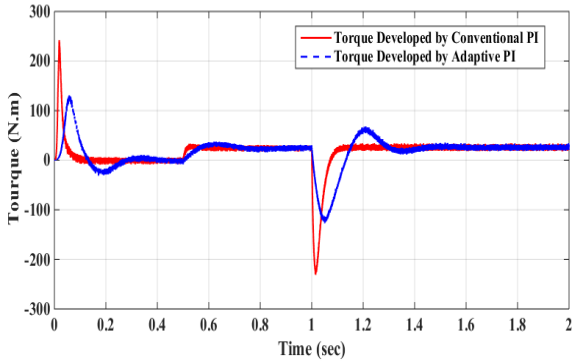


Figure 4-17: Initial torque developed for conventional and self-tuning PI controller corresponding to the speed given in figure. 4-16

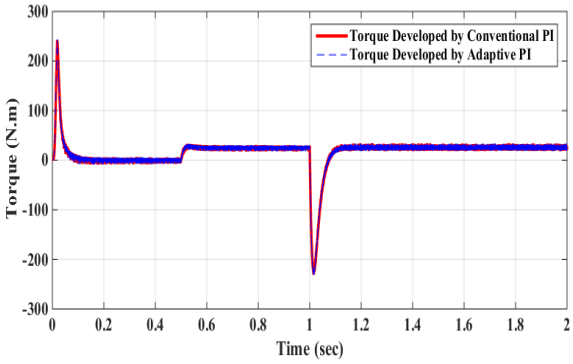


Figure 4-20: Final torque developed for conventional and self-tuning PI controller corresponding to the speed given in figure. 4-16

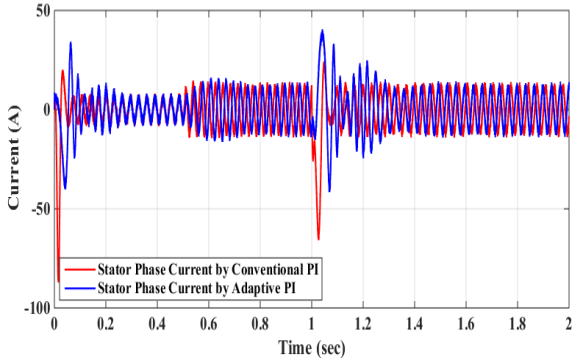


Figure 4-18: Initial one-phase stator current for conventional and self-tuning PI controller corresponding to the speed given in figure. 4-16

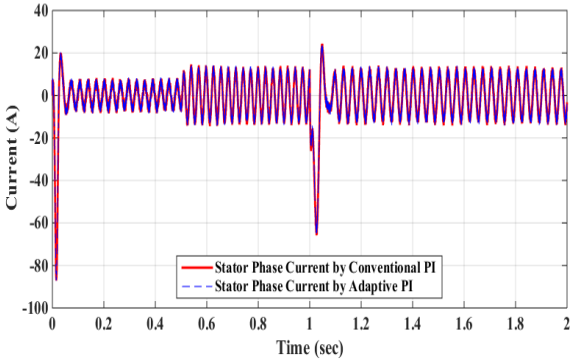


Figure 4-21: Final one-phase stator current for conventional and self-tuning PI controller corresponding to the speed given in figure. 4-16

4.9 Summary

A traditional fixed-gain linear proportional integral (PI) controller is inadequate to provide satisfactory transient performance under varying operating conditions. In this chapter, two schemas were proposed as alternatives to solve this problem. It was first proposed to use a gain-scheduled PI scheme for the IM Drive with a pre-determined range of variance, which improves transient and steady-state performance of the system in comparison with a fixed-gain PI controller. Secondly, a Grid-Search Method can be used to determine optimum PI controller gains in conjunction with a performance index such as integrated squared error.

Chapter 5.

A Fast Fuzzy Logic Controller Based on Vector Control for Three Phase Induction Motor Drives

Fuzzy Logic Control has evolved into being one of the most desirable controller methods in current use. Due to its flexibility, simplicity and the lack of need for complex mathematics even if the system model is unknown, the popularity of the Fuzzy Logic Controller (FLC) is growing and spreading in varied applications [108-125]. This chapter presents an Induction Motor (IM) drive which incorporates an FLC with a novel output membership function that helps to increase controller speed. The vector control scheme of the FLC-based IM drive is carried out experimentally using a dSPACE digital signal processor board DS-1104 and laboratory ¼ hp IM. To verify the efficiency of the FLC-based IM drive under consideration, various operating conditions including sudden change in command speed and step change in load are tested in simulation as well as lab experiments, and then compared to those achieved from the conventional proportional-integral (PI) controller-based drive. As will be discussed, the FLC has been proved to be an effective alternative to the traditional PI controller for the high performance required in industrial drive applications.

5.1 Introduction

AC motor drives are widely used in industrial and process high-performance applications where motor speed must precisely follow a specified trajectory regardless of parameter variations, load disturbances and other uncertainties. In such applications,

controller design plays a critical role in system performance. The decoupled flux and torque vectors of vector controlled IM drives are undesirably affected by parameter variations in the running motor. The traditional fixed gain proportional-integral (PI) controllers commonly used to control IM drives in industrial applications, however, are very sensitive to these parameter variations and load disturbances. Thus, the control performance of the conventional PI controller method can deteriorate substantially under parameter variation [38][73]. Consequently, for optimal control, the controller parameters must be continually adapted.

Among the adaptive controller methods that have been applied to overcome these challenges are the model reference adaptive control (MRAC) [74-77] and the sliding mode control (SMC) [78-84]. The design of these controllers relies on the precise mathematical model of the system. It is often difficult, however, to develop an accurate mathematical model because of unknown and difficult to predict variations in load and operating parameters caused by temperature variations, saturation, and system disturbances. Additionally, the system parameters gradually change during the operating period; therefore, after a long running time, the control performance can be degraded if changed system parameters are not updated. In order to overcome these problems, and provide control which is not dependent upon precise knowledge of the system model, the fuzzy-logic controller (FLC) is used for IM control purposes. A fuzzy rule-based system alters the control parameters in a fuzzy logic controller, long considered a viable and straight-forward model of human behavior regarding process control. FLC has several benefits compared to conventional controllers, the four main ones of which are as follows:

1) Accurate mathematical modelling of a system is not necessary to build an FLC.

2) FLCs are generally more robust than conventional controllers.

3) FLCs are sufficiently flexible to handle systems under nonlinear control.

4) FLC principles are contingent on linguistic control rules that form the basis of human logic.

Several recent studies have investigated the viability of developing fuzzy algorithms for motor control applications as well as system modelling [22-23] [86-91] [108-125]. However, applying FLCs in motor drives raises a number of issues around technology use, especially during hardware implementation, due to the relatively high computational burden it can impose. Earlier works reporting fuzzy logic applications in motor drives were based either on simulation or experimental results at low operating speed conditions [90].

The objective of this chapter is to present a simplified FLC-based field-oriented speed controller for the IM with a high performance standard, utilizing a simple structure for the output membership function. Consequently, computational burden is minimized, allowing computers with limited processor and memory resources to operate the control algorithm in real time, at increased motor speeds and reduced currents. A specific FLC for the IM drive is designed and successfully implemented in real time and compared with results obtained from conventional PI controllers, both in simulation and experimentally. It is found that the proposed FLC is not affected by temperature changes, inertia variations, or load torque disturbances. This novel FLC could thus prove to be an appropriate alternative for the conventional PI controller to achieve high performance in IM drive systems.

5.2 Fuzzy Logic Controller Structure

The basis for fuzzy logic principles is the simulation of human thought patterns through mathematically-framed linguistic concepts. The FLCs are built according to the tenets of both the control process and the quality demanded. Due to the increasing desirability and applicability of fuzzy control, various possible controller structures are being developed. Not all fuzzy controller structures are alike. Here are some of the ways in which they can be different: the number of inputs and outputs, the number of input and output fuzzy sets and their shape of membership function, the type of inference engine, the form of control rules, and the defuzzification method. It is up to the designer to decide which controller structure would be optimal for a particular control problem.

The fuzzy system is comprised of input fuzzy sets, fuzzy rules, and output fuzzy sets. Fuzzy control processes generally proceed according to the following stages:

- Fuzzification is the process of relating a numerical (crisp) value of a variable to a fuzzy set by means of a membership function. This type of function is typically represented graphically and shows the extent to which a crisp variable belongs to a fuzzy set. Two of the most popular types of membership functions are trapezoids and triangles.
- The Inference Engine decides which rules from the fuzzy rule-base are applicable to the given input conditions and executes those rules to compute the fuzzy output functions. The fuzzy rule-base consists of rules to govern and execute the relations between inputs and outputs for the system.
- The process of “defuzzification” manipulates the fuzzy output functions created by the Inference Engine in order to crisp the output values. In the present study, the

method used for defuzzification is centroid calculation of the area under the output function.

5.3 Control Strategy

Corresponding to PI-based motor speed control, inputs into the FLC would be error (E) and the integral of error (I), and controller output would be the control signal Q. In this work, trapezoid membership functions are used for both controller inputs. Two linguistic variables are used for each of the input variables, as shown in figure 5-1 and 5-2. Errors and integral of errors can be positive or negative, so the minimum number of rules for each input is 2 which makes the output 2 by 2 or 4. Adding an extra rule for each input to focus attention near zero gives 3 by 3 or 9 rules. For example, negative (N) means that the actual speed is more than the set point (reference speed) and positive (P) means that the actual speed is less than the set point. The output variable fuzzy set is shown in figure 5-3. Again, linguistic variables are used: Negative Negative (NN), Negative Positive (NP), Positive Negative (PN), and Positive Positive (PP). The applicable ranges of input error and integral of error are given as [-30, 30] and [-3, 3], respectively, and the output variable as [-10, 10]. These ranges require careful consideration and have been specified according to previous results obtained from the conventional PI controller.

A membership function provides the degree of membership of an input value to every fuzzy set. An input variable might be associated with more than one fuzzy set and, generally, neighbouring membership functions have some degree of overlap. However, output functions are not allowed to overlap for the sake of simplifying controller performance. A high degree of control accuracy can be achieved by selectively adding

fuzzy rules. However, this also increases complexity and may be difficult to practically implement due to increased computational burden.

The objective is to design a novel fuzzy logic controller that is robust, simple and capable of serving as an alternative to the traditional PI, with minimum IM speed error.

The first step requires the conversion of the crisp input into a fuzzy one. Since there are two inputs for the traditional PI controller, there will be two crisp values to convert. The first value is the level of error, while the second value is the level of integral of error. The sets designated for both speed error and integral speed error are negative (N) and positive (P), as demonstrated in figure 5-1 and 5-2.

What should be noted is that on the left side of error membership, when the error is lower than negative error breakpoint (-BE), its degree of membership in the negative fuzzy set (N) is equal to one, while its degree of membership in the positive fuzzy set (P) is equal to zero.

On the right side of the same error membership, when the error is greater than positive error breakpoint (+BE), its degree of membership in the positive fuzzy set (P) is equal to one, while its degree of membership in the negative fuzzy set (N) is equal to zero.

As for the error that occurs between the two break points (-BE, +BE), in this case, its degree of membership in the negative fuzzy set can be calculated according to this formula:

$$NE = 1 - \frac{E - (-BE)}{BE - (-BE)} = 1 - \frac{E + BE}{2BE} \tag{5.1}$$

while its degree of membership belonging to the positive fuzzy set is calculated according to this formula:

$$PE = \frac{E - (-BE)}{BE - (-BE)} = \frac{E + BE}{2BE} \tag{5.2}$$

where NE and PE represent negative error and positive error respectively, and –BE and +BE represent negative error break point and positive error break point respectively. Of course, the total degree of their membership function is equal to one.

The same principle will be applied for integral error (I). As for the integral error that occurs between the negative integral break point (-BI) and the positive integral break point (+BI), in this case, its degree of membership in the negative integral fuzzy set can be determined according to this formula:

$$NI = 1 - \frac{I - (-BI)}{BI - (-BI)} = 1 - \frac{I + BI}{2BI} \tag{5.3}$$

while its degree of membership associated with the positive integral fuzzy set is determined according to this formula:

$$PI = \frac{I - (-BI)}{BI - (-BI)} = \frac{I + BI}{2BI} \tag{5.4}$$

A graph is used to show the output membership function. In figure 5-3, the rules are converted to a plot of four fuzzy sets (NN, NP, PN, PP) applied together to create one fuzzy set. In other words, each rule has one output fuzzy set. According to the defined rules used in this application, the fuzzy operation is the conjunction (AND). Therefore, the membership function of the two fuzzy sets is defined as the minimum of the two individual membership functions.

Suppose the inputs are speed error (X= –3) and integral speed error (Y= 0.5). The fuzzy values for these crisp values can be obtained by using the membership functions of the applicable sets.

Thus, error membership in the negative fuzzy set (N) is equal to 0.65, while its membership degree in the positive fuzzy set (P) is equal to 0.35.

$$E_{N(-3)} = 0.65$$

$$E_{P(-3)} = 0.35$$

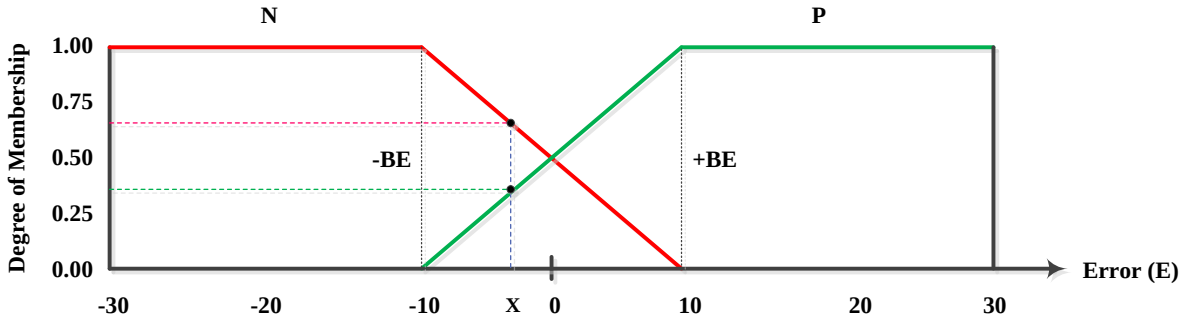


Figure 5-1: Membership Functions (MFs) for Input Error (E)

The fuzzy values for integral error are shown below.

$$I_{N(0.5)} = 0.25$$

$$I_{P(0.5)} = 0.75$$

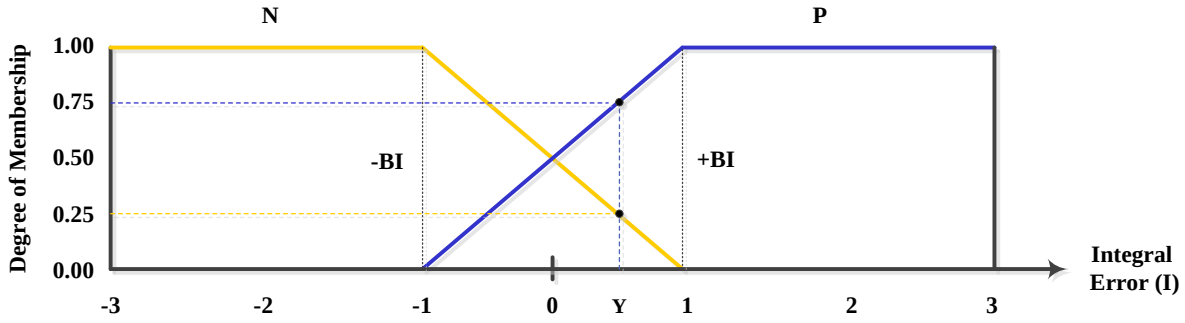


Figure 5-2: Membership Functions (MFs) for Input Integral Error (IE)

Once the fuzzy values are determined, then the fuzzy rules must be applied to arrive at the final fuzzy value.

5.4 Rule Evaluation

To evaluate the rules in a fuzzy logic controller, a conditional statement needs to be applied. An example would be (if..., then...,). In this example, a linguistic form would be used to express the antecedent (if …) and the consequent (then...). Two processes are followed in the rule evaluation. The fuzzy operator (i.e., AND or OR) is attached to the antecedent, after which the implication is transferred from the antecedent to the consequent. The fuzzy operators utilized in the stated fuzzy rules are AND (∩), OR (∪) and NOT (−), which are explained in detail below.

a) AND means classical intersection: $m_{A\cap B} = \min\{m_A(x), m_B(x)\}$

b) OR means classical union: $m_{A\cup B} = \max\{m_A(x), m_B(x)\}$

c) NOT means classical complement: $\bar{m}_a = 1 - m_a(x)$.

Typical rules which are used in this application are as follows:

1. If the error (E) is negative and the integral of error (I) is negative, then the control signal Q is a large negative (LN).
2. If the error (E) is negative and the integral of error (I) is positive, then the control signal Q is a small negative (SN).
3. If the error (E) is positive and the integral of error (I) is negative, then the control signal Q is small positive (SP).
4. If the error (E) is positive and the integral of error (I) is positive, then the control signal Q is large positive (LP).

Each fuzzy logic statement or rule has an IF/THEN structure in the main part of the fuzzy controller obtained from human knowledge and expertise.

For a specific set of error and integral of error inputs, manipulation of each fuzzy logic rule produces a contribution to an overall fuzzy range of outputs. It is assumed that when a rule has an IF/AND/THEN structure, such as is the case for each rule here, the minimum criterion is being used for the fuzzy AND operation and consequently is mapped on the corresponding output. If the preceding rules are followed, these results will be obtained:

Rule (1) = MIN (0.65, 0.25) = 0.25

Rule (2) = MIN (0.65, 0.75) = 0.65

Rule (3) = MIN (0.35, 0.25) = 0.25

Rule (4) = MIN (0.35, 0.75) = 0.35

The fuzzy logic of the preceding rules is demonstrated in figure 5-3.

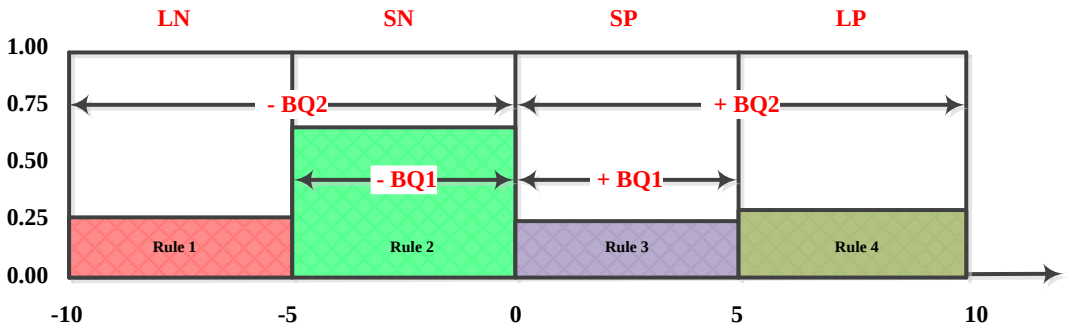


Figure 5-3: Crisp Output Membership Functions

5.5 Aggregation and Defuzzification

The fuzzy inference process of defuzzification generates the final output. However, the aggregation process must first be evaluated before defuzzification can proceed. The aggregation process is defined as bringing together all of the fuzzy sets of output rules and combining them into one set. The end result of this initial process (i.e., combining the outputs of the rules) then becomes the input for the defuzzification process, the output of which is a single number.

As the term implies, the defuzzification process is the opposite of fuzzification. The input for defuzzification comprises the combined output of each fuzzy rule, leaving the output to emerge as a single number. In the present work, the centroid method has been chosen as the most appropriate defuzzification approach for study purposes. The centroid method proceeds by adding all of the moments around the center area and evaluating them as

$$\text{output } i = \frac{\sum_{k=1}^{N} i\mu_{c(k)}\left(i_k\right)}{\sum_{k=1}^{N} \mu_{c(k)}\left(i_k\right)} \tag{5.5}$$

where N is the total number of rules and $\mu_{c(k)}(i_k)$ denotes the output membership grade for the kth rule with the output subset of i_k.

An illustration of the application of this process in the current study is provided below. Writing the sum in expanded form gives:

$$Q = \frac{LN(G_{QN})(-H_{QN}) + SN(G_{QN})(-H_{QN}) + SP(G_{QP})(H_{QP}) + LP(G_{QP})(H_{QP})}{LN(G_{QN}) + SN(G_{QN}) + SP(G_{QP}) + LP(G_{QP})} \quad (5.6)$$

$$Q = \frac{0.25(5)(-7.5) + 0.65(5)(-2.5) + 0.25(5)(2.5) + 0.35(5)(7.5)}{0.25(5) + 0.65(5) + 0.25(5) + 0.35(5)} = \frac{-1.25}{7.5} = -0.1667$$

where LN, SN, SP, LP are the height of the large negative, small negative, small positive and large positive fuzzy sets, respectively. G_{QN} and G_{QP} represent the widths of each fuzzy set. H_{QN} and H_{QP} represent the moments arm of the centre area for each set.

5.6 Simulation Results

In order to evaluate the proposed algorithm, a series of simulation tests was carried out on an indirect vector controlled IM drive using both a conventional tuned PI controller and an FLC controller. The speed and current responses of the IM have been observed under different operating conditions, such as rapid change in reference speed and step change in load.

Figure 5-4 shows that when the rated load is applied to the IM, both the conventional PI and the proposed control technique are able to track speed commands. It can be observed that motor speed is shown to converge on the desired speed, and they both have the ability to recover quickly from load disturbances. The PI controller has no

overshoot, but it has a lower rise time when compared to the FLC. In general, for this case it is possible to say that the PI has better performance than the FLC for the above reason. From figure 5-7 it is obvious that the FLC drive has overshoot while the PI speed response has fluctuation in the rise time before reaching its steady state and then tracking the reference speed with acceptable performance. Developed motor torque and stator phase currents corresponding to figure 5-4 are shown in figure 5-5 and figure 5-6 respectively. A small disturbance of stator current occurred at the moment of load application, but recovered within a short time. Therefore, the FLC is more robust than the PI controller. Figure 5-8 and figure 5-9 show the torque and phase currents corresponding to the speed response shown in figure 5-7.

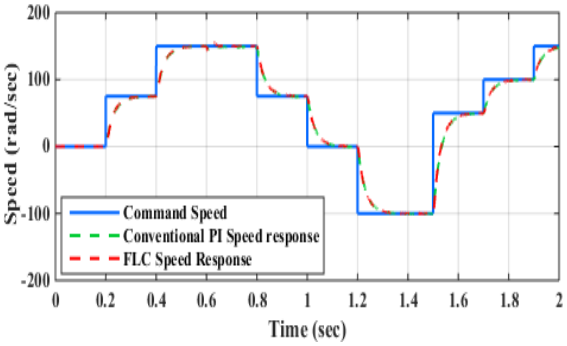


Figure 5-4: Speed tracking response for PI & FLC techniques with rated load torque (2Nm) applied at 0.6 sec.

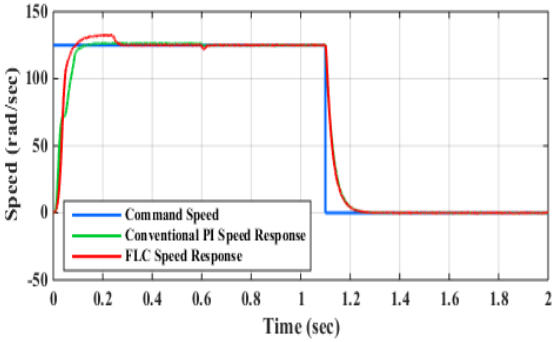


Figure 5-7: Speed tracking response for PI & FLC techniques with load torque (2Nm) applied at 0.6 sec.

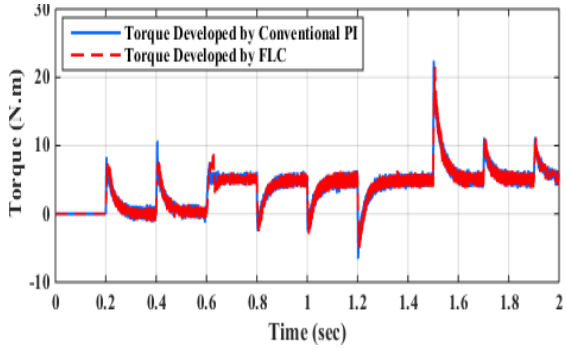


Figure 5-5: Developed motor torque for PI & FLC techniques with load torque (2Nm) applied at 0.6 sec

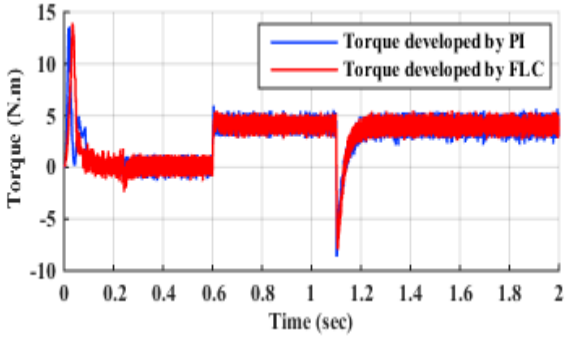


Figure 5-8: Developed motor torque for PI & FLC techniques with Load torque (2Nm) applied at 0.6 sec.

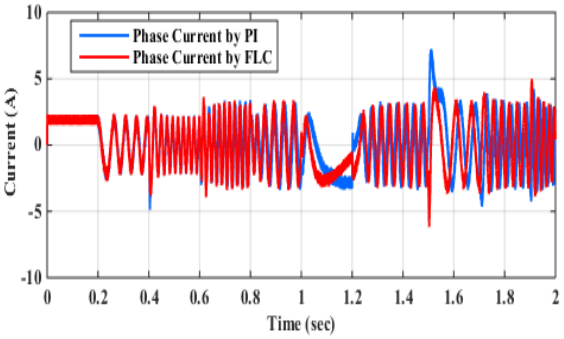


Figure 5-6: Stator phase currents for PI & FLC techniques with load torque (2. Nm) applied at 0.6 sec.

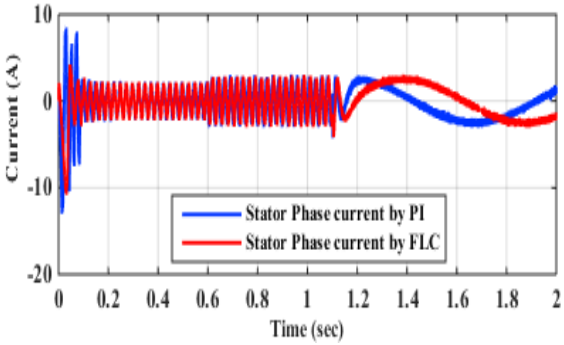


Figure 5-9: Stator phase currents of IM for PI & FLC techniques with load torque (2Nm) applied at 0.6 sec.

5.7 Experimental Results and Discussion

The experimental speed response of the PI based drive is shown in figure 5-10 and the results for the FLC controller are shown in figure 5-11.

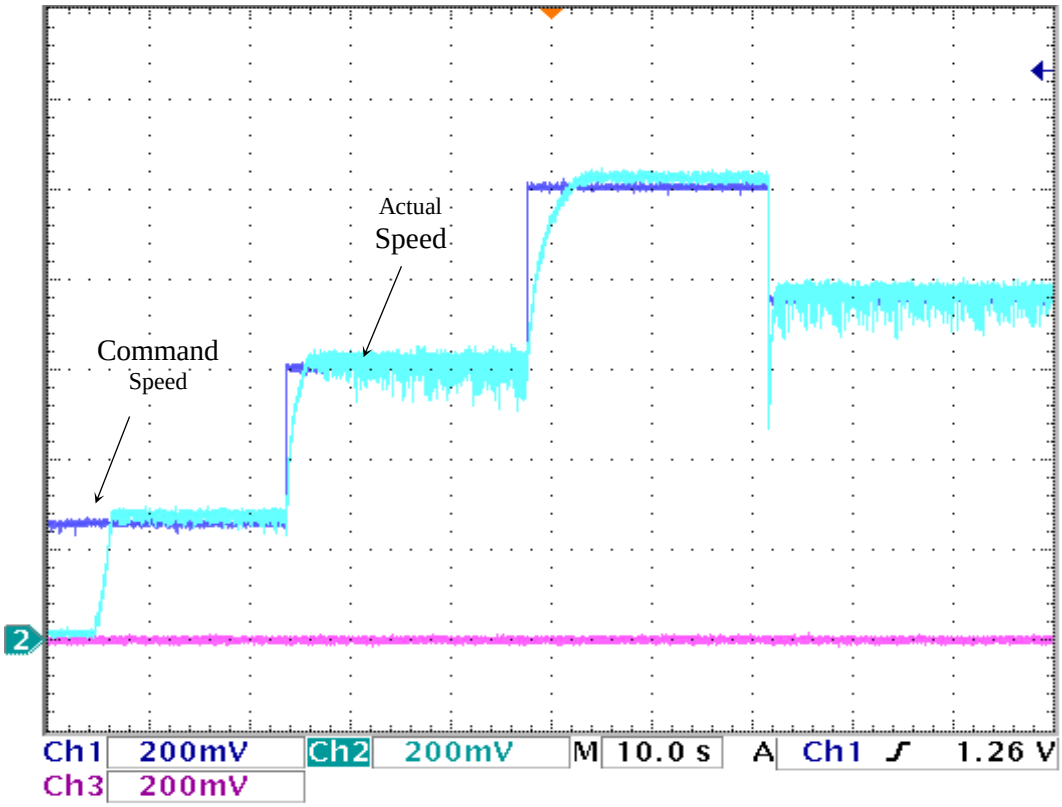


Figure 5-10: Experimental speed response of the drive for the step changes of the command speed based on PI controller, (div=20 rad/sec)

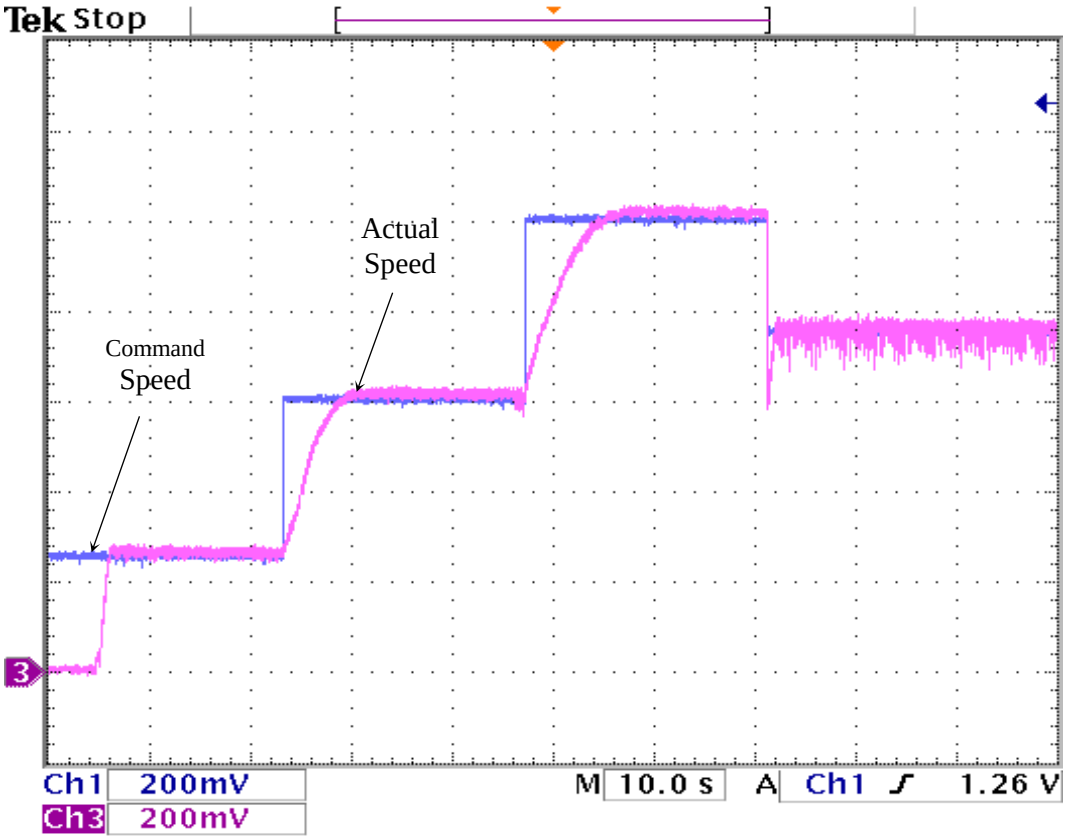


Figure 5-11: Experimental speed response of the drive for the step changes of the command speed based on FLC controller, (div=20 rad/sec)

The results show that the proposed drive system has the ability to follow the reference speed under no load conditions. It should be mentioned that the effect of sudden load impact on dynamic speed response and the evaluation of speed with parameter variation of the FLC based IM drive were performed solely by computer simulation as depicted in figures 5-4 and 5-7. It can be noted for these experimental results that the initial performance of the FLC controller takes time to be able to follow the command speed compared to the performance of the PI controller. This is because the FLC needs time to process its required computation. The results show that under the

conditions presented here, the FLC controller is, in fact, producing control signals in agreement with the reference PI controller.

5.8 Conclusion

To validate the performances of the novel FLC structure, a series of simulation results and a comparative study between the conventional PI and proposed FLC algorithm are provided. Simulation results show that the FLC algorithm, without overlap in the output fuzzy sets, has the ability to reject load disturbances and precisely maintain the desired speed command. Laboratory test results confirm the simulated performances and the validity of this method. Both simulated and experimental results show that the proposed FLC system is robust for IM drive applications. Furthermore, the simplified FLC configuration scheme allows for a significant reduction in computational burden, facilitates ease of implementation and can be used for high performance IM drive control, particularly at very low speeds.

5.9 Extended Applications for the IM Motor Drive

The previous study of the fuzzy logic controller has been extended by adding other fuzzy sets for both inputs, and additional rules that have been applied to the IM. These additions allow an enhanced, accurate performance response control action. These additional rules are indicated below:

1. If the error (E) is negative (N) and the integral of error (I) is negative, then the control signal Q is a negative negative (NN).
2. If the error (E) is negative (N) and the integral of error (I) is zero (Z), then the control signal Q is a negative zero (NZ).
3. If the error (E) is negative (N) and the integral of error (I) is positive (P), then the control signal Q is a negative positive (NP).
4. If the error (E) is zero (Z) and the integral of error (I) is negative (N), then the control signal Q is a zero negative (ZN).
5. If the error E is zero (Z) and the integral of error (I) is zero, then the control signal Q is zero zero (ZZ).
6. If the error (E) is zero (Z) and the integral of error (I) is positive, then the control signal Q is zero positive (ZP).
7. If the error (E) is positive (P) and the integral of error (I) is negative, then the control signal Q is a positive negative (PN).
8. If the error (E) is positive (P) and the integral of error (I) is zero (Z), then the control signal Q is a positive zero (PZ).
9. If the error (E) is positive (P) and the integral of error (I) is positive (P), then the control signal Q is a positive positive (PP).

What follows are the same rules, presented in table format.

Table 5-1: Fuzzy rules

I \ E	N	Z	P
N	NN	ZN	PN
Z	NZ	ZZ	PZ
P	NP	ZP	PP

The fuzzy membership functions and rule decision tables are obtained, in most cases, according to the system behaviour. Consequently, the variation of the scale of the fuzzy sets and thus the rule decision tables depends on the system to be controlled. Once determined, the membership functions and the rule table should satisfy the stable operation of the controlled system. However, any changes that occur in the membership functions is accompanied by changes in the rule decision table and the stability of the system. The present study investigates the performance of the induction motor after using a decision table composed of 9 rules, as well as the effects of altering the breakpoints of the fuzzy sets on the output of the fuzzy logic controller.

It should be noticed that setting the range of speed error input and the interval of membership functions has great impact on the outcome of a fuzzy logic controller. For instance, for the motor drive application illustrated in figures 5-12 to 5-14, setting the input range is determined according to the previous results obtained from PI controller. In addition, triangular and trapezoidal membership functions are formed using straight

lines. These straight line membership functions have the merits of being simple to implement and having fast computation. Furthermore, the fuzzified inputs may take on any value between 0 (no membership in a given linguistic variable's membership function) and 1 (full membership in a given linguistic variable's membership function).
In this example, the speed error occurred at E=10, the integral of error is at I= - 0.6, and the triangular fuzzy set is defined by a lower limit =-15 and an upper limit=+15, as indicated in figure 5-10. However, it's possible to use other values within the range, and it is not necessary for the two limits to be symmetric about zero.

In the first step, fuzzy logic applications requires converting crisp data to fuzzy data using different members of the associated membership functions based on its value.

The red line in figure 5-12 below is a trapezoidal membership function. Its linguistic variable is designated negative (N) and represents a negative slope linear transformation which, for values between –15 and 0, is expressed as:

$$y = \frac{-x}{15} \tag{5.7}$$

Any value less than (-15) will be assigned a 1 and anything above 0.0 a 0.

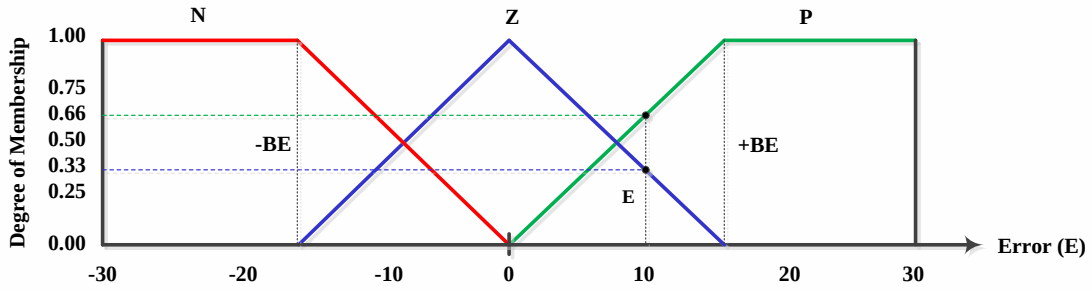


Figure 5-12: Membership Functions (MFs) for Input Error (E)

The blue line in figure 5-12 represents a triangular membership function and its linguistic variable is designated zero (Z). Its negative and positive slope linear transformations are defined as:

$$y = \frac{1}{15}x + 1 \tag{5.8}$$

$$y = \frac{-1}{15}x + 1 \tag{5.9}$$

The green line in figure 5-12 is a trapezoidal membership function. Its linguistic variable is designated positive (P) and represents a positive sloped linear transformation, for values between 0 and +15, which can be described as:

$$y = \frac{x}{15} \tag{5.10}$$

Any value below 0 will be assigned a zero and anything above 15 a 1.

The error E = 10 in this example has two degree membership functions (DMF) , one of which intersects with the zero fuzzy set (Z) and has a DMF equal to 1/3, as

indicated by the dotted blue line. This value can be evaluated by substituting the value of *x*, which in this case is *x* =10, in the equation (5.9).

The other membership degree intersects with the positive fuzzy set (P), and its degree membership function is equal to 2/3, as indicated by the dotted green line. This value can easily be obtained since the full membership function has to be equal to 1, the remaining value of the full membership function is 1-1/3 = 2/3, or it can be obtained from substituting the value of x=10 in the equation (5.10).

E = 10 has no membership at all in the negative fuzzy set (N).

Similarly, if it's assumed the triangular membership function associated with the integral error is defined by a lower limit =-1.7, an upper limit=+1.7, and the integral of error is also assumed to be at point I= - 0.6, then we obtain the graph in figure 5-13.

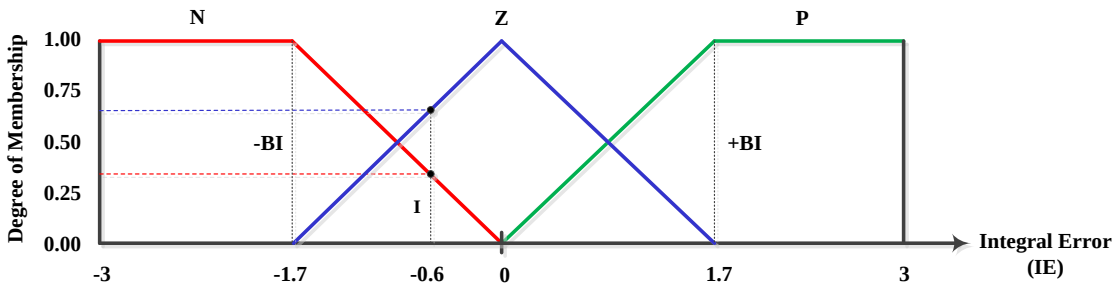


Figure 5-13: Membership Functions (MFs) for Input Integral Error (IE)

Likewise the integral error also has two degree memberships intersecting with different membership functions. One intersects with the negative fuzzy set and its

membership is equal to 0.353, as indicated by the dotted blue line. This value can be evaluated by substituting the value of *x*, which in this case is - 0.6, in the equation (5.11).

$$y = \frac{x}{-1.7} \tag{5.11}$$

The other intersects with the zero fuzzy set, and its membership is equal to 0.647, as indicated by the dotted green line. This value can be evaluated by substituting the value of *x*, which in this case = - 0.6, in the equation (5.12).

$$y = \frac{x}{1.7} + 1 \tag{5.12}$$

In the second step, to begin the fuzzy inference process, it is pertinent to combine the membership functions with the control rules to derive the control output. The rules are considered to be the core of the fuzzy inference process, and those rules are illustrated in table 5-1.

After evaluating the output of each rule, these results should be combined to determine a final result. This task is called inference. The results obtained from individual rules can be combined in a variety ways. However, in this application the minimum criterion is being used for the combination. Figure 5-14 illustrates these membership functions for each of the rule table entries.

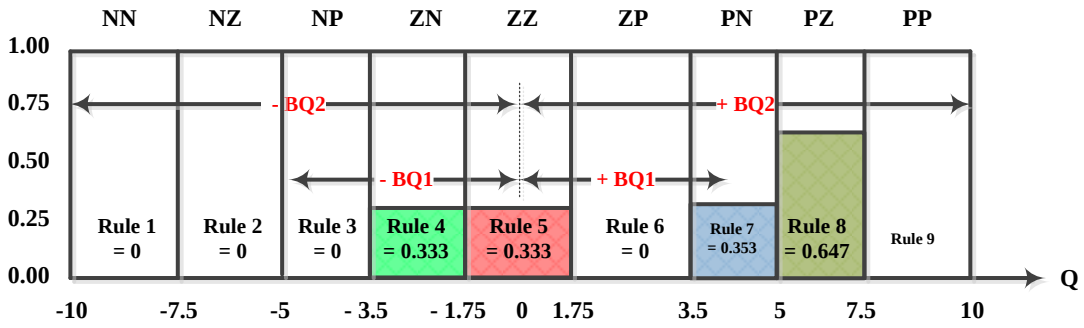


Figure 5-14: Crisp Output Membership Functions

After finishing the process of the fuzzy inference, the outcome result is a fuzzy value; therefore this fuzzy value should be converted to a single crisp to obtain final output. This is what the defuzzification does. For instance, assume that the result in figure 5-10 is the end of the fuzzy inference. In this figure, the colour areas compose the entirety of this fuzzy result. The overall goal is to get a crisp value from the fuzzy result. To calculate the control output, the centre of the area is used to combine the results obtained from each individual rule and gives a crisp control action. The output of each individual rule is calculated and illustrated in table 5-2.

Table 5-2: results obtained from obtained from each rules

Speed error			E= 10	Degree of Membership function (DMF)		
Integral speed error			I= -0.6			
Error break points			BE= 15	NE= 0.0	ZE= 0.333	PE= 0.667
Integral error break points			BI= 1.7	NI= 0.353	ZI= 0.647	PI= 0.0
Rule		boundaries	Centre	width	Mass	First Moments
1	NN=0.0	-10 -7.5	-8.75	2.5	0	0
2	NZ=0.0	-7.5 -5	-6.25	2.5	0	0
3	NP=0.0	-5 -3.5	-4.25	1.5	0	0
4	ZN=0.333	-3.5 -1.75	-2.625	1.75	0.583	-1.531
5	ZZ=0.333	-1.75 1.75	0	3.5	1.166	0
6	ZP=0.0	1.75 3.5	2.625	1.75	0	0
7	PN=0.353	3.5 5	4.25	1.5	0.5294	2.25
8	PZ=0.647	5 7.5	6.25	2.5	1.617	10.11
9	PP=0.0	7.5 10	8.75	2.5	0	0
SUM					3.8954	10.829

$$Q = \frac{10.829}{3.897} = 2.779$$

5.10 Simulation Results

In order to evaluate the proposed algorithm, a series of simulation tests was carried out on an indirect vector controlled IM drive using both a conventional tuned PI controller and the FLC controller. The speed and current responses of the IM have been observed under different operating conditions, such as rapid change in reference speed and step change in load. As is observed in figure 5-15, when the command speed is 120 (rad/sec) and the motor is run without applying load, both PI and FLC controllers are successful in making the motor speed precisely follow the command speed. On the other hand, when the same test is repeated at a speed command of 125 (rad/s), (figure 5-18), only the PI performance response shows fluctuations, while the FLC performance remains the same. Figure 5-16 shows that when the load torque (2 Nm) is applied to the IM, both the conventional PI and the proposed control technique are able to track speed commands. It can be observed that motor speed converges with the desired speed, and they both have the ability to recover quickly from load disturbances. Fluctuation occurs in the PI performance response compared to the smooth performance obtained by the FLC, which further has a lower rise time. Therefore, the FLC performs better than the PI for the above reasons. From figure 5-18, it can be seen that both the PI and the FLC are similar in their ability to track speed. Despite the slight dip that occurs at the moment of applying load, both FLC and PI techniques quickly recover from the load disturbance. The change in speed (observed in figure 5-16) induces changes in torque and current, as can be seen in figure 5-17 and figure 5-21, respectively. Similarly, changes in speed seen in figure 5-19 cause the corresponding changes in torque and current demonstrated in figures 5-20 and 5-22, respectively.

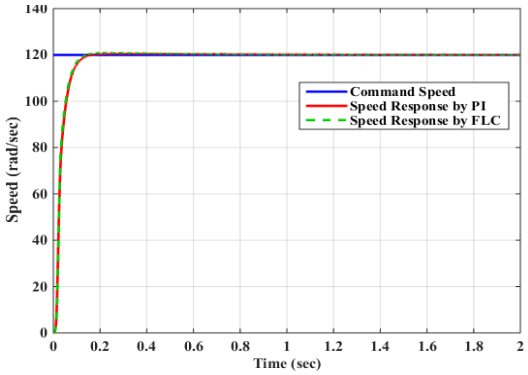


Figure 5-15: Speed tracking responses at no load, 120 rad/sec

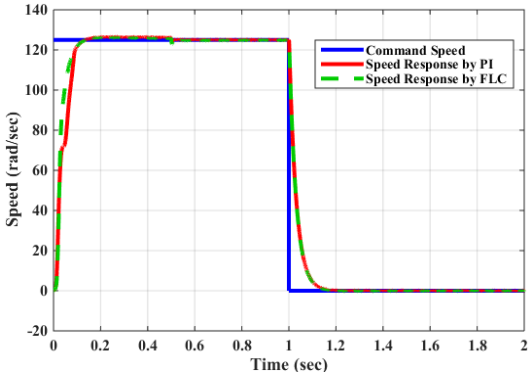


Figure 5-16: Speed tracking responses at load torque (2 Nm), 125 rad/sec

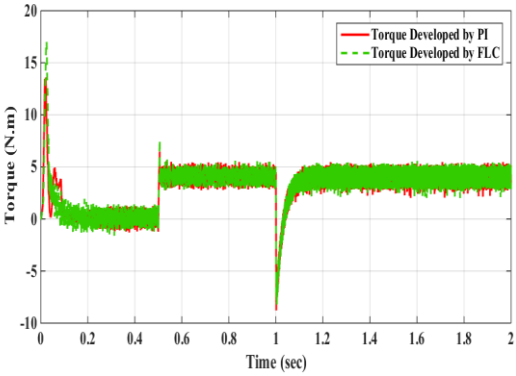


Figure 5-17: Torque developed at rated load, corresponding to the speed given in figure 5-16

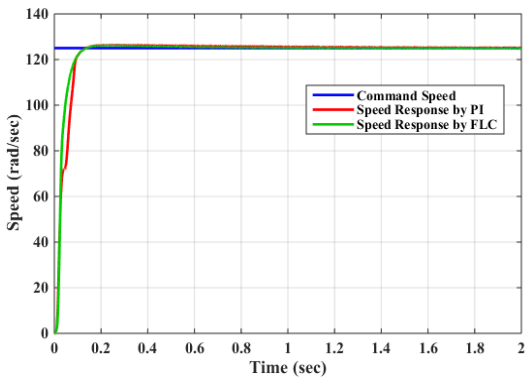


Figure 5-18: Speed tracking responses at no load, 125 rad/sec

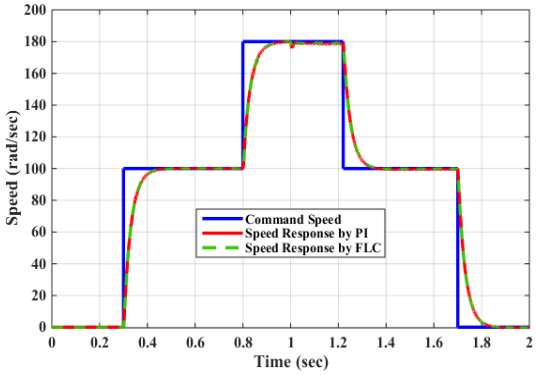


Figure 5-19: Speed tracking responses at load torque (2 Nm), 180 rad/sec

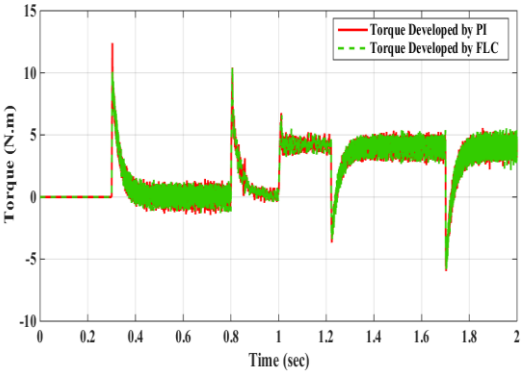


Figure 5-20: Torque developed at rated load, corresponding to the speed given in figure 5-19

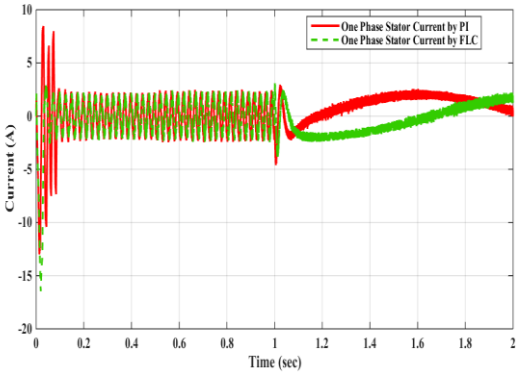


Figure 5-21: Stator phase currents corresponding to the speed given in figure 5-16

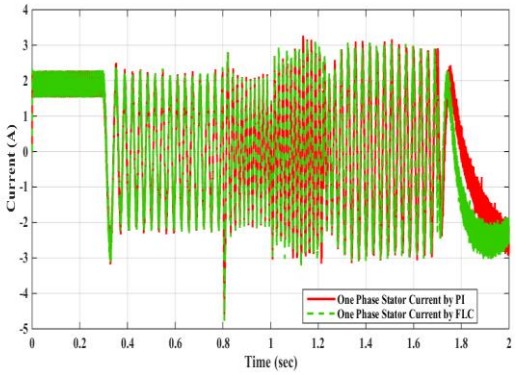


Figure 5-22: Stator phase currents corresponding to the speed given in figure 5-19

5.11 Experimental Results and Discussion

To verify the efficiency of the proposed algorithm, a series of practical tests were performed on the motor that is under investigation. The results obtained from these tests were compared with those obtained by the conventional PI control. Figures 5-23, 5-24 and 5-25 demonstrate the speed performance response of a motor using the FLC, where different speed commands are applied. In contrast, figures 5-26, 5-27, and 5-28 illustrate the speed performance response resulting from the PI control. The results show that the proposed drive system has the ability to follow the reference speed under no load conditions. It should be mentioned that the effect of sudden load impact on dynamic speed response and the evaluation of speed with a parameter variation of the FLC based IM drive were performed solely by computer simulation, as depicted in figure 5-16 and 5-19. It can be noted from these experimental results that the performances of the FLC and PI controllers precisely follow command speed. The results show that under the conditions presented here, the FLC controller is in fact producing control signals that correspond well with the reference PI controller.

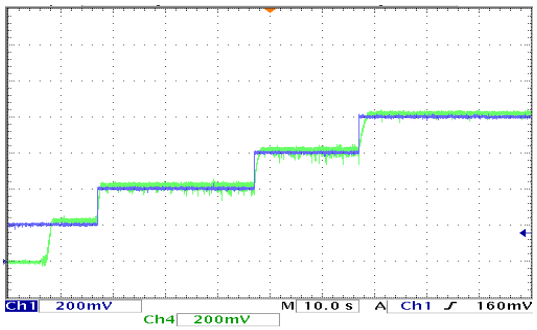


Figure 5-23 Experimental speed response of the drive for the step changes of the command speed based on FLC,(div=20 rad/sec)

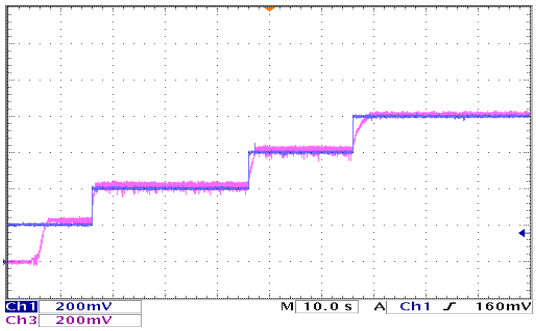


Figure 5-26: Experimental speed response of the drive for the step changes of the command speed based on PI controller,(div=20 rad/sec)

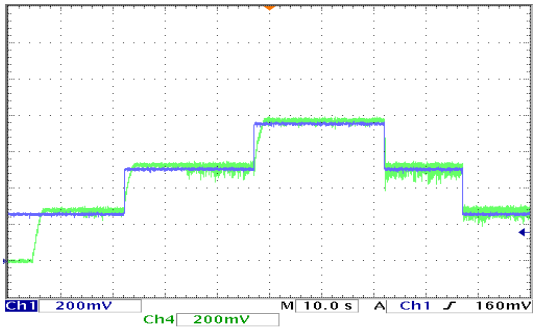


Figure 5-24: Experimental speed response of the drive for the step changes of the command speed based on FLC,(div=20 rad/sec)

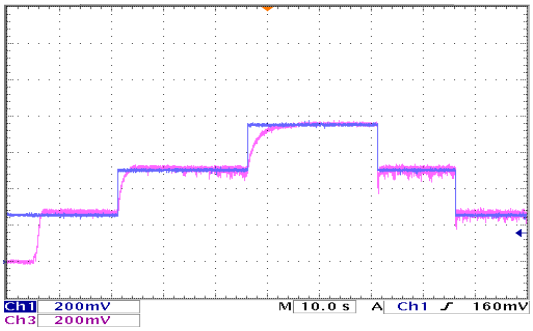


Figure 5-27: Experimental speed response of the drive for the step changes of the command speed based on PI controller,(div=20 rad/sec)

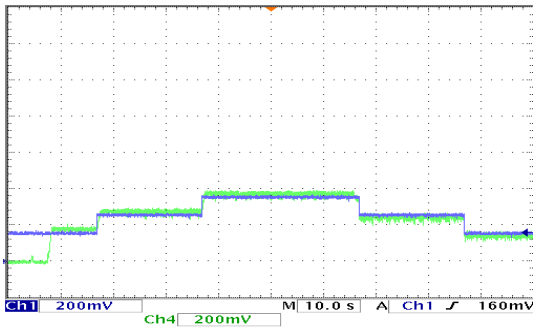


Figure 5-25: Experimental speed response of the drive for the step changes of the command speed based on FLC,(div=20 rad/sec)

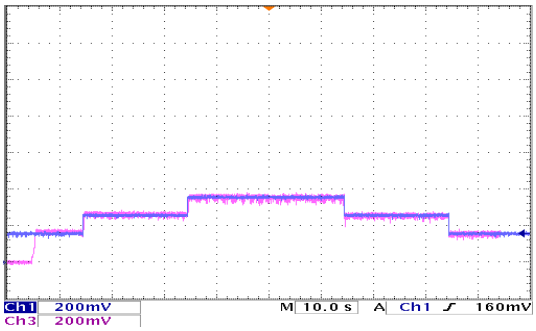


Figure 5-28: Experimental speed response of the drive for the step changes of the command speed based on PI controller,(div=20 rad/sec)

5.12 Fuzzy Sets Breakpoints Effect

There is no clear guideline reported for selecting the values of input and output fuzzy sets' breakpoints, whose values can be determined by trial and error until the performance is improved. To illustrate the impact of changing the breakpoint on fuzzy sets, figures 5-29 and 5-30 demonstrate the effect of changing the breakpoints of output fuzzy sets (BQ1, BQ2) while fixing the remaining breakpoints of the input fuzzy set (BE, BI). It is clear from the figure that the optimum values selected for (BQ1, BQ2) which provide high performance, are 10 and 40 respectively. Figures 5-31 demonstrates the impact of changing output fuzzy sets' breakpoints on torque performance. In addition, slight variation of the breakpoint of the input fuzzy sets had a negligible effect on the output of the controller (speed and torque) while maintaining the breakpoint of the output fuzzy set because the relationship between the input and output is smooth.

Figure 5-32 is presented in order to analyze and determine whether there is any relationship between the breakpoints, fuzzy controller signal (Q) and output performance response.

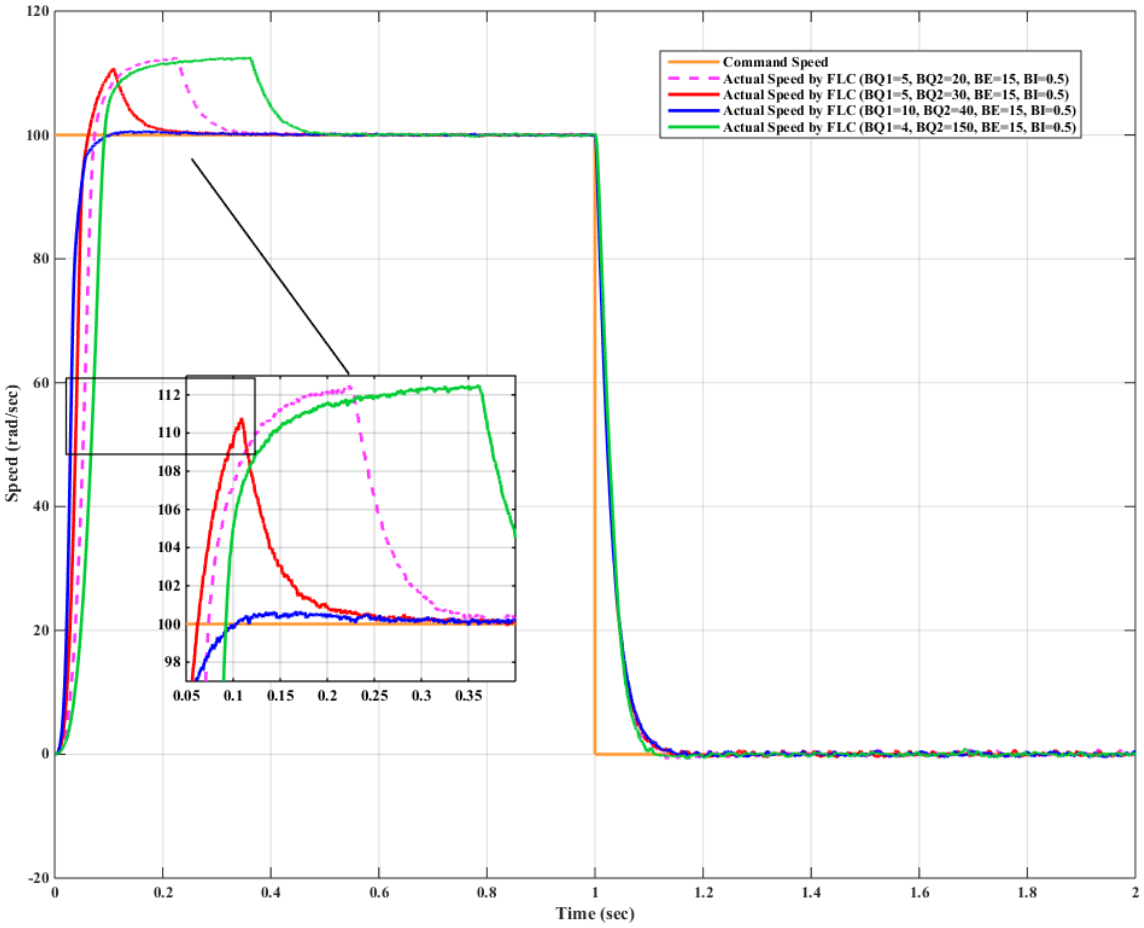


Figure 5-29: Impact of changing output fuzzy sets' breakpoints on speed performance

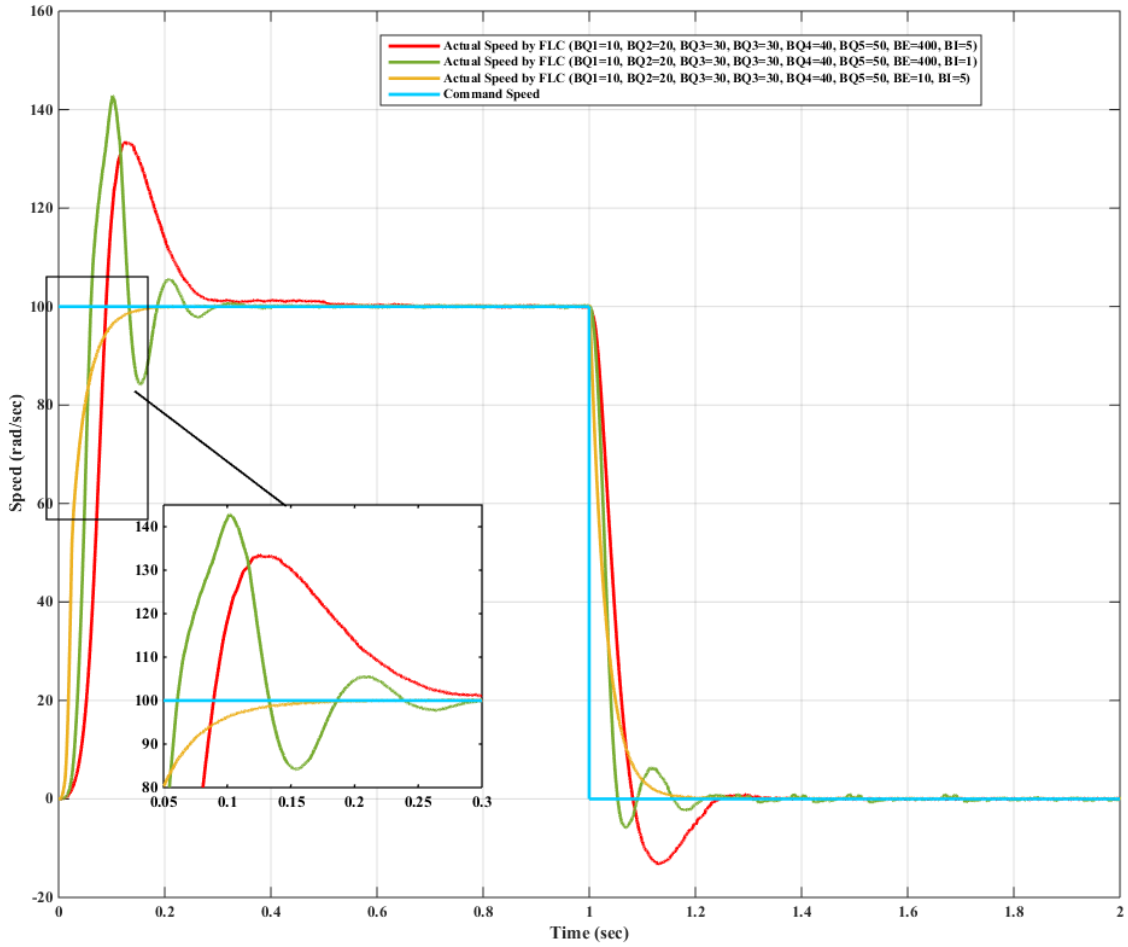


Figure 5-30: Impact of changing output fuzzy sets' breakpoints on speed performance

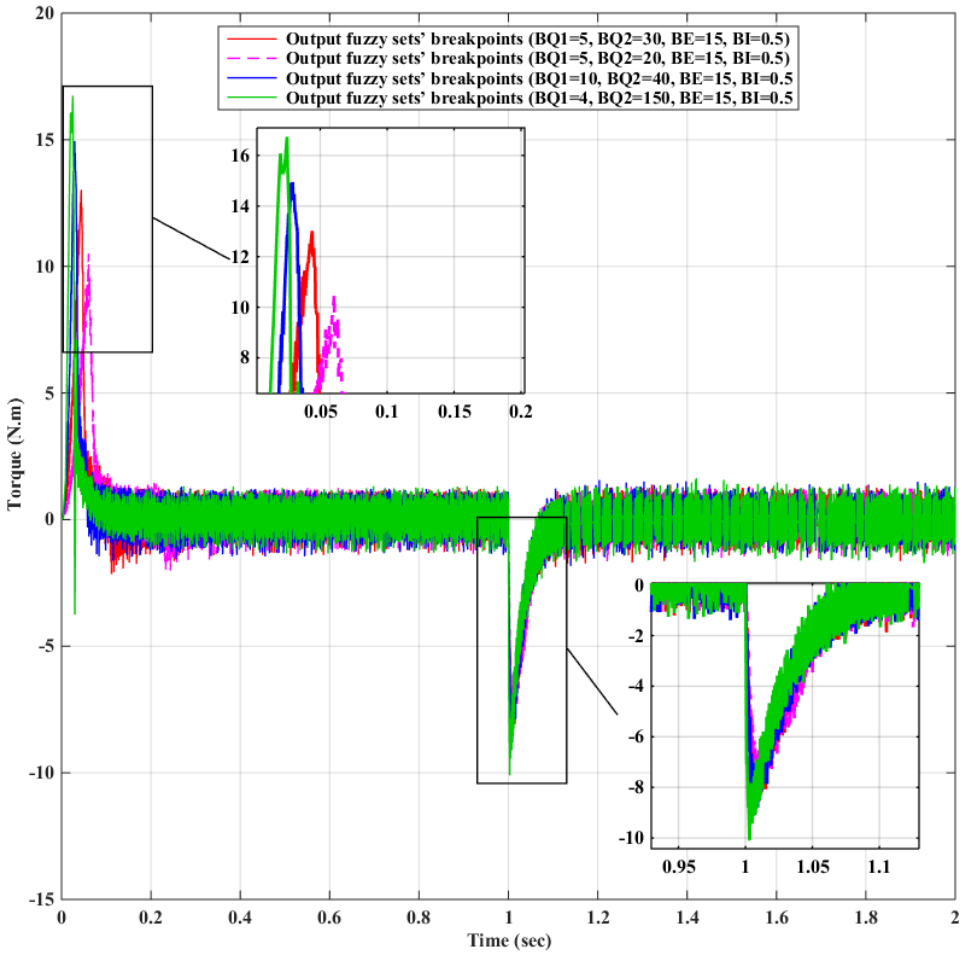


Figure 5-31: Impact of changing output fuzzy sets' breakpoints on torque performance

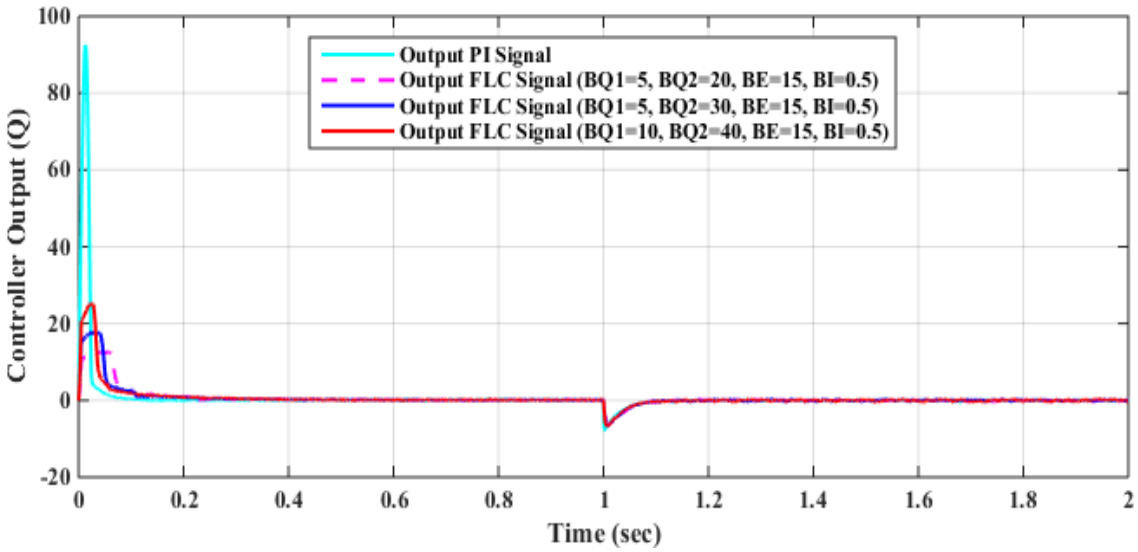


Figure 5-32: The resulting corresponding outputs controller signals from changing output fuzzy sets breakpoints

It is worth mentioning that in certain circumstances, one may not be aware that the break points in a simplified fuzzy logic controller can be related to the PI controller gains. For the same application using 4 rules (NN, NP, PN and PP) consider these cases where the integral input is 100% positive and 0% negative. As the error input E varies from 100% positive to 100% negative between the break points, $+BE$ and $-BE$, the output Q varies from +1.5 BQ (the output breakpoint)] to -0.5 BQ along a straight line as shown in figure 5-29, figure 5-30 and figure 5-31.

In this situation, figure 5-33 demonstrates the cases when the integral is fixed so it has 100% positive membership and 0% negative membership. Meanwhile, the error has 0% negative membership and 100% positive membership, 50% negative membership and 50% Positive membership and finally, 100% negative membership and 0% positive membership.

At the same time, figure 5-34 demonstrates the cases when the integral is set at 50% positive and 50% negative, while the error is 0% negative and 100% positive, 50% negative and 50% positive and finally 100% negative and 0% positive.

Correspondingly, figure 5-35 demonstrates the situations when the integral is 0% positive and 100% negative, while the error 0% negative and 100% positive, 50% negative and 50% positive and 100% negative and 0% positive.

Each case within the three cases constructs three points which fall on the straight line as shown in figure 5-36. However, the three constructed parallel lines passing through these points have the same slope (BQ/BE).

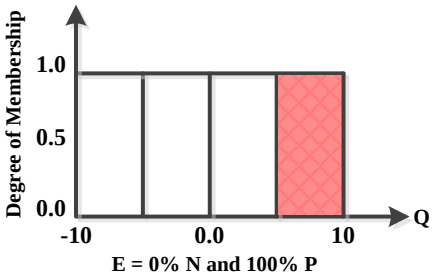
The slope of the line is $2BQ/2BE$ and is similar to the proportional gain K_p in a conventional PI controller. + BQ is equal to the saturation level on Q axis. This gives an equation for BQ. The gain K_p will therefore also be known. This leads to an equation for BE, which is expressed as follows:

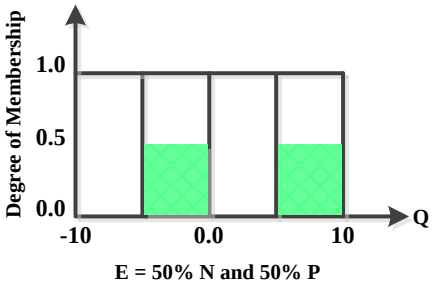
$$BQ = QS \tag{5.13}$$

$$BQ = QS \tag{5.14}$$

$$BE = \frac{Qs}{K_p} \tag{5.15}$$

where QS is the saturation limit





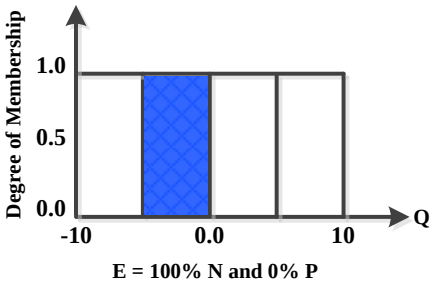
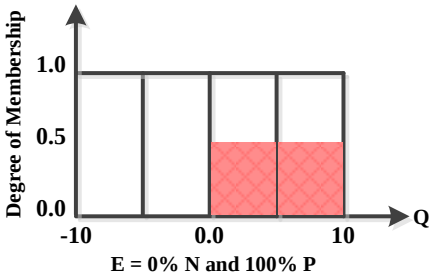
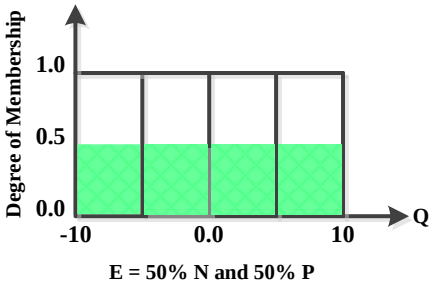


Figure 5-33: Output membership function when integral is set at 0% N and 100% P, while the error is changed





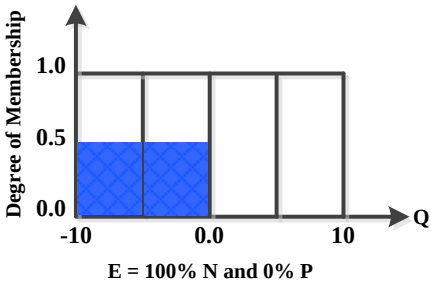


Figure 5-34: Output membership function when integral is set at 50% N and 50% P, while the error is changed

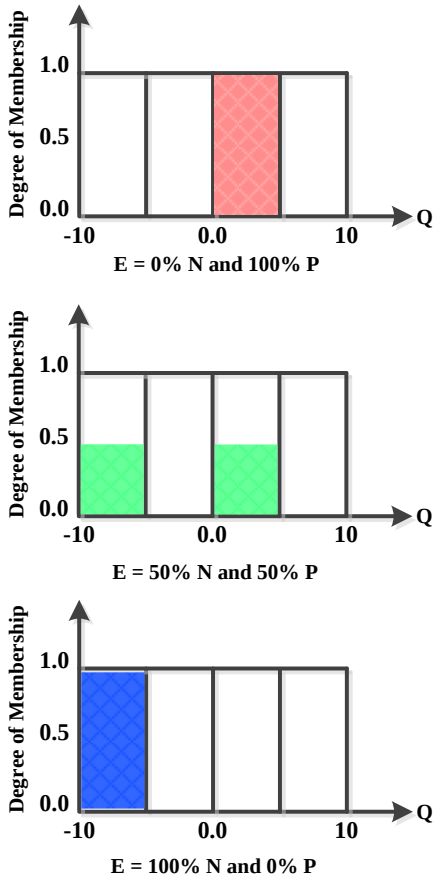


Figure 5-35: Output membership function when integral is set at 100% N and 0% P, while the error is changed

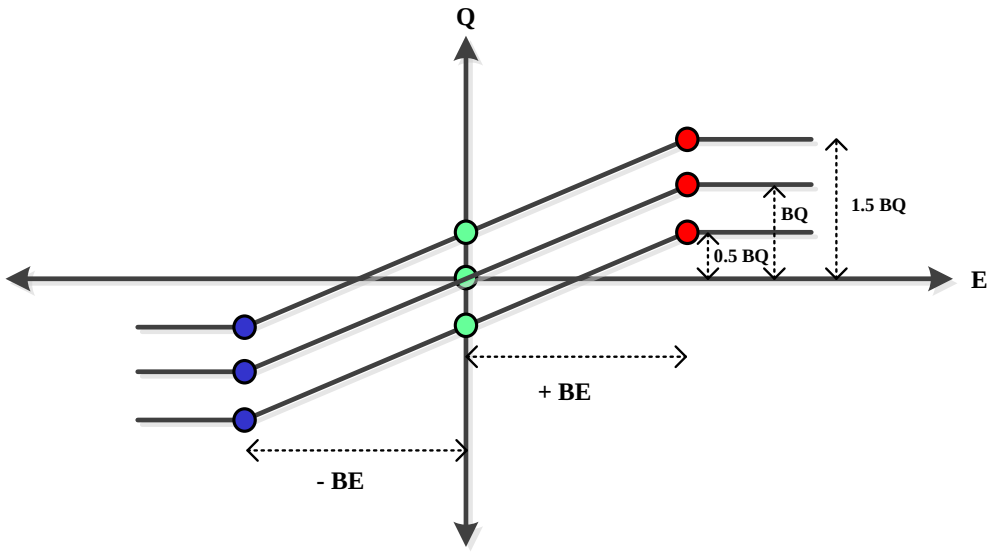


Figure 5-36: the relation between output fuzzy logic against the error

5.13 Chapter Summary

This chapter presents a simplified design of the Fuzzy Logic Controller. The main discrepancy between this present design and that of the previously used controller is the lack of overlap in the output membership function, which diminishes the computational burden experienced to a minimum. By keeping this burden at a low level, computers with limited processors and memory storage can be employed to operate the control algorithm in real-time at both higher motor speeds and lower currents. This technique can be applied using two separate strategies, a four-rule method and a nine-rule method, and it is compared by applying them to the same induction motor for field-oriented speed control. Several tests are conducted under various operating conditions in real-time to confirm the effectiveness of the techniques, and they are compared with the conventional PI controller in simulation and experimentation. It is found that the proposed FLC can be an appropriate alternative to the conventional PI controller to achieve high performance in IM drive systems.

Chapter 6.

Vector Control of Induction Motor Using Neural Networks

This chapter introduces an Artificial Neural Network (ANN) based speed controller using the field oriented control (FOC) scheme. For many years, conventional linear controllers such as PI and PID have found widespread application to Induction Motor (IM) drives, due to their simplicity and ease of implementation in real time. Nevertheless, because of uncertainties such as saturation, temperature variation, sudden changes in command speed and load disturbances, these controllers are very sensitive to parameter variations. Moreover, it is difficult to precisely tune the controller parameters for both online and off-line implementations. Therefore, these types of controllers are not always suitable for high performance applications. Consequently, researchers have recently applied intelligent controllers for motor drive applications for various reasons. The main advantages of intelligent controllers are that the design of intelligent controllers is independent of the system parameters, and they can also handle system nonlinearity.

6.1 Introduction

In order to describe the performance of real systems and to formulate mathematical models which will represent them, algebraic and differential equations are employed. However, in order to imitate the system operation, accurate knowledge of the dynamic system is required to predict and apply numerical techniques. However, physical systems most often involve complicated nonlinear relations, and it is a challenge to model these using conventional techniques. For example, a three phase induction

motor combined with the drive system can be considered as a complex nonlinear system due to the various issues accompanying its dynamics. Therefore, finding a way to controlling IMs has prompted extensive research attention. The current study proposes the application of Artificial Neural Networks (ANNs) to control IMs in order to obtain desired performance.

Thus, the need arises for increasingly advanced control techniques, in order to ensure appropriate and economic handling of their nonlinear dynamics. The ANN is characterized by learning adaptation and nonlinear mapping capabilities that have made them useful for predicting, modelling and controlling complicated, uncertain systems for which traditional techniques have been inadequate. Using ANNs has become of considerable interest to scientists, because of their many advantages compared with conventional algorithmic techniques. Some of these advantages include capability of training, simplicity of construction, ability to approximate nonlinear functions, ability to withstand network distortion, and capacity for functioning without an accurate mathematical model [126-135]. A further advantage of ANNs is that, because they are able to approximate a wide range of nonlinear functions to any desired degree of accuracy, they can be employed in the identification and control of nonlinear systems. For instance, in ac motor application, stator voltages and currents can be used as inputs to the network for speed or rotor resistance identification [133]. Further, stator currents and voltages, as well as speed, can be used as input to the network to estimate developed torque in still other applications [134], while rotor flux and torque can be estimated by feeding stator currents as input into the neural network. In addition, ANNs can be employed to control inverters when the gate pulse is generated based on error obtained

by comparing the actual and reference stator currents. A detailed review of relevant literature is provided in chapter one. Neural networks can be divided into two principle categories. The first one, called a fixed network, has weights that cannot be changed $\frac{dW}{dt} = 0$. In such networks, the weights are fixed a priori according to the problem to be solved. The second, called an adaptive network, has weights which are able to be changed $\frac{dW}{dt} \neq 0$. The network learning process of artificial neural networks during training consumes time. The execution time for training, particularly off-line training, can be high, depending upon the detailed understanding of motor performance for a particular drive system. In addition, when the parameters outside the training set are incurred in the system, ANN may exhibit poor drive performance. To address these shortcomings, the first section of this chapter uses the fixed network to estimate the synaptic weights required by the network. These weights are used in the current application of the ANN scheme based speed drive for indirect field oriented control of IMs in fixed networks, to replace the traditional PI controller. This will speed up the convergence rate by lowering the execution time, and reduce the burden and expense on the processor required for carrying out all computations.

6.2 Artificial Neural Network Model

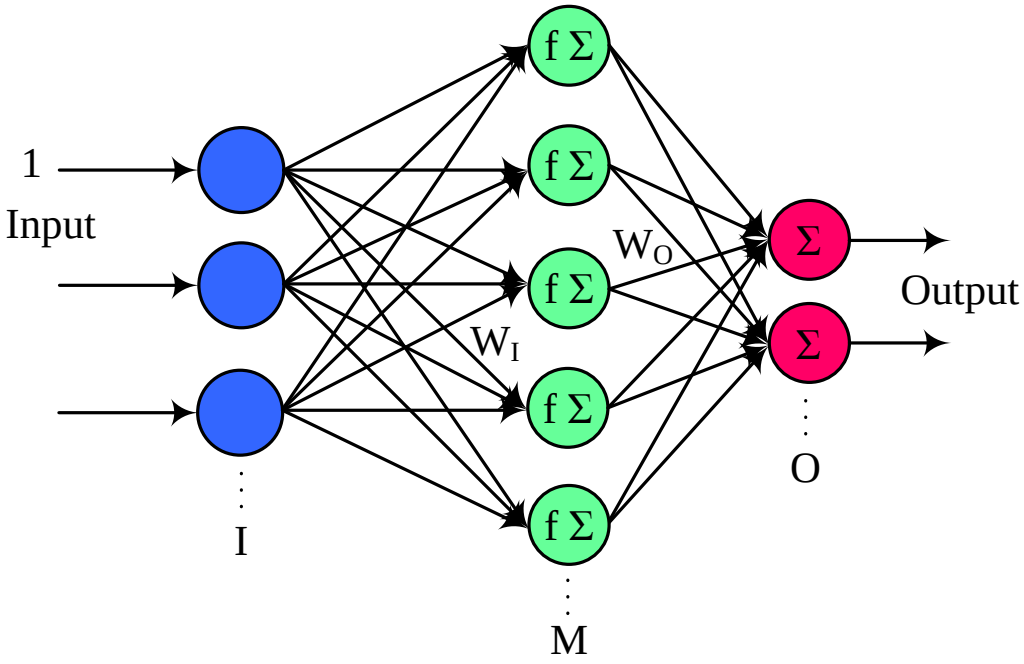


Figure 6-1: Multilayer feed-forward artificial neural network

In a network model, neurons can be interconnected by using their output as inputs to other neurons. The interconnection of neurons forms the layers of a neural network. In figure 6-1, a simple mapping of a neural network is presented. As can be seen, the network comprises an input along with middle or hidden layer, and an output layer. The 2-3-1 ANN structure was used just as an illustration. Many neurons could have been selected. The map corresponding to 2-3-1 ANN structure has dead-band in one input direction, and it also has control signal saturation. In this work 2-2-1 ANN structure with a neuron dedicated to each input was used to make it match the conventional PI controller. Both the input and hidden layers hold one bias

neuron with input unity. The processing of the summed inputs into the hidden layer neurons proceeds through a nonlinear activation function, such as a sigmoidal function. The sigmoid or S shape applied by the majority of researchers in the field serves as an activation function, as follows:

$$f(i) = \frac{1}{\left(1 + e^{-i}\right)} \tag{6.1}$$

This particular function is also employed in the present thesis.

Squashing represents the transferring or "firing" action of biological neurons during the natural process. Information can then flow in a forward manner through the network, from input to output. For example, the neuron depicted in figure (6-2) is comprised of four inputs with weights:

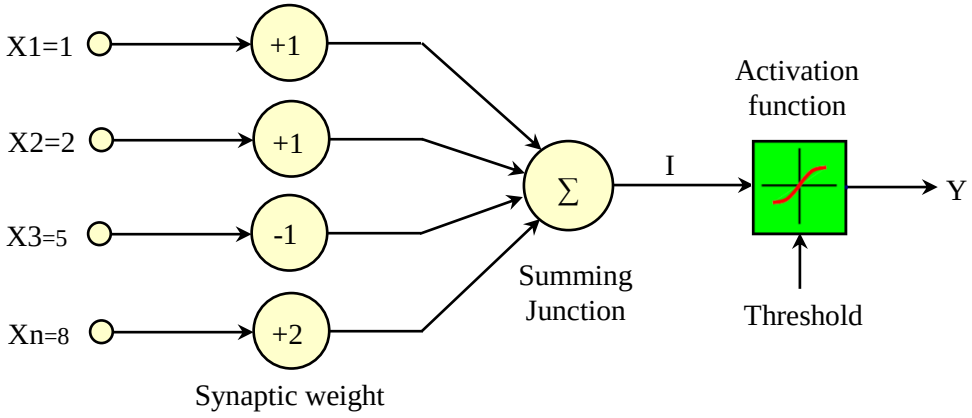


Figure 6-2: Neuron structure of the example

Before reaching the activation function step, the network's output, I, is

$$I = X^T.W = [1 \quad 2 \quad 5 \quad 8].\begin{bmatrix} 1 \\ 1 \\ -1 \\ 2 \end{bmatrix} = 14$$

Then, with a binary activation function, the output of the neuron is

Y (threshold) = 1.

In the network illustrated in figure (6-3), which features a single input, I, along with a single output, O, the following mapping equation for a single neuron case is used

$$O = W_{OB} + W_{OM} f\left(W_{IM} I + W_{IB}\right) \tag{6.2}$$

where O represents the output of the neural network controller, W_{OB} is the output bias, W_{OM} is the output scaling factor, W_{IM} is the input scaling factor, W_{IB} is the input bias, and the neuron I represents the proportional input signal [135].

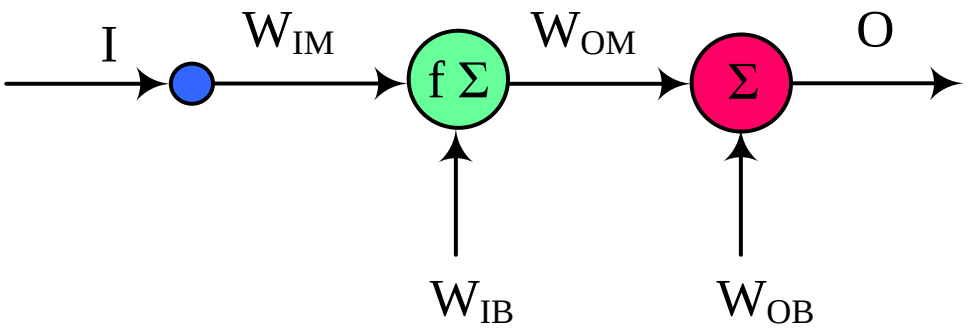


Figure 6-3: Neuron network with a single input and output.

The above equation illustrates how the weights can be applied in an O/I plot in order to scale and shift the sigmoid function vertically along O and horizontally along I .

In figure (6-4), W_{OB} moves the sigmoid vertically and W_{IB} moves it horizontally, while W_{OM} scales it vertically and $1/W_{IM}$ does so horizontally.

From this type of set-up, a map can be formed by joining together, in patchwork fashion, the shifted and scaled sigmoid functions. This lends the process a local flavour similar to finite elements and thus enables very good fits through the allowance of sharp local gradients in O/I connections.

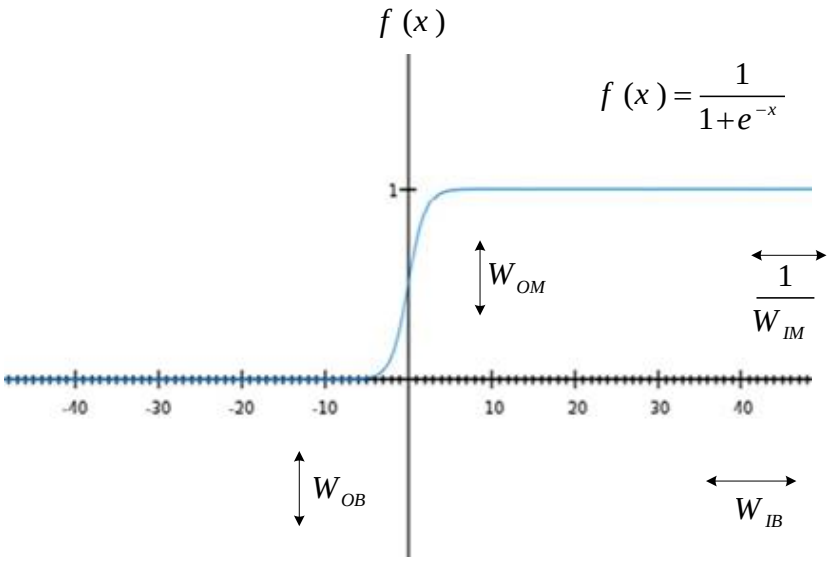


Figure 6-4: Sigmoid Activation Function

Figure (6-5) depicts slightly more complex neural networks. Each of these networks has two inputs (i, j) and one output (O). As illustrated in the following equation, the individual networks are formed from the three neurons X & Y & Z:

$$O = W_{ob} + W_{ox} f\left(W_{ix} i + W_{jx} j + W_{xb}\right) + W_{oy} f\left(W_{iy} i + W_{jy} j + W_{yb}\right) + W_{oz} f\left(W_{iz} i + W_{jz} j + W_{zb}\right) \tag{6.3}$$

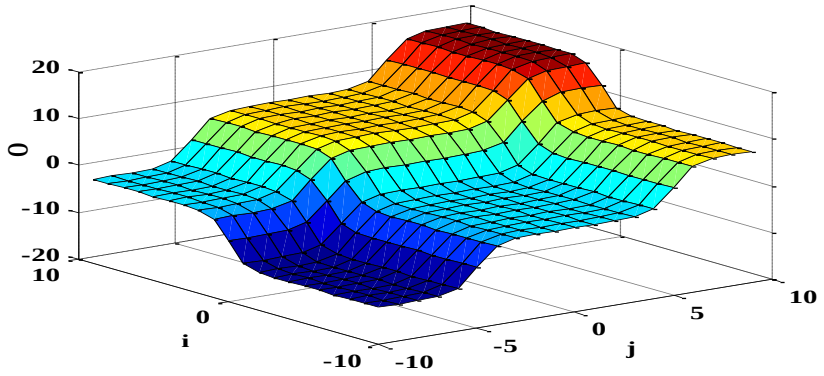


Figure 6-5: Neural network map

The map shown in figure (6-5), generated by equation (6.3), when $W_{ix} = 0,\ W_{iy} = 0,\ W_{jz} = 0$, is a PI type of controller map. It is clear from the map of figure (6-5), each net has sharp local gradients. Even though the standard polynomial fit finds these kinds of gradients challenging, such local maps can nonetheless be created by the human brain.

The aim of the motor speed case is to find the weights W_O and W_I that make the net speeds (O_N) match the corresponding target speeds (O_T). The majority of neural network codes apply the concept of steepest descent iteratively to trim and balance the weights after an initial random distribution. The error provided in Equation (6.4) helps determine the motor speed:

$$e = O_N - O_T \tag{6.4}$$

where $O_{\mathrm{N}} = \sum W_{\mathrm{O}} M$, and $\boldsymbol{M}$ is a hidden layer neuron squashed output.

The aim here is to reduce the square of the error (e^2). Thus, for each input/output training pair, the squared error depends on W_o and M being fed into it. For small shifts in W_o and M, alterations to e^2 can be formulated as follows:

$$\Delta e^2 = \sum \left(\frac{\partial e^2}{\partial W_o} \right) \Delta W_o + \left(\frac{\partial e^2}{\partial M} \right) \Delta M \tag{6.5}$$

Differentiation of $M = f\sum(W_I I)$ with I fixed gives the connection $\Delta M = \sum \left(\frac{\partial M}{\partial W_I} \right) \Delta W_I$, so Equation (6.5) therefore becomes

$$\Delta e^2 = \sum \left(\frac{\partial e^2}{\partial W_o} \right) \Delta W_o + \sum \left(\frac{\partial e^2}{\partial M} \right) \sum \left(\frac{\partial M}{\partial W_I} \right) \Delta W_I \tag{6.6}$$

According to the steepest descent, $W_{O(NEW)}$ and $W_{I(NEW)}$ are given by Equations (6.7) and (6.8), respectively:

$$W_{O(\mathrm{NEW})} = W_{O(\mathrm{OLD})} - K \left(\frac{\partial e^2}{\partial W_O} \right) \tag{6.7}$$

$$W_{\mathrm{I(NEW)}} = W_{\mathrm{I(OLD)}} - K \left(\frac{\partial e^2}{\partial M} \right) \frac{\partial M}{\partial W_I} \tag{6.8}$$

where the small value of K promotes iteration stability. Hence, if the squared error is so sensitive to a certain weight that the weight causes it to increase, it is necessary to slightly decrease the weight during the training stage. Similarly, if reducing the weight leads to an increase in the error, the weight must thus be increased. Thus, when stable, this iteration, if fed numerous times in an input/output training set with each data point,

usually reduces the global e^2 versus weight bowl in hyperspace towards the global minimum, as is shown below in figure (6-6).

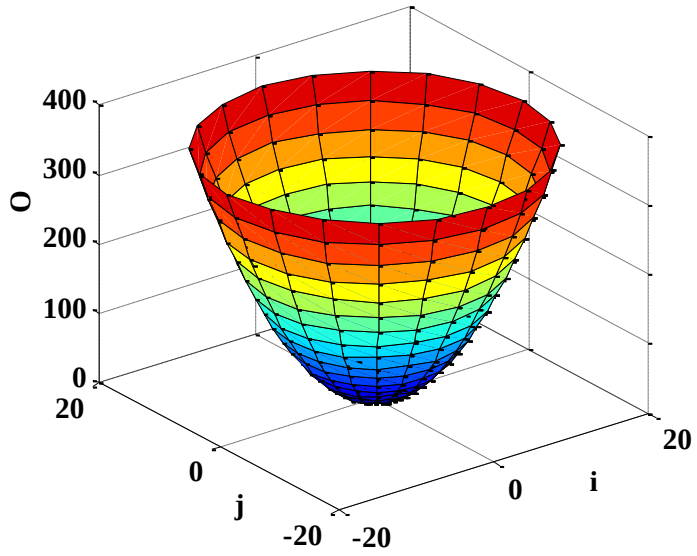


Figure 6-6: Global e^2 versus weight bowl

To prevent the convergence of local minima, the iterations can be reordered by adding noise to the weights. Thousands of iterations are usually needed to achieve the global minimum. Alternative training methods (e.g., approaches that simulate genetics processes) are faster but less well-developed. Genetics-based formulations run numerous parent nets at the same time, permitting only those nets that create the best fits to mate by exchanging weights and thus generating offspring nets. Of these, the very best of the offspring nets are randomly mutated, after which they are combined with the best parent nets to produce the next generation of parents.

6.3 Artificial Neural Network Based Speed Controller for IM Drive

A high performance IM drive system is characterized by an efficient speed controller, causing the drive to follow the command speed under a wide range of varying operating conditions. This tracking of command speed is achieved rapidly and precisely regardless of system parameter variations and unexpected load disturbance. Conventional controllers suffer from the fact that an accurate system model of the IM is necessary for their design. This model is difficult to obtain due to necessary approximations, parameter variations, saturation effects, load disturbances, temperature variations, noise, etc. The consequence is that these controllers typically perform well only over the narrow range of specific conditions for which they were designed.

Traditional artificial neural network-based motor controllers require extensive off-line training, incur high computational burden (time) and rely on the past characteristic behaviour of the motor drive for a specific system. However, drive behaviour is unpredictable when parameters outside the training set are encountered.

In applying the above situations to this research, ANN is used only to mimic PI controllers, which means suitable fixed weights are required to precisely follow the desired speed. The complete drive system consists of an ANN-based speed controller, a voltage source inverter and a squirrel cage induction motor. The ANN-based controller consists of the input layer, containing two input neurons, one hidden layer, containing three neurons, and the output layer, containing one neuron. This type of structure was chosen because it is the only one which allows the creation of a controller with deadband and saturation in one input direction and saturation in the other direction. This contrasts with other structures which are not as efficient. For example, one neuron with one input

would have allowed the creation of the controller with saturation in one input direction, like proportional or integral, while two neurons with one input would have allowed the creation of a controller with deadband and saturation in one input direction.

For comparative evaluation purposes, a PI controller-based IM drive system has also been employed and tested. Furthermore, extensive simulations have been conducted so as to assess the performances of both drives at different dynamic operating conditions. Finally, the simulation results are presented and discussed.

6.4 Vector Control of the IM with an Artificial Neural Network Controller

The indirect vector control scheme of the IM drive, as described in Chapter 3, has been implemented, with the ANN fulfilling the role of the speed controller. This system is illustrated in figure 6-7.

The speed sensor on the rotor shaft relays rotor speed information; it is added to the slip in order to calculate rotor position. Along with this and the desired command speed, the input calculator uses phase current readings to determine the appropriate reference command torque. Command speed and speed error are then fed to the ANN-based speed controller. The ANN controller generates the appropriate command torque from which the i_{qs}^{*} component can be produced. Then the correct i_{ds}^{*} component is determined from the flux command, which is assumed to be constant at the saturation level, and applied to the motor stator via current transformations and then to the hysteresis current controller through a voltage source inverter.

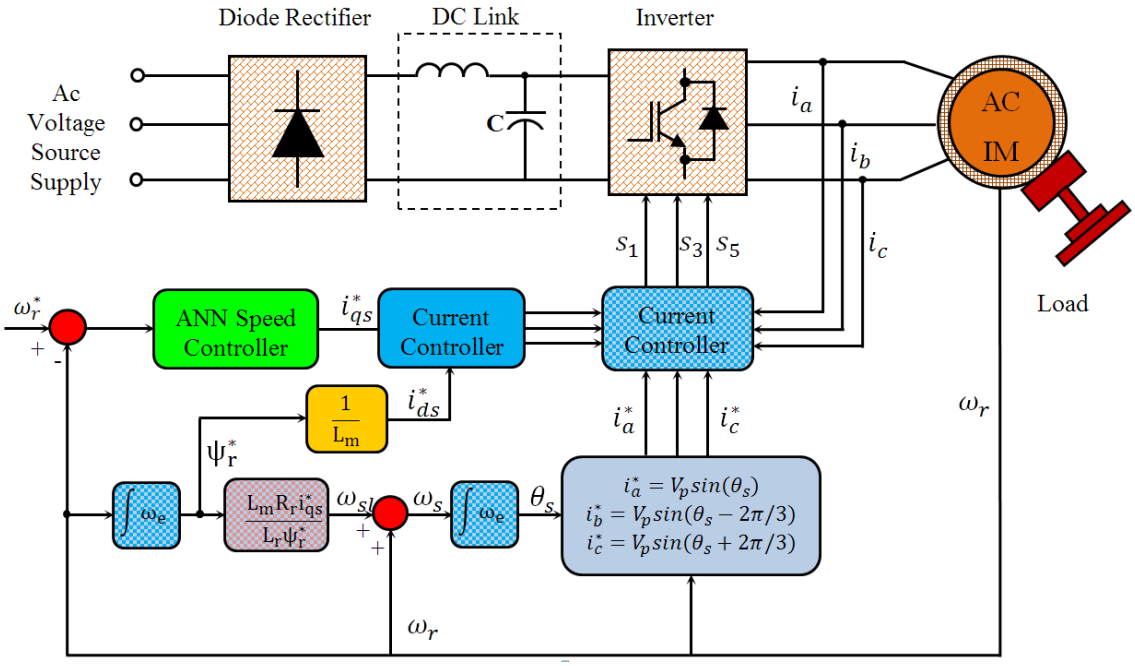


Figure 6-7: Illustration of vector control scheme using ANN

6.4.1 Artificial Neural Network Architectures

The number of nodes in the input layer is dependent on the number of inputs to the ANN. Since in this application the fixed neural network is used to mimic the PI controller, the number of nodes required in the input layer is two (i.e. *i* and *j*), where one is used to represent proportional gain, and the other is used to represent integral gain. The number of hidden layers and the number of neurons in the selected hidden layer were chosen with the goal of achieving the optimal balance between drive performance and computational complexity. The output of the ANN is command torque, requiring only one neuron in the output layer. Figure 6-7 shows the ANN scheme for speed drive using the indirect field oriented control of IM.

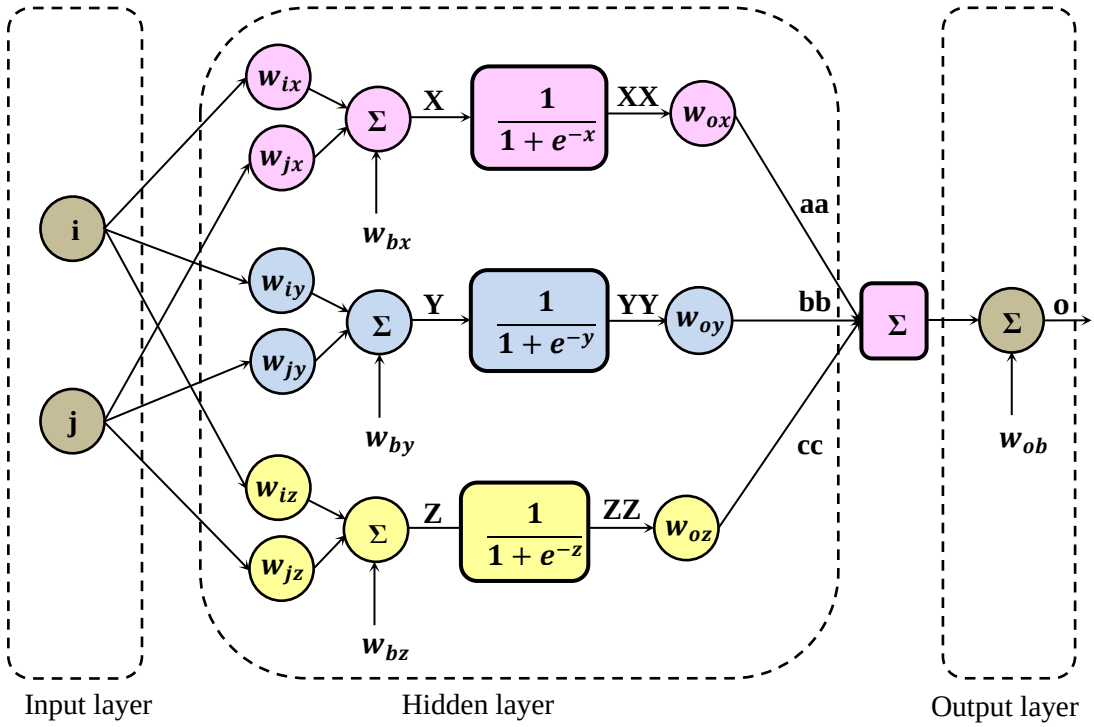


Figure 6-8: Artificial Neural network architectures

Figure 6-8 shows the ANN architectures employed in this study, where *i* and *j* represent error and integral of error, respectively, and O is the control signal. Alternately stated, *i* and *j* represent the inputs of the neuron and O is the output of the neuron. The summed inputs into hidden layer neurons are processed by a nonlinear sigmoid function as they pass through the neuron. The sigmoid function, or 'S' shape as illustrated in figure 6-9, is used as the activations function for all neurons and is defined as follows:

$$Y = f(i) = \frac{1}{1+e^{-\alpha i}}, \qquad 0 \le f(i) \le 1 \tag{6.9}$$

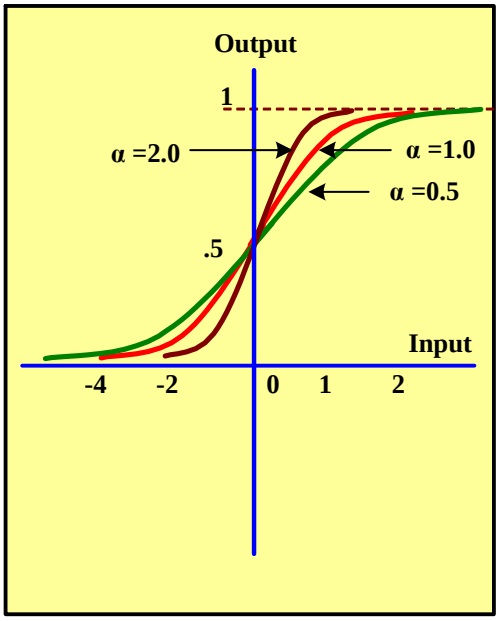


Figure 6-9: Sigmoidal functions

Note that as I tends to +∞, the value of $f(i)$ tends to +1, while as i tends to -∞, $f(i)$ tends to 0, with a smooth transition between the two. When i is equal to 0 the value of $f(i)$ is 0.5. By varying α, different shapes of the function can be obtained, which adjusts the abruptness of the function as it changes between the two asymptotic values.

The sigmoid function $f(i)$ and its derivative with respect to α, $f'(i)$, are shown in figure 6-10 and expressed as follows:

$$f'(i) = \frac{e^{-\alpha i}}{(1+e^{-\alpha i})^2} \tag{6.10}$$

Note that as i tends to +∞ or -∞ the value of $\frac{df}{di}$ tends to 0, and when i is equal to 0 the value of $\frac{df}{di}$ is $\frac{1}{4}$.

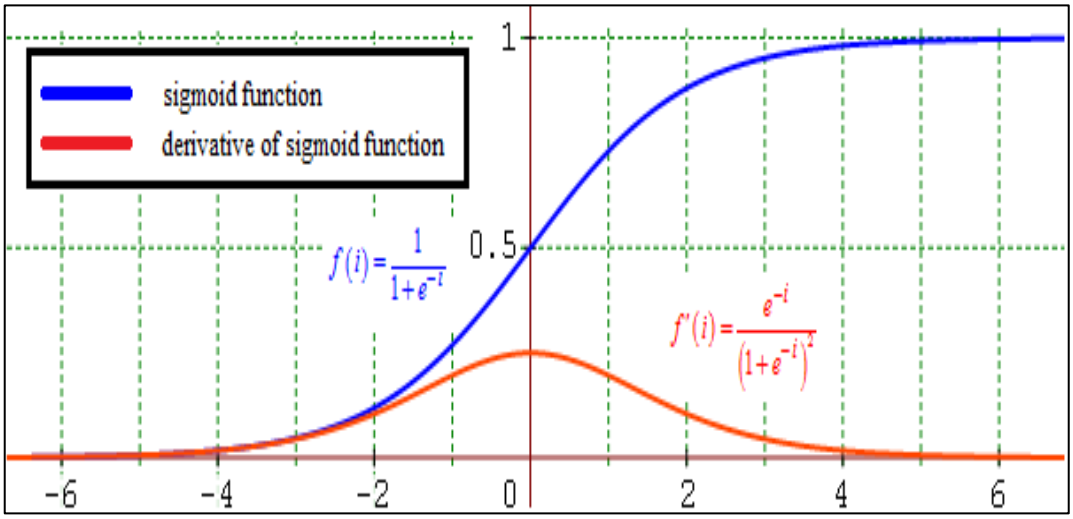


Figure 6-10: Sigmoid function and derivative of sigmoid function

The output of the proposed ANN is expressed as

$$o = W_{ob} + W_{ox} f\left(W_{ix} i + W_{jx} j + W_{xb}\right) + W_{oy} f\left(W_{iy} i + W_{jy} j + W_{yb}\right) \tag{6.11}$$

where i is the speed error and j is the integral of the speed error.

The output of a classical PI controller is

$$o = K_P i + K_I j \tag{6.12}$$

This controller gives zero output when i and j are both zero. To obtain a neural network controller which mimics the PI controller, the conditions in equations (6.13) and (6.14) will need to be met:

$$W_{jx} = W_{iy} = 0 \tag{6.13}$$

$$W_{bx} = W_{by} = 0 \tag{6.14}$$

The saturation limits of the controller are set by the values of W_{ox} by W_{oy} and would be determined by hardware, meaning that W_{bo} would generally be known. The output of the neural network controller reduces to

$$o = W_{ox} + f\,(\mathrm{W}_{ix}\, i) + W_{oy} f\,(\mathrm{W}_{jy}\, j) + \mathrm{W}_{ob} \tag{6.15}$$

Differentiation of the PI controller output with respect to i gives the proportional gain K_P, while differentiation of the output with respect to j gives the integral gain K_I. Differentiating the neural network controller output with respect to i when i is zero, and recalling that f is a sigmoid function, gives:

$$\frac{do}{di} = W_{ox} \frac{df\,(W_{ix} i)}{di} = W_{ox} \frac{1}{4} W_{ix} \tag{6.16}$$

Setting this equal to the proportional gain gives

$$\frac{do}{di} = K_P \tag{6.17}$$

$$K_P = W_{ox} \frac{1}{4} W_{ix} \tag{6.18}$$

Rearranging (6.18) gives

$$W_{ix} = 4K_P / W_{xo} \tag{6.19}$$

Differentiation of the neural network controller output with respect to j when j is zero gives

$$\frac{do}{dj} = W_{oy} \frac{df\left(W_{jy} j\right)}{dj} = W_{oy} \frac{1}{4} W_{jy} \tag{6.20}$$

Setting this equal to the integral gain gives

$$\frac{do}{dj} = K_I \tag{6.21}$$

$$K_I = W_{oy} \frac{1}{4} W_{jy} \tag{6.22}$$

Rearranging (6.22) gives the following:

$$W_{jy} = 4K_I / W_{yo} \tag{6.23}$$

The above tuning method produces a neural network controller with initial weights and biases that mimic a PI controller, but with the traditional constant gains K_P and K_I replaced with non-linear sigmoid functions. A set of weights has been used to create a suitable map which makes the neural network have more degrees of freedom and a more local character.

6.4.2 Simulation Results and Discussion

In order to verify the effectiveness of the ANN based speed controller for the IM drive, several tests are conducted. The aim of using ANN is to create a suitable map. This map has to have more degrees of freedom and a more local character than the conventional PI controller. To train a controller based on the integrated squared difference over a set period of time, it is preferable to develop a model of the system and train it before uploading the controller to the actual system. For online training it is best

to select the most sensitive parameters based on simulation and adjust only those parameters step by step in time. However, the ANN controller map must be able to mimic the conventional PI controller when there is no change in its parameters.

For comparative evaluation purposes, a PI controller-based IM drive system has been implemented and tested. Furthermore, extensive simulations have been carried out in order to evaluate the performances of both the ANN and PI techniques at different operating conditions. The performances of the IM drive for both the conventional and proposed speed controllers are investigated for step changes in command speeds and load torque.

The simulated performances of the IM drive system are shown in figures 6-11 to 6-16. Figures 6-11 and 6-14 show the simulated speed responses for ANN and PI based speed controller performances respectively, when the rated load is applied to the IM. It should be noted that the two responses follow the tracking of the command speed. It is evident from figure 6-11 that the ANN-based speed drives have a capability to reject the load disturbance applied to the IM, similar to the conventional PI speed controller. Figures 6-12 and 6-15 show the torque responses of both controllers with load torque (2 Nm) applied at 1 sec. The corresponding stator currents of the ANN and PI controller-based IM drives are shown in figures 6-13 and 6-16, respectively.

From these results, it is clear that the ANN controller has been successfully tuned, via the off-line PI-referenced method presented in this section, to exhibit performances equal to those obtained with the conventional PI controller upon which it is based. The results show that under the conditions presented here, the ANN controller is, in fact, producing control signals in agreement with the reference PI controller.

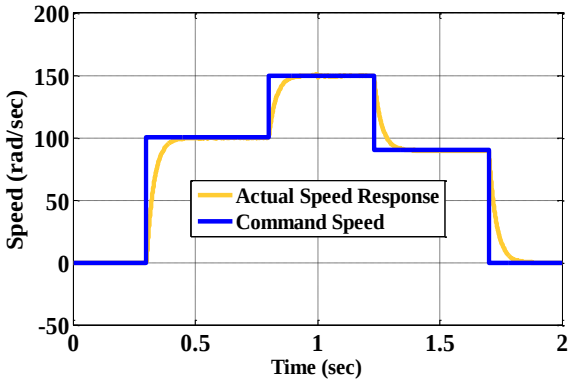


Figure 6-11: Simulated response of the PI controller-based IM to step changes of command speed with applying load torque (2 Nm).

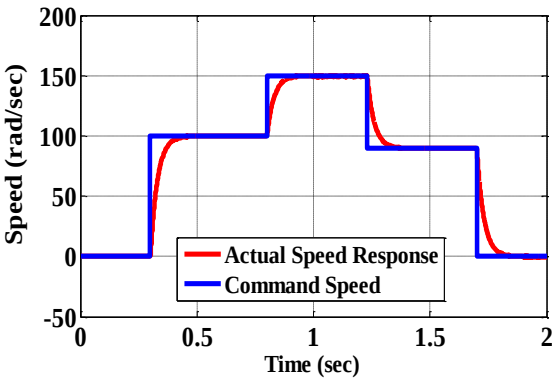


Figure 6-14: Simulated response of the ANN controller-based IM to step changes of command speed with applying load torque (2 Nm).

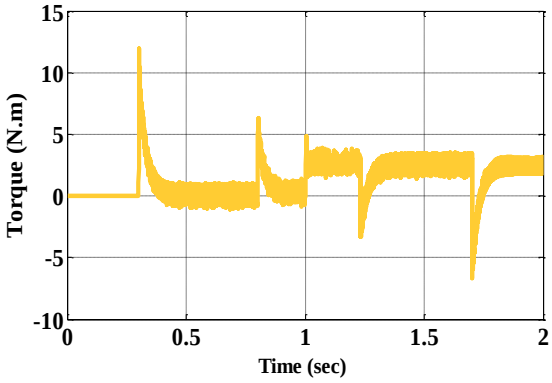


Figure 6-12: Simulated torque of the PI controller-based IM drive under the application of load torque (2 Nm).

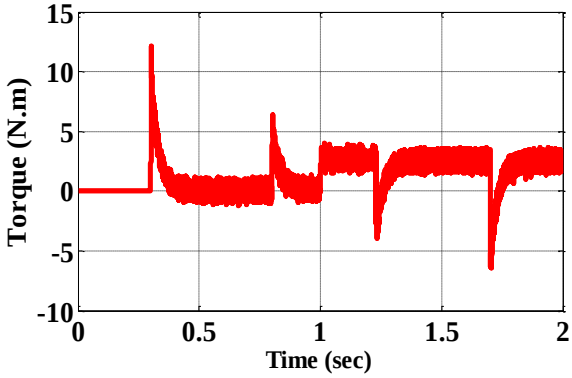


Figure 6-15: Simulated torque of the ANN controller-based IM drive under the application of load torque (2 Nm).

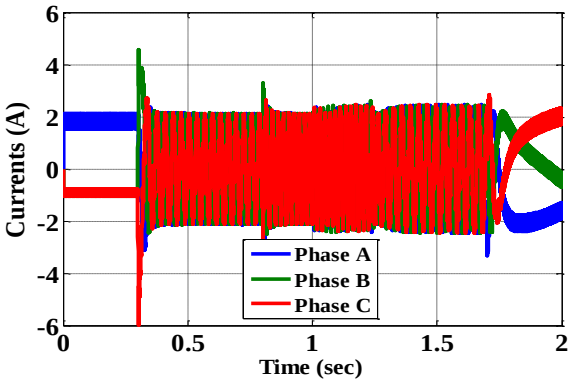


Figure 6-13: Simulated currents of the PI controller-based IM drive under the application of load torque (2 Nm).

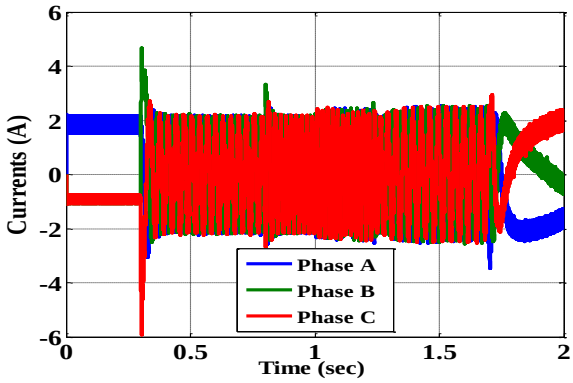


Figure 6-16: Simulated currents of the ANN controller-based IM drive under the application of load torque (2 N m).

6.5 Experimental Results and Discussion

The control algorithm for the ANN-based IM drive was implemented through software by developing a program in ANSI C language. After initializing the required variables, the initial set of weights and biases attained from the offline training of the ANN structure was downloaded.

This work evaluates the performances of the proposed ANN based drive system. The result obtained for the PI controller is shown in figure 6-17 and the result for the ANN controller is shown in figure 6-18.

The results show that the proposed drive system has the ability to follow the reference speed under no load conditions. It should be mentioned that the effect of sudden load impact on dynamic speed response and the evaluation of speed with parameter variation of the ANN based IM drive were performed solely by computer simulation, as depicted in figures 6-11 and 6-14. It can be noted for these experimental results that the performance of the ANN controller is as accurate in following the command speed as that of the PI controller. The results show that under the conditions presented here, the ANN controller again, has the ability to perform similarly to the reference PI controller.

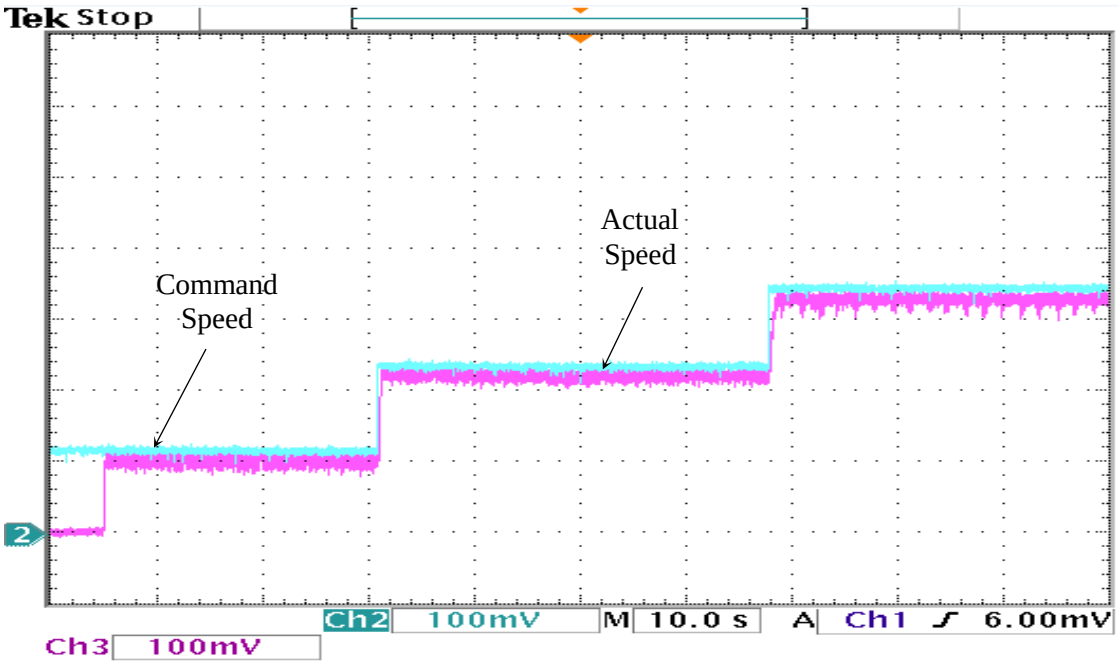


Figure 6-17: Experimental speed response of the drive for the step changes of the command speed based on the PI controller, (div=10 rad/sec)

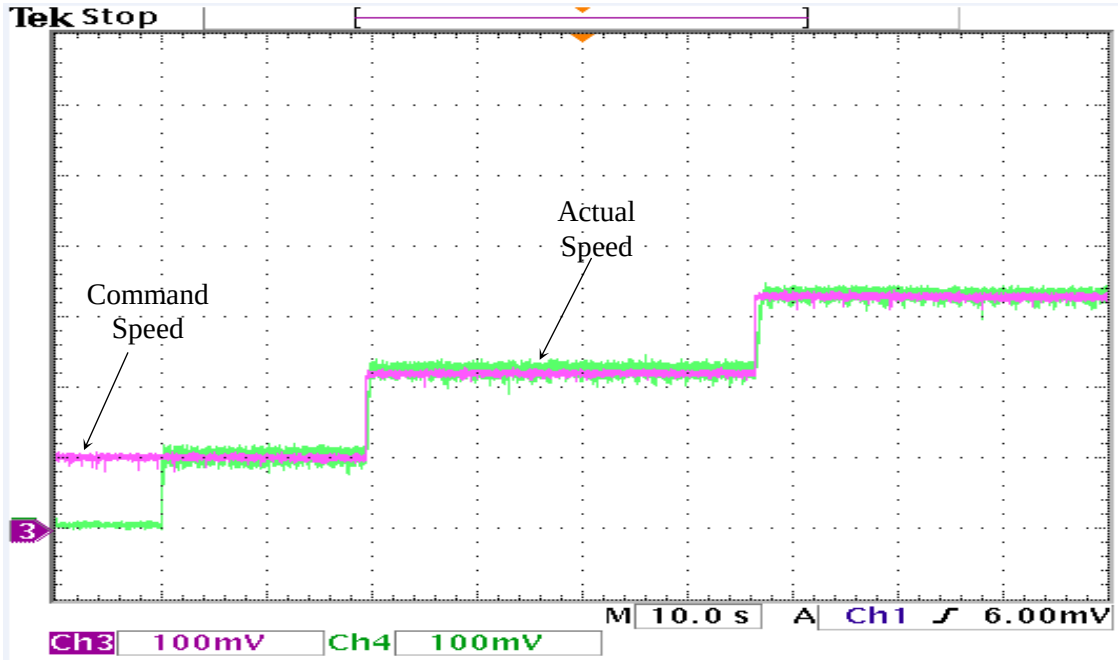


Figure 6-18: Experimental speed response of the drive for the step changes of the command speed based on the ANN controller, (div=10 rad/sec)

6.6 Conclusion

It has been shown that a set of weights was derived using mathematical reference to a conventional PI controller and used to create a suitable mapping, which characterizes the artificial neural network possessing two neurons with more degrees of freedom. ANN structures for this application are found to produce acceptably fast and accurate command speed tracking. This is due to their precision, ease of implementation, capacity to produce both positive and negative outputs, and quick convergence rates when used in the ANN for IM drives. Simulation results reveal some of these interesting features and show that the ANN network is appropriate for use as an alternative to the conventional PI control of induction motors. Both simulation and laboratory test results are given to verify the validity of this method.

6.7 Training Processes and Properties of Learning

A major advantage of artificial neural networks is their ability to learn from the presentation of samples that demonstrate system performance. Once the network has learned the relationship between inputs and outputs, it can generalize solutions, which means that the network can produce an output close to the desired output of any particular input value. Training a neural network means imposing a set of steps either to increase or decrease the weights and thresholds of its neurons. Such an iterative adjustment process, also known as a learning algorithm, aims to tune the network so that its outputs are close to the desired values.

6.8 Training Algorithms for Adaptive Artificial Neural Network

Artificial neural networks depend upon three things: input and activation functions of the neuron, neural architecture, and the weight of each input connection. Two of these things, input and activation function of the neuron and architecture of the network, are fixed, while the third, the weights of each neuron, vary. Therefore, the ANN performance is determined by the amount of these weights. These weights are initially set to arbitrary values. Once the input neurons are fed in, the desired output will be compared to each input of the neuron. Next, the entire neural network weights are slightly adjusted to bring the network's output more closely in correspondence with the desired output.

Neural networks learn by particular algorithms called learning algorithms. Their function adjusts network weights to improve their performance and minimize error, so as to reach the optimum result of these weights, which enables the network to achieve the best result. Figure (6-19) shows an adaptive general diagram of the way weights are adjusted.

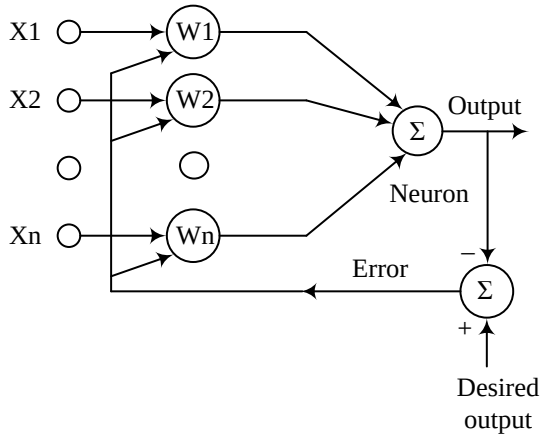


Figure 6-19: Adaptive diagram of adjusted weights

By contrast, in this study, the Grid Search Method (GSM) is used to search for the optimal weights required to make the neural network output converge to the desired output, even in instances where the system encounters unpredicted changes in parameters. The application of traditional artificial neural networks for motor drives encounters several shortcomings. One of them is that a greater amount of off-line training is necessary. This training takes a lot of time and depends upon detailed understanding of motor performance for the particular drive system. A further shortcoming is that when parameters outside the training set are found, poor drive performance may occur. Therefore, to address these shortcomings, the previous section is focused on the estimation of optimal initial weights with the aim of increasing speed of training. Limited off-line training will be involved in order to determine a new training set for optimal weights corresponding with desired output. The suggested method successfully produces efficient performance regardless of the operating parameter conditions. Furthermore, this method can be adapted easily to be used by different drive systems. For a more detailed explanation of the Grid Search Method (GSM) discussed in this section, refer to sections 4.5 and 4.74.

What has to be highlighted here is to illustrate how optimal weights are selected. After choosing the suitable size of the grid for every step, the desired output is compared to the actual output, and the weights of each neuron are altered based on the amount of error contributed. This error is squared, integrated, and stored. After many of these iterations, the weights reach values that provide the least error. All of these errors are stored as data; the size of data obtained depends on the grid size. Figure 6-20 demonstrates the

flowchart of self-tuning weights using a grid search optimization method. The process of selecting optimum weights is illustrated in figure 6-21.

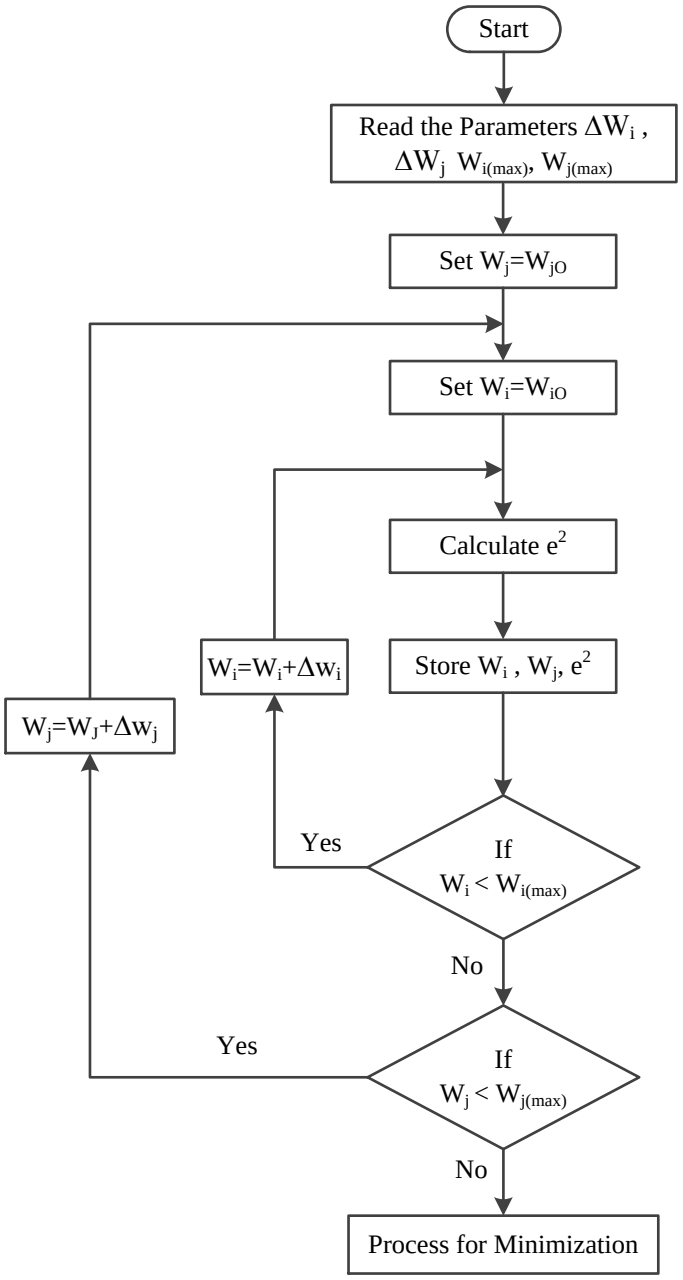


Figure 6-20: Flowchart of self-tuning weights using grid search optimization method

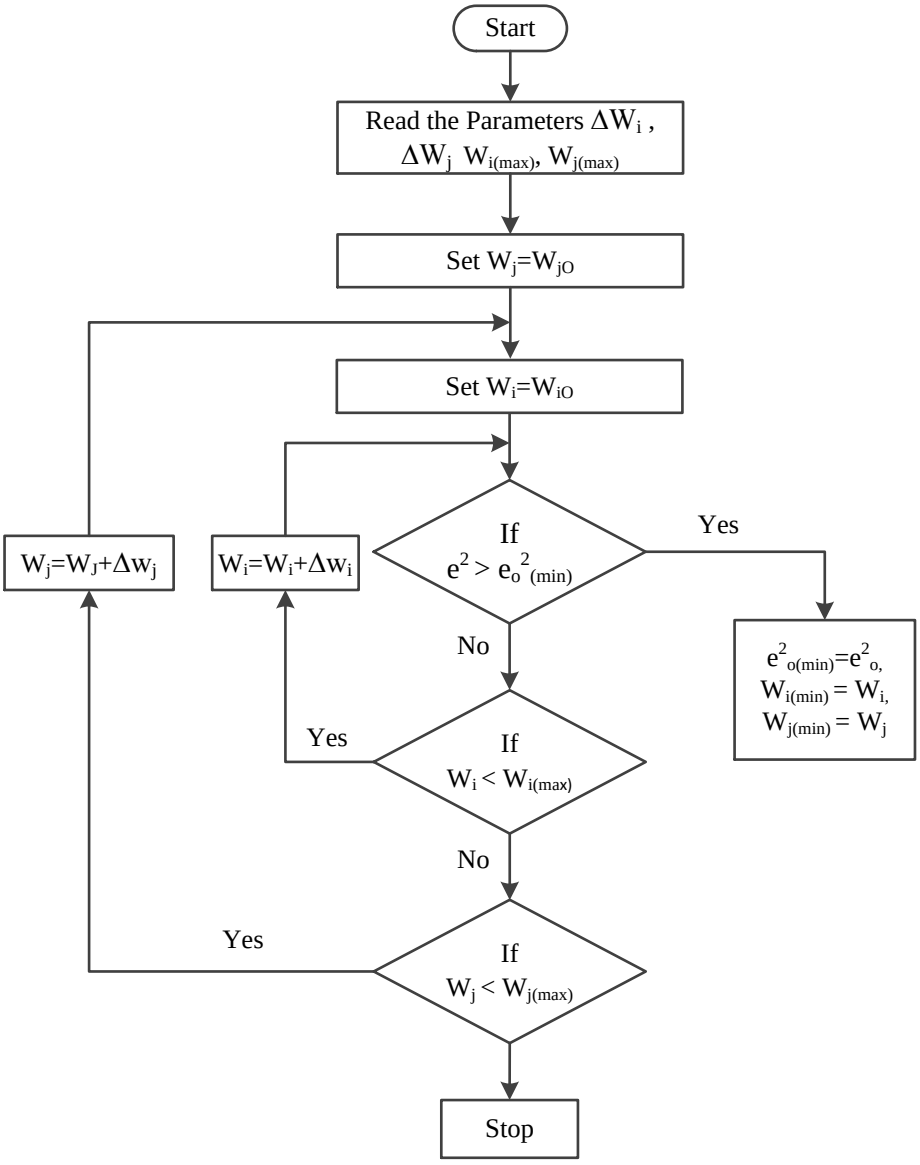


Figure 6-21: Flowchart illustrating the process of selecting optimum weights using GSM

The Grid Search Method (GSM) is in two stages. In the first stage, a coarse grid is created based on the knowledge of the system. The optimum weights, W_i , W_j , are those that were calculated from optimum K_p , and K_i . Then the coarse grid is set up between

weights, W_i versus W_j and the squared error (e^2) is evaluated at each point in the grid. Next, simple logic is used to determine the W_i, W_j point that has smallest squared error (e^2). In the second stage, an error squared (e^2) versus W_i quadratic equation is fitted through the smallest squared error (e^2) point, and neighbouring points east (W_{ie}) and west W_{iw}. Following this, setting the derivative of the squared error (e^2) with respect to W_i to zero $(\partial e^2 / \partial W_i) = 0$ and solving for the W_i gives the optimum weight (W_i). A similar approach using neighbouring points north (W_{in}) and south (W_{is}) gives the optimum weight (W_j). To express this in mathematical terms a quadratic formula is used as indicated in the equations (6.24) and (6.25).

$$\begin{cases} q_e = a + bW_{ie} + cW_{ie}^2 \\ q_p = a + bW_{ip} + cW_{ip}^2 \\ q_w = a + bW_{iw} + cW_{iw}^2 \end{cases} \tag{6.24}$$

$$\begin{cases} q_n = a + bW_{jn} + cW_{jn}^2 \\ q_p = a + bW_{jp} + cW_{jp}^2 \\ q_s = a + bW_{js} + cW_{js}^2 \end{cases} \tag{6.25}$$

These optimum values should be the same as the optimum values based on K_p and K_i.

When these three points are placed in (6.24) and (6.25), both sets of equations should produce zero values. In figure (6-22), the minimum point on the surface is going to occur when the slope with respect to W_i and the slope with respect to W_j are both zero.

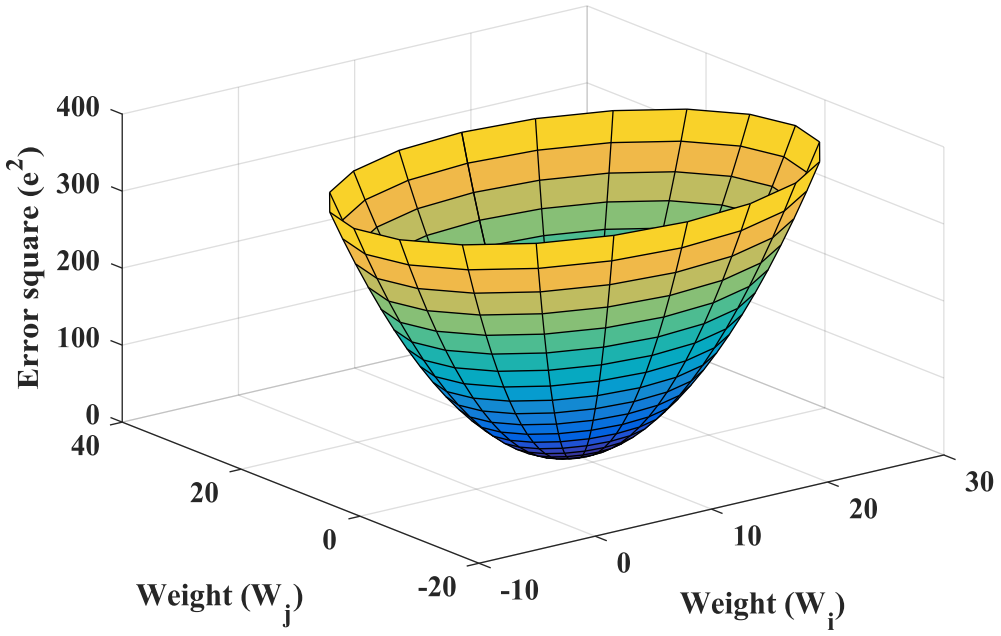


Figure 6-22: Error Square (e^2) versus weights (W_i, W_j) bowl

Using the above quadratic equations (6.24) and (6.25), coefficients (a, b and c) are calculated, followed by the derivatives of q with respect to W_i, W_j. These derivatives are set to zero in order to find the weight at the minimum.

It should be noted that the plot in a two dimensional plane as demonstrated in figure (6-23) is for squared error function versus one of directional weight (W_i, W_j).

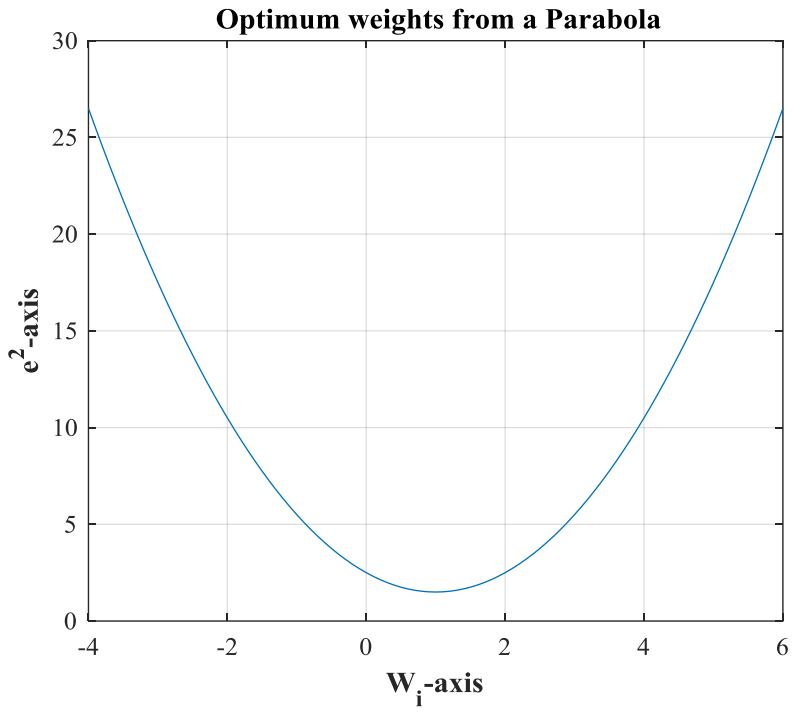


Figure 6-23: Error Square (e^2) versus weights (W_i)

6.8.1 Simulation Results Using Grid Search Method

In order to evaluate the effectiveness of the proposed algorithm, a series of simulation tests are conducted and observed under different operating conditions including step change in speed command and change in load command using MATLAB/SIMULINK. As illustrated in figures 6-24 to 6-37, the actual performances of speed response when the motor is run and the training algorithm is initialized by weights that are initially set to arbitrary values. As visible from figures 6-24, 6-25, 6-26, 6-30 and 6-31 the performance fluctuates in the beginning, which is expected because optimum weights are not used. The performance gradually improves, however, as a result of the minimized resulting difference error. The normalized integrated square error (ISE)

training law determines the value of the weights. After a certain number of iterations, the weights are slightly adjusted to bring the networks' output more closely in correspondence with the desired output. When the weights are properly adjusted and reach their optimum values, the speed performance accurately tracks speed command, as demonstrated in figure 6-27. The same principle is repeated, but only the rated load (2 Nm) is applied to the motor, with other different step changes in speed command given, as illustrated in the figures 6-25 to 6-34. It is worth mentioning that the slight dip caused by the load, quickly recovers from the load disturbance.

Speed difference affects the torque and current performance, a relationship which is demonstrated explicitly in figures (6-32), (6-35), (6-36), and (6-37), which correspond to the speed command illustrated in figure (6-24) and (6-27). Small disturbances of currents occur at the initial startup of the motor from standstill. At step change, the current response also changes significantly, but within a short time it recovers and at the sudden load disturbance there is a slight change in currents. This work has located the optimum proportional and integral gains corresponding to the minimum objective function over the grid for the self-tuning ANN controller. It is obvious from simulation results that the proposed approach is proven to be robust for IM drive applications. Additionally the developed algorithm approach is capable of high precision rotor speed tracking. Further, according to the simulation results, the proposed algorithm performs efficiently for speed control of the FOC IM model.

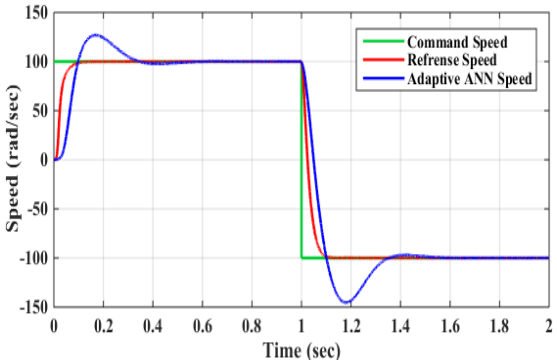


Figure 6-24: Initial speed response at 100 rad/sec without load

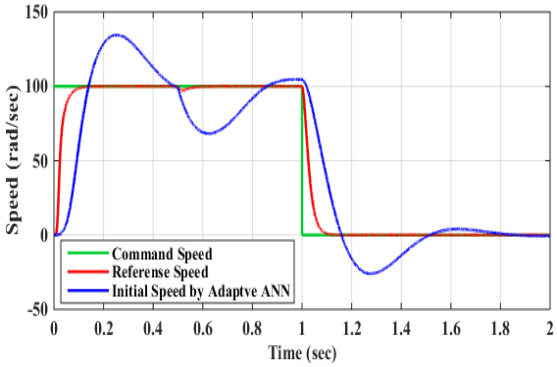


Figure 6-25: Initial speed response at 100 rad/sec for load (2 Nm) applied at 0.5 sec

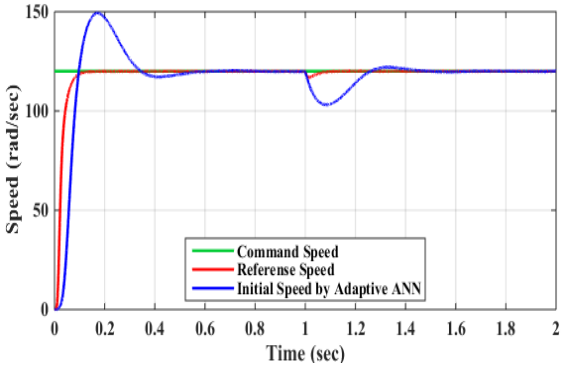


Figure 6-26: Initial speed response at 100 rad/sec for load (2 Nm) applied at 1 sec.

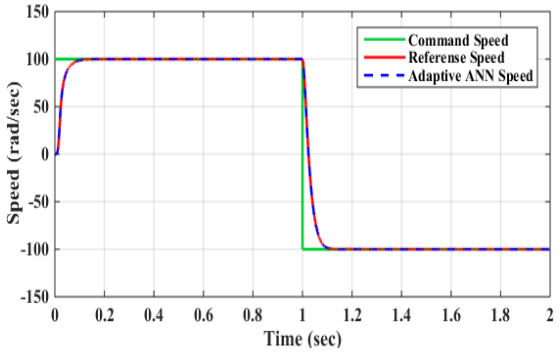


Figure 6-27: Final speed response at 100 rad/sec without load

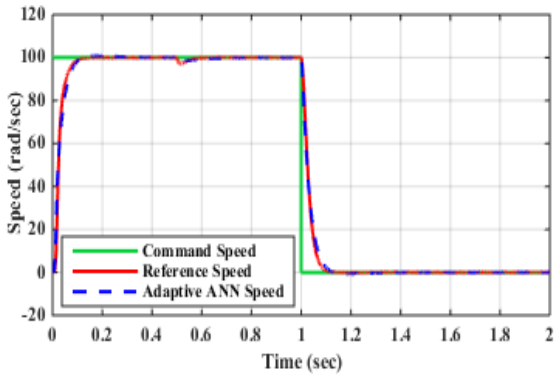


Figure 6-28: Final speed response at 100 rad/sec for load (2 Nm) applied at 0.5 sec

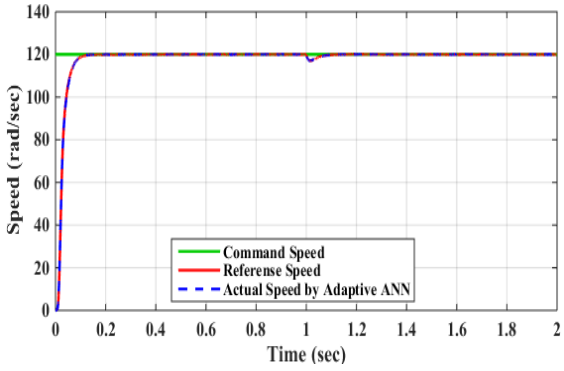


Figure 6-29: Final speed response at 100 rad/sec for load (2 Nm) applied at 1 sec.

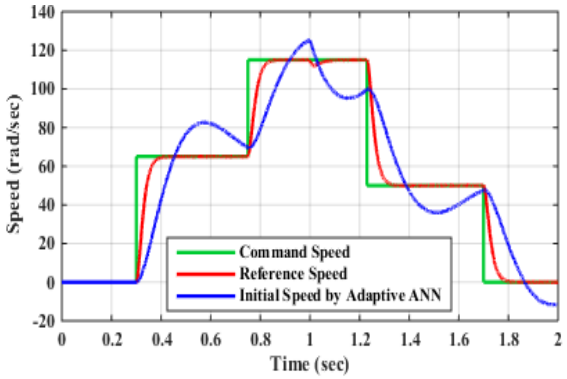


Figure 6-30: Initial speed response at 115 rad/sec for load (2 Nm) applied at 1 sec

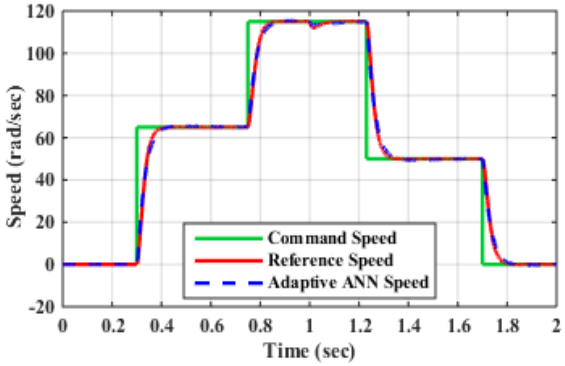


Figure 6-33: Final speed response at 115 rad/sec for load (2 Nm) applied at 1 sec

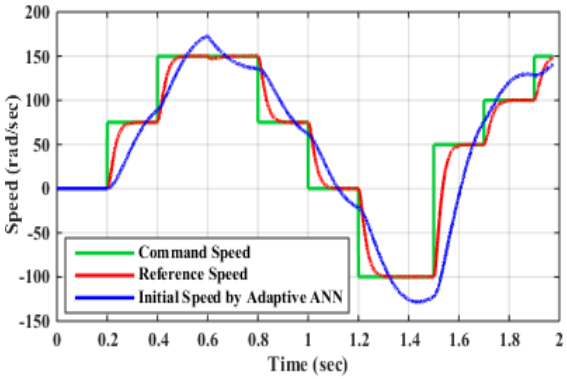


Figure 6-31: Initial speed response at 150 rad/sec for load (2 Nm) applied at 0.6 sec

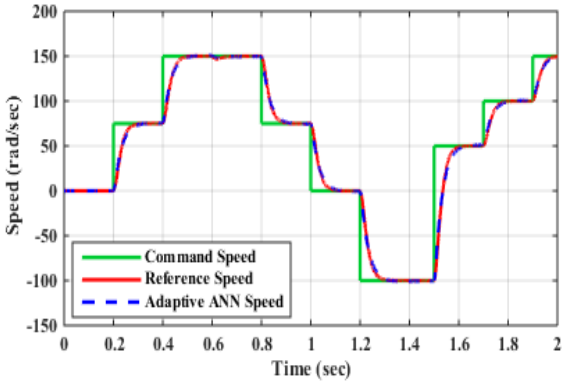


Figure 6-34: Final speed response at 150 rad/sec for load (2 Nm) applied at 0.6 sec

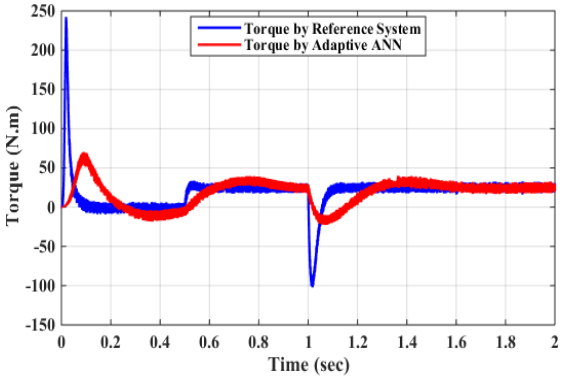


Figure 6-32: Initial developed torque corresponding to the speed illustrated in figure (6.24)

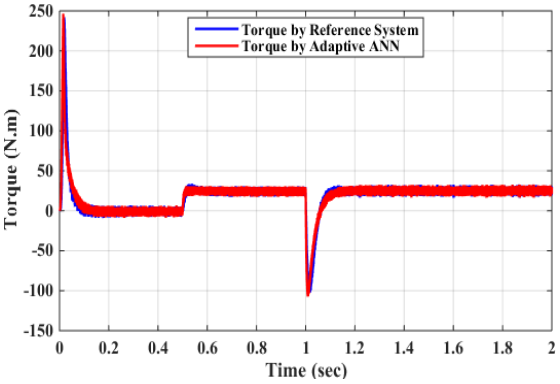


Figure 6-35: Final developed torque response corresponding to the speed illustrated in figure (6.27)

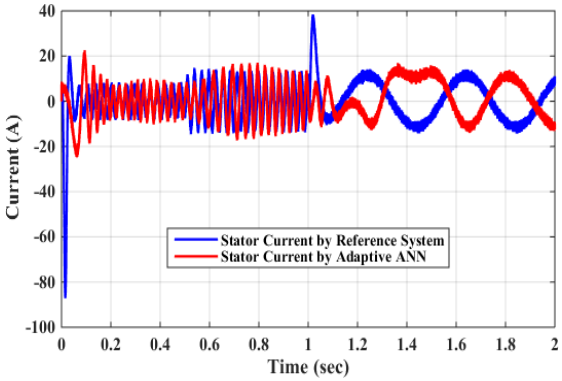


Figure 6-36: Initial current response corresponding to the speed illustrated in figure (6.24).

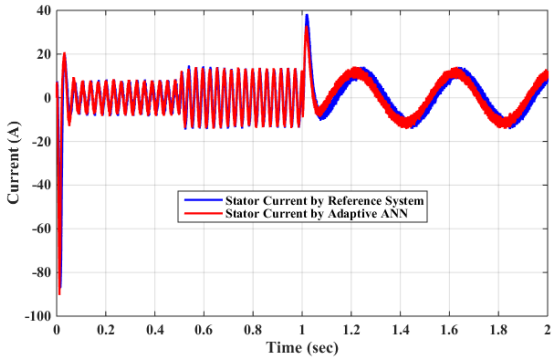


Figure 6-37: Final current response corresponding to the speed illustrated in figure (6.27).

6.9 Chapter Summary

In this chapter, both fixed and self-tuning neural networks have been introduced. In the first case, the weights in the neural networks are fixed and cannot be changed. Therefore, the weights are predetermined according to the problem to be solved. For instance, in motor applications where the PI is used to drive the motor, this chapter demonstrates a method for estimating synaptic weights derived and used to represent a fixed neural network. Hence, this method will speed up the convergence rate and reduce the burden and expense on the processor required for carrying out all the computations. The weights in self-tuning neural networks are changed by the grid search method incorporated with the integrated squared error as a performance index, and have been used to attain the optimum weights that are required by the neural network to make the output network follow the desired output. Both schemes have been applied to the induction motor drive. A series of tests has been undertaken in order to validate the efficiency of these approaches. The simulation results obtained demonstrate their success. In addition, a comparison of the results obtained with the traditional PI controller is provided. Furthermore, implementation for the fixed neural network has been successfully performed in the laboratory. All of the controllers except fuzzy logic were made self-tuning. By adjusting breakpoints, the fuzzy logic could have been made self-tuning. Since each of the intelligent controllers was designed to mimic the conventional PI controller, each of them could be made self-tuning.

Chapter 7.

Finite Element Controller Map Method (FECMM)

7.1 Introduction

It is known that induction motors (IMs) are the industry standard. Three phase induction motors have been widely used in ac adjustable speed drives. Design and control of IMs pose a challenge due to these motors' non-linear behaviours, such as saturation, temperature effects, parameter variations, etc. Extensive research exists in the literature for finding the most desirable techniques to meet these challenges.

Most of these models, though, are complex, and depend upon particular parameters. Furthermore, they incorporate assumptions that decrease the accuracy of the mathematical model, which is not desirable.

Thus, it clearly still remains desirable within the motor control industry to design control algorithms which achieve simplicity, better tracking performance, robustness against parameter variations, saturation, load disturbance rejection, fast-tracking convergence and adaptability in real time to variations in drive system behaviour. Therefore, advanced control algorithms are developed to address these challenges. This chapter thus introduces a novel control scheme for induction motor drives based on a Finite Element Controller Map Method (FECMM). This technique allows an approximate solution using a linear combination of simpler functions. These simpler functions are called shape functions. This approach of FECMM has the capability to provide high performances including high efficiency, robustness, stability and fast convergence rate and creates a

complex contour that provides a high potential for adaptation for industrial drives. Furthermore, this FECM technique can also be upgraded to develop the algorithm for offline tuning to adjust the values of element nodes for sudden changes in operating conditions of the induction motor drives.

Before explaining this FECMM approach, it is necessary to define some of the concepts associated with this technique.

7.2 Nodes

The boundaries of an element are identified by selected nodes, which are specific, selected points used to determine basic unknowns by using approximation, interpolation, or shape function. Any unknown point within the element can be identified using these interpolation functions. There are two kinds of nodes: external and internal.

External nodes include nodes that are located either on the corners or along the sides of an element. Internal nodes, rather, are located inside the element.

7.3 Coordinate Systems

The following terms are commonly referred to in FECMM:

(i) Global coordinates

(ii) Local coordinates

(iii) Natural coordinates

7.3.1 Global coordinates

The global coordinates system is the system used to determine an entire structure's points. The Cartesian global coordinates system is often used to define ordinary cases.

7.3.2 Local coordinates

A local coordinates system is a different coordinate system employed for each element to attain element properties in a convenient manner.

7.3.3 Natural coordinates

A natural coordinates system allows a point to be specified in an element using a set of numbers with no dimensions, the magnitude of which is never greater than 1. The coordinates of any point inside the element can be defined by assigning weight to those coordinates. Therefore, this system has a feature such that the value for any particular node has to be unity, while 0 is the value at all other nodes. When any element is mapped according to the natural coordinates system, finding the properties for individual nodes within the element is an easier process than with the Cartesian system, because the algebra involved is simpler.

7.4 Shape Functions

To obtain an approximate solution as close as possible to the exact solution, it is necessary to assume the approximate solution, using a linear combination of simpler functions. These simpler functions are called shape functions. The sum of all the shape functions at any node must be equal to unity at that node, and zero at other nodes. This condition has to be satisfied for all nodes on the element. The solution accuracy is only ensured by the number of nodes assigned to the discrete element. Introducing additional nodes results in increasing the order of the polynomial function, which consequently improves the accuracy of the results, but the price is the amount of computational burden (time) required.

The known values of the nodes in the field variable can be employed to determine the approximate value of the unknown nodes within the element, using the interpolation function of the nodal values. The finite element method formulation of the problem domain results in interpolation functions which are usually expressed as polynomial forms with independent variables. These functions are used to achieve necessary nodal requirements [136-141].

7.5 Isoperimetric Element

Assume the quadrilateral element that is illustrated in figure 7-1.

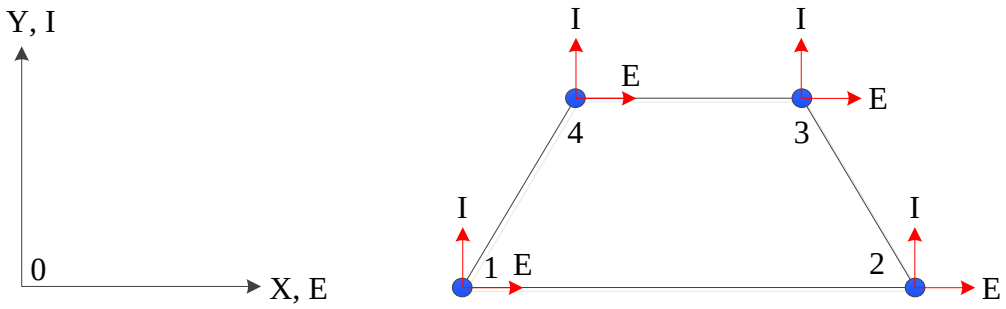


Figure 7-1: Construction of quadrilateral element

To construct the expression for the shape function which relates to the quadrilateral element, mapping this element directly in the Cartesian coordinates system is technically difficult because the shape functions obtained will be complicated. Therefore, mapping the element into natural coordinates (α, β) makes it easier to construct the shape functions for any nodes within the element.

7.6 Lagrange Interpolation Functions: Linear Rectangular Element

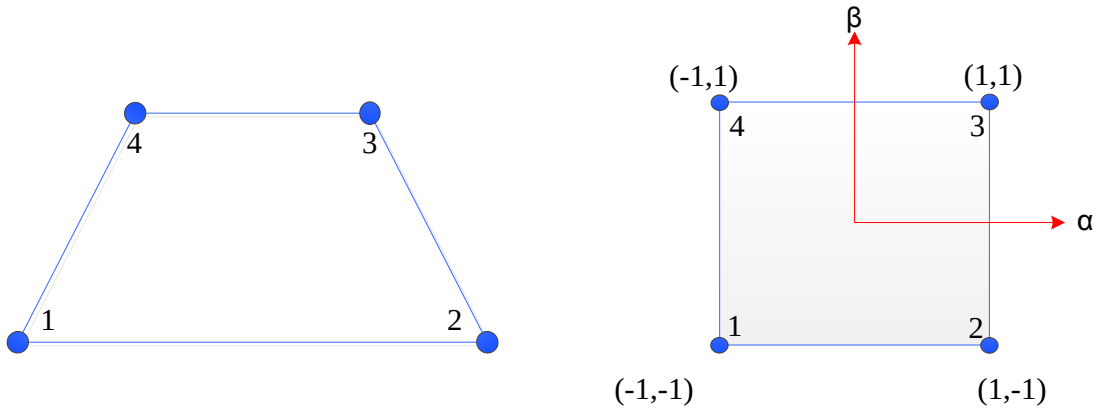


Figure 7-2Four-node quadrilateral element of Lagrange interpolation functions

Figure 7-3: Four-node quadrilateral element of Lagrange interpolation functions

The element shape and local coordinates for the four-node quadrilateral element are shown in Figure 7-2.

In order to construct the shape functions, consider node 1 as an example. Node 1 disappears over nodes 2, 3 and 4, and must be made equal to unity at node 1 by adjusting c in equation (7.1). Node 1 vanishes on the lines $1-\beta=0$, $1-\alpha=0$.

Therefore, N_1 takes the form

$$N_1(\alpha,\ \beta)=c\ L_{2-3}\ L_{3-4} \tag{7.1}$$

where the symbol L_{2-3} , L_{3-4} denotes the local coordinate line that passes through nodes 2 and 3, nodes 3 and 4 respectively

$$N_1(\alpha,\ \beta)=c\,(1-\alpha)(1-\beta) \tag{7.2}$$

To find c, the shape function at node 1 (N_1) must be equal to unity, and the natural coordinates (α, β) have to be made equal to their corresponding point, which in this case is (–1, –1). This leads to

$$N_1(-1, -1) = c*2*2 = 4c = 1 \quad \Rightarrow \quad c = 1/4 \tag{7.3}$$

$$N_1(\alpha, \beta) = 1/4(1-\alpha)(1-\beta) \tag{7.4}$$

Next, to construct the shape function for node 2, N_2 vanishes along the lines $1+\alpha = 0$, $1-\beta = 0$ Let

$$N_2(\alpha, \beta) = c(1+\alpha)(1-\beta) \tag{7.5}$$

$N_2 = 1$ at $(\alpha, \beta) = (1, -1)$, this leads to $c = 1/4$

$$N_2(\alpha, \beta) = 1/4(1+\alpha)(1-\beta) \tag{7.6}$$

Similarly, applying the same principle as above,

$$N_3(\alpha, \beta) = 1/4(1+\alpha)(1+\beta) \tag{7.7}$$

$$N_4(\alpha, \beta) = 1/4(1-\alpha)(1+\beta) \tag{7.8}$$

Once shape functions have been identified, formulation of higher order polynomial functions for quadrilateral elements is developed according to Pascal's triangle for the Lagrange quadrilateral elements, as shown in figure 7-3.

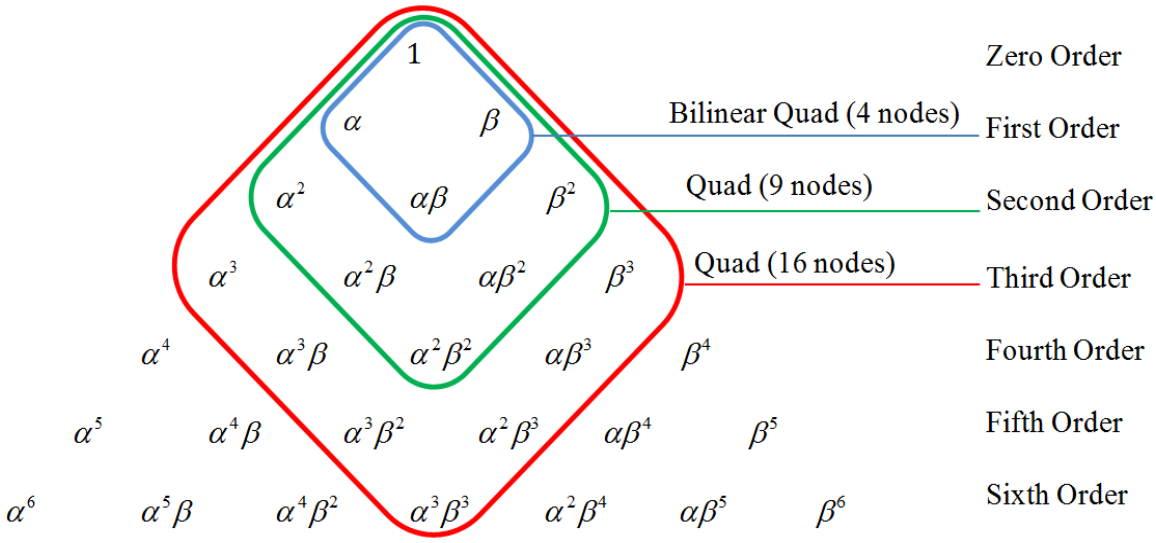


Figure 7-4: Triangle used to determine binomial coefficients

Pascal's triangle reveals the number of required nodes for modeling the solution region of the field problem for any order with great accuracy.

Figure 7-3 illustrates the triangle, which defined as the array of the missing binomial coefficients. For example, for a polynomial having four terms, the missing binomial coefficients can be obtained if the following expression is selected from the triangle.

$$E(\alpha,\ \beta) = a_1 + a_2\alpha + a_3\beta + a_4\alpha\beta \tag{7.9}$$

7.7 Finite Element Controller Map Method

Normally, a Finite Element Method (FEM) is used to find the solution of the boundary conditions problem governed by partial differential equations (PDE). Many PDEs cannot be solved analytically but are usually solved using numerical methods to obtain an approximate solution. The solution generated by this method is an approximation of the real solution to the PDEs. However, in this study, the FEM is used to construct a controller map. The finite element controller map method depends on the number of nodes assigned within the discrete finite elements. However, in this study a finite element controller differs from what has been published, and is used to create a map of the output response. This map has more degrees of freedom and a more local character.

At a point within a finite element, the control signal Q is given as the sum of scaled shape functions:

$$Q = \sum \mathrm{N.\ m} \tag{7.10}$$

where m is the control signal at a node and N is a shape function. The shape functions are generally given in terms of local coordinates. Each local coordinate has a range -1 to +1. For a PI controller, the global coordinates would be the error E and the integral of error I. At a point within an element, the global coordinates can be written as

$$\mathrm{E} = \sum \mathrm{N.a} \tag{7.11}$$

$$\mathrm{I} = \sum \mathrm{N.\ b} \tag{7.12}$$

where a and b are nodal values of E and I respectively. At any instant in time, the current values of E and I would be known. Simple geometry can be used to determine which

element contains the operating point. Then the E and I equations can be used to find the local coordinates of the operating point. Once the local coordinates are known, the control signal Q can be calculated.

For a three-dimensional (3-D) PID element, the shape functions are as follows:

$$\begin{aligned} N_1 &= 1/8(1-\varepsilon)(1-\alpha)(1-\beta) & N_2 &= 1/8(1-\varepsilon)(1+\alpha)(1-\beta) \\ N_3 &= 1/8(1-\varepsilon)(1-\alpha)(1+\beta) & N_4 &= 1/8(1-\varepsilon)(1+\alpha)(1+\beta) \\ N_5 &= 1/8(1+\varepsilon)(1-\alpha)(1-\beta) & N_6 &= 1/8(1+\varepsilon)(1+\alpha)(1-\beta) \\ N_7 &= 1/8(1+\varepsilon)(1-\alpha)(1+\beta) & N_8 &= 1/8(1+\varepsilon)(1+\alpha)(1+\beta) \end{aligned} \tag{7.13}$$

where α, β and ε are local coordinates. Each has a range -1 to +1.

The global coordinates are the error (E), the integral of error (I), and the derivative of error (D). The local versus global connection is shown as follows:

$$\begin{aligned} E &= \sum N.a \\ I &= \sum N.b \\ D &= \sum N.c \end{aligned} \tag{7.14}$$

where *a*, *b* and *c* are nodal values of E, I and D.

Higher order PID shape functions would produce a more complex PID map.

Linear PI shape functions for a 2-D finite element are expressed in the equations given below:

$$
\begin{aligned}
N_1 &= 1/4(1-\alpha)(1-\beta) \\
N_2 &= 1/4(1+\alpha)(1-\beta) \\
N_3 &= 1/4(1+\alpha)(1+\beta) \\
N_4 &= 1/4(1-\alpha)(1+\beta)
\end{aligned}
\tag{7.15}
$$

The local versus global connection is given as follows:

$$
\begin{aligned}
E &= \sum N.a \\
I &= \sum N.b
\end{aligned}
\tag{7.16}
$$

Linear P shape functions for a 1-D finite element are expressed as follows:

$$
\begin{aligned}
N_1 &= 1/2(1-\alpha) \\
N_2 &= 1/2(1+\alpha)
\end{aligned}
\tag{7.17}
$$

The global coordinate is the error (E). The local versus global connection is given as

$$
E = \sum N.a \tag{7.18}
$$

7.8 Finite Element Maps

A classic controller is commonly used in order to regulate the motor speed in a closed loop and achieve decoupled rotor flux and torque control. The PI speed controller generates the reference torque command. These torque and rotor flux reference commands are utilized to generate the motor d-q reference currents as indicated in figure 7-4. Consequently, the rotor flux angle θ_e for field orientation is obtained from adding the measured rotor speed to the calculated slip angle based on motor parameters. The classical PI controller was used to generate data to the finite element controller. Two typical types of FECMM are used in this study. One of them has nine elements, while the other has only five elements to minimize the computational burden (time) required by the scheme.

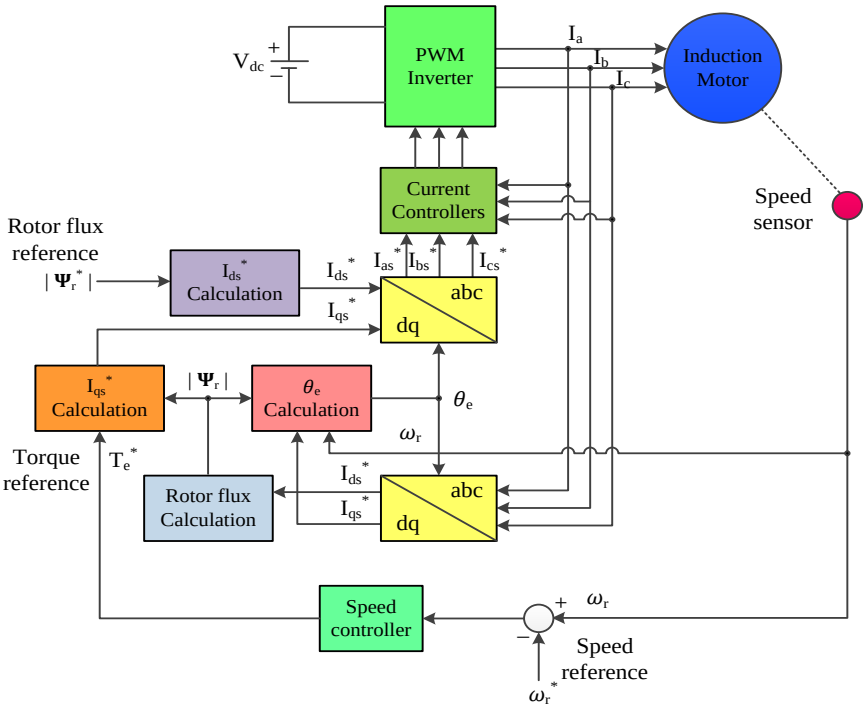


Figure 7-5: Schematic diagram of vector control of induction motor

7.8.1 Nine Element Controller Map

A finite element system map links inputs to outputs using scaled shape functions. For example, typical mapping of a two-input and one-output finite element controller can be depicted in a 3-D plot. The plot is often referred to as the control surface plot, such as the one shown in figure 7-5. The typical finite element inputs are the signals of error (E) and integral speed error (I). The finite element output (Q) is the controller action obtained from a sum of scaled finite element shape functions. Figure 7-5 shows the entire output controller surface of the system based on the 2-D PI shape function. The map has 9 rectangular elements with 16 nodes. Sets of elements are interconnected with a set of specified nodes as shown in figure 7-6.

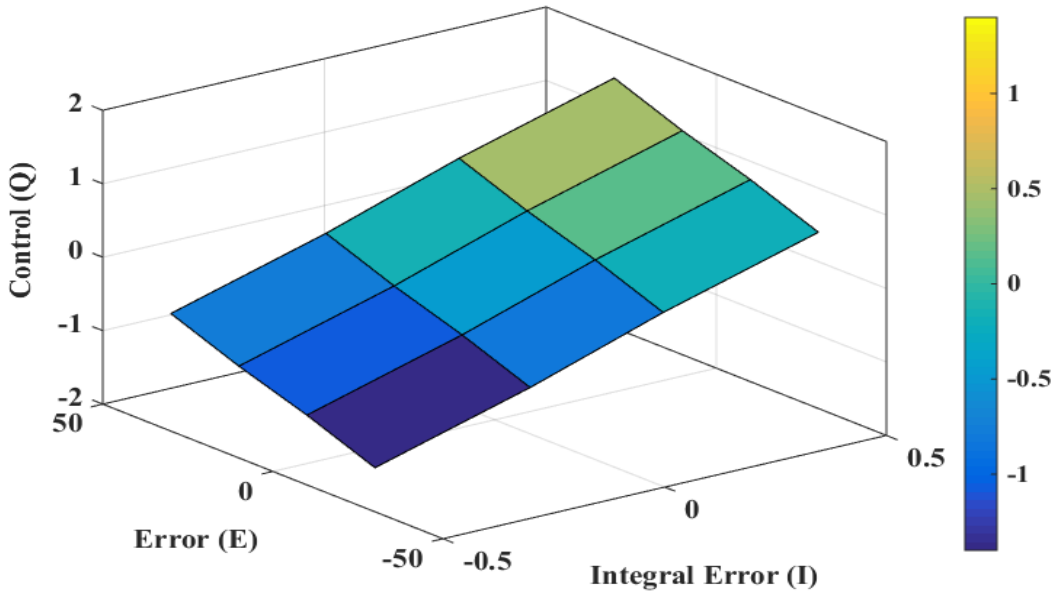


Figure 7-6: Output surface of proposed finite element controller based on 2-D PI shape function (speed controller)

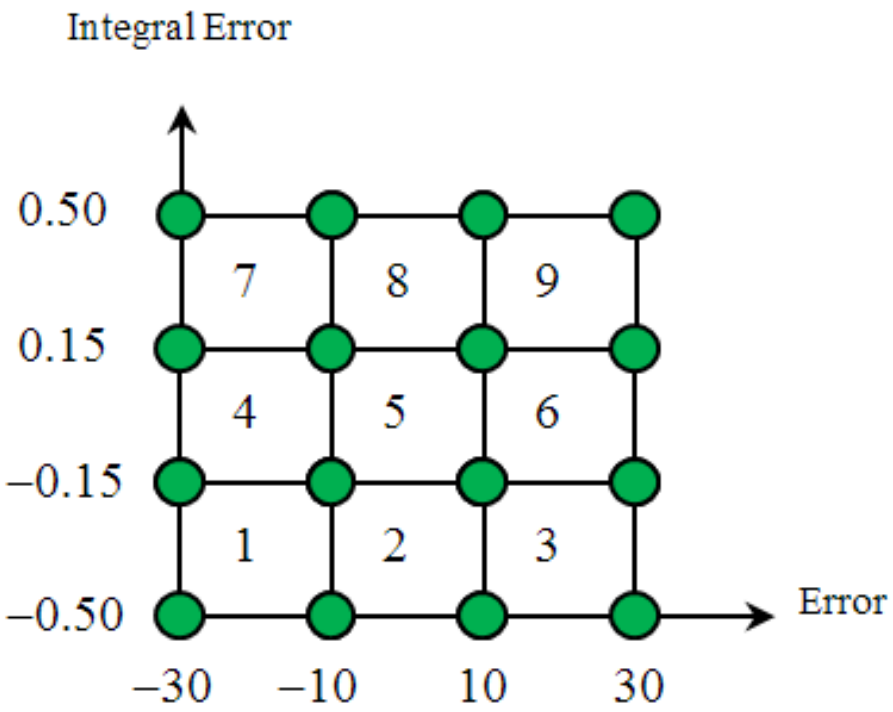


Figure 7-7: Sets of elements interconnected with a set of specified nodes

Table 7-1: Control Signal at Nodes

I \ E	–30	–10	10	30
–0.50	dd11 –97.8	dd12 –33.8	dd13 30.2	dd14 94.2
–0.15	dd21 –96.54	dd22 –32.54	dd23 31.46	dd24 95.46
+0.15	dd31 –95.46	dd32 –31.46	dd33 32.54	dd34 96.54
+0.50	dd41 –94.2	dd42 -30.2	dd43 33.8	dd44 97.8

7.9 Simulation Results and Discussion

7.9.1 Nine Element Results

In order to demonstrate the effectiveness of the proposed intelligent finite element based on vector control of the IM drive, simulations are carried out under a variety of operating conditions. The ratings and parameters for the tested induction motor are listed in Appendix A. Several tests were performed to evaluate the performance of the proposed method to ensure that it can operate in a wide range of conditions, in MATLAB/SIMULINK. The speed responses were observed under different operating conditions such as a sudden change in command speed, step change in load, etc. The results are presented in this section.

Several tests were performed in this research to evaluate the performances of both the PI-controller and the Finite Element Controller Map technique for IM motor drives. Speed and current responses under various operating conditions, such as change in reference speed, step change in load, parameter variations, etc., were observed. Tests were performed to obtain the speed response for various reference speeds with full load. Figures 7-7 and 7-10 demonstrate the responses of the FECM-based drive under the same operating conditions. It is observed from Figure 7-10 that the performances of the FECM-based IM motor drive system are comparable to those of the PI-controller-based system. However, the PI controller-based drive system can experience overshooting when a change in reference speed occurs. Although the PI constants are chosen to

provide the critically damped speed response based on the available motor design parameters, the motor parameters undergo change in actual operation. The designed PI controller provides the speed responses with slight overshooting. The design values of the PI controller were not modified because they provide reasonable rise times, which are comparable with those of the FECM-based system. The PI controller may be redesigned under damped conditions to reduce overshooting. In that case, the rise time becomes sluggish. The speed response of the FECM-based system can be robust against the change in operating conditions of step change in reference speed, or parameter variations, if the nodes which created the shape function map are updated automatically.

As visible from Figure 7-7, the motor was running at no-load until the moment 1.04 seconds when the constant load torque (2N.m) was applied suddenly. A slight dip was caused by the load, but it quickly and successfully recovered from load disturbance. The FECM adjusts its output to this changing condition of sudden load impact and provides appropriate control voltage so that the drive system responds according to the reference speed. Figure 7-10 shows comparable responses between traditional PI and proposed FECM-based systems. It is quite evident from figure 7-10 that the performances of the FECM-based system are equivalent to those of the PI-controller-based system. Since FECM nodes and output controller signal is calculated to mimic the conventional PI, the good agreement is not surprising. While speed difference affects the torque and current performance, this relationship is demonstrated explicitly in figures 7-8, 7-9, 7-11 and 7-12, which correspond to the speed command illustrated in figures 7-7 and 7-10. Small disturbances of currents occurred at initial start-up of the motor from standstill. At

step change the current response also changed significantly, but within a short time it recovers and at the sudden load disturbance there was a slight change in currents. In figure 7-12 phase currents blue and red lines converge eventually, since the speed command is going to zero after 1 sec.

The FECM again is equivalent to the PI controller, as is evident from figure 7-10. This is expected, as the FECM is designed to mimic the existing PI controller. It tracks the desired speed quite accurately.

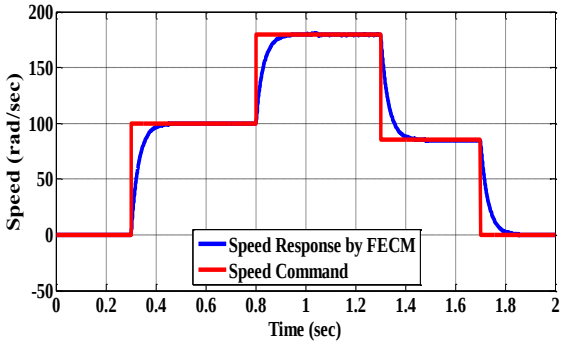


Figure 7-8: Speed tracking response for FECM technique with load (2 Nm) applied at 1.04 sec.

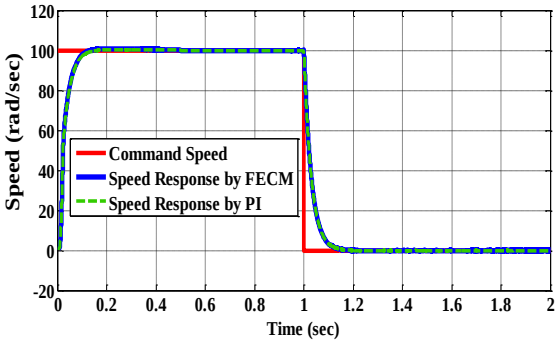


Figure 7-11 : Speed tracking response for FECM and PI controller techniques with load (2 Nm) applied at 0.5 sec.

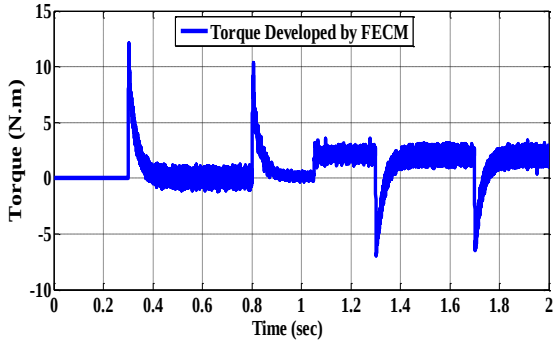


Figure 7-9: Developed torques of the FECM technique corresponding to the speed command given in Fig. 7-7 under the application of load (2 Nm) at 1.04 sec.

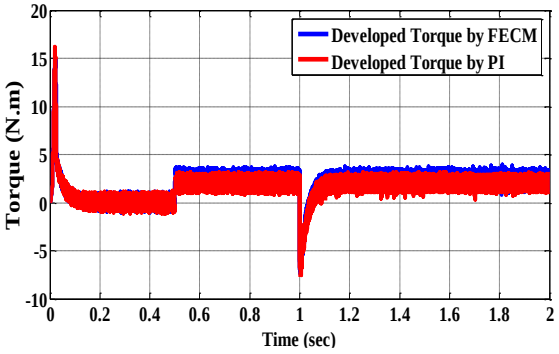


Figure 7-12: Developed torques by FECM and PI controllers corresponding to the speed command given in figure7-10 under the application of load (2 Nm) at 0.5 sec.

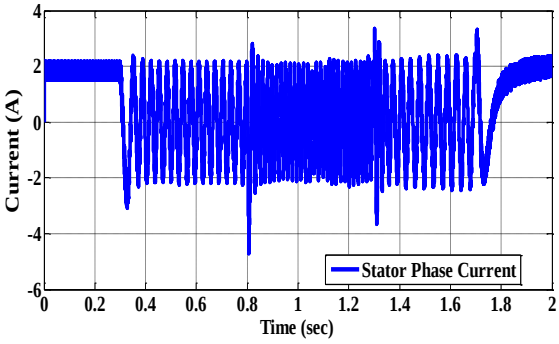


Figure 7-10: Stator Phase Current of the FECM technique corresponding to the speed command given in Fig. 7-7 under the application of load (2.0 Nm) at 1.04s sec.

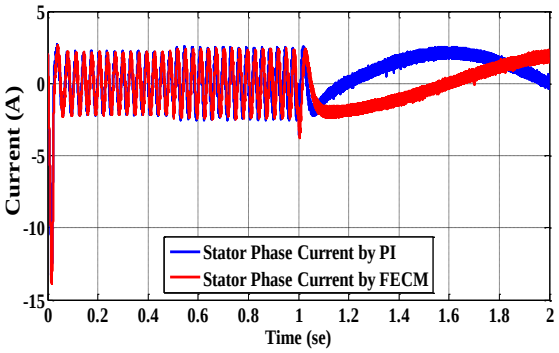


Figure 7-13: Stator Phase Currents of the FECM and PI controllers corresponding to the speed command given in figure 7-10 under the application of load at 0.5 sec.

7.10 Five Element Controller Map Method

7.10.1 Five Element with Four Nodes for each Element

The scheme strategy needs to determine where the global coordinates error (E) and integral of error (I) are located. Then a logical statement can be used to determine the suitable element corresponding to the error and integral of error. Then local coordinates α and β each need to be computed. Once the local coordinates are identified, the control function Q_O is calculated as a sum of the scaled shape. The control strategy requires specifying the control signal as a sum of scaled shape functions.

This section is a complement to the previous section of chapter 7, but it will focus on the five elements controller map instead of the nine elements map.

The location of global coordinates is determined to take advantage of controller studies, particularly the PI controller. Regarding the elements, after determining the total number of elements required, they need to be identified. Every single element is specified by four nodes placed in its corners. Each node is determined by the following formulae

$$V_b = V_1 + \left(\frac{V_3 - V_1}{h_3 - h_1} \right) (h - h_1) \tag{7.19}$$

$$V_t = V_7 + \left(\frac{V_5 - V_7}{h_5 - h_7} \right) (h - h_7) \tag{7.20}$$

$$V_B = V_4 + \left(\frac{V_2 - V_4}{h_2 - h_4}\right)(h - h_4) \tag{7.21}$$

$$V_T = V_6 + \left(\frac{V_8 - V_6}{h_8 - h_6}\right)(h - h_6) \tag{7.22}$$

To specify each single element, and to make it simple and easy to be understood, it is assumed that some points are placed as indicated in figure (7.13).

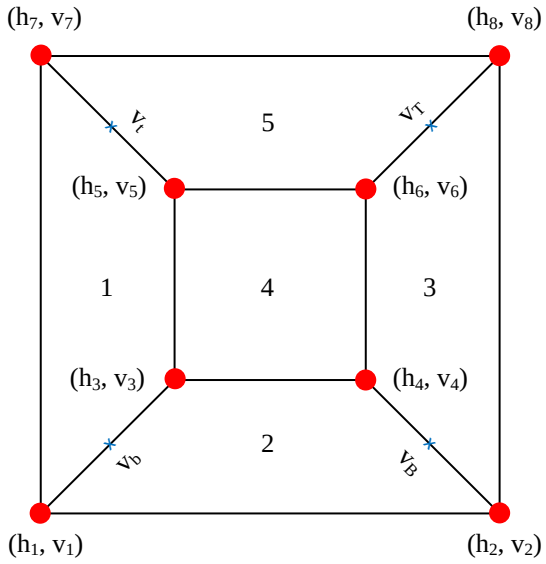


Figure 7-14: Four Node configurations for five discrete elements

For instance, when the error is greater than point V_b while the integral error is lower than a certain value, for example $V_3 = -0.15$, then the global coordinates will be considered to be located in the element numbered one. The same principle is repeated with the other four elements. Therefore, a logic statement is used to determine in which elements the

global error coordinates are located. After specifying the element, it is necessary to identify the local coordinates (α, β). In order to do that, the following formula is used:

$$E_0 = M_1 E_1 + M_2 E_2 + M_3 E_3 + M_4 E_4 \tag{7.23}$$

where E_0 represents the error that comes from Simulink and E_1, E_2, E_3, E_4 represent the error at each node located in the corners of all five elements.

In order to calculate the local coordinate α , it is necessary to plug in the corresponding value for each variable and then evaluate the expression. The expression of E_0 is given as:

$$\begin{aligned} \mathrm{E}_0 = {} & 1/4\left(1-\alpha-\beta+\alpha\beta\right)\mathrm{E}_1 + 1/4\left(1+\alpha-\beta-\alpha\beta\right)\mathrm{E}_2 \\ & +1/4\left(1-\alpha+\beta-\alpha\beta\right)\mathrm{E}_3 + 1/4\left(1+\alpha+\beta+\alpha\beta\right)\mathrm{E}_4 \end{aligned} \tag{7.24}$$

Combining like terms simplifies the polynomial expression given above. This creates the form

$$\mathrm{G} = \mathrm{E}_0 - 1/4\left(\mathrm{E}_1 + \mathrm{E}_2 + \mathrm{E}_3 + \mathrm{E}_4\right) \tag{7.25}$$

$$\mathrm{X} = 1/4\left(-\mathrm{E}_1 + \mathrm{E}_2 - \mathrm{E}_3 + \mathrm{E}_4\right) \tag{7.26}$$

$$\mathrm{Y} = 1/4\left(-\mathrm{E}_1 - \mathrm{E}_2 + \mathrm{E}_3 + \mathrm{E}_4\right) \tag{7.27}$$

$$\mathrm{Z} = 1/4\left(\mathrm{E}_1 - \mathrm{E}_2 - \mathrm{E}_3 + \mathrm{E}_4\right) \tag{7.28}$$

By replacing an expression consisting of multiple variables with a single given variable,

$$\mathrm{G} = \mathrm{X}\alpha + \mathrm{Y}\beta + \mathrm{Z}\alpha\beta \tag{7.29}$$

$$\alpha = \left(\mathrm{G} - \mathrm{Y}\beta - \mathrm{Z}\alpha\beta\right)/\mathrm{X} \tag{7.30}$$

The same method is used for integral error I_0:

$$I_0 = M_1 I_1 + M_2 I_2 + M_3 I_3 + M_4 I_4 \tag{7.31}$$

where I_0 represents the integral error that comes from Simulink and I_1, I_2, I_3, I_4 represent the integral errors at each node located in the corners of all five elements.

For the purpose of determining the local coordinate β, it is necessary to insert the corresponding value for each variable and then evaluate the expression. Expansion of I_0 gives the following formulae:

$$\begin{aligned} I_0 = {} & 1/4(1-\alpha-\beta+\alpha\beta)I_1 + 1/4(1+\alpha-\beta-\alpha\beta)I_2 \\ & + 1/4(1-\alpha+\beta-\alpha\beta)I_3 + 1/4(1+\alpha+\beta+\alpha\beta)I_4 \end{aligned} \tag{7.32}$$

$$H = I_0 - 1/4(I_1 + I_2 + I_3 + I_4) \tag{7.33}$$

$$U = 1/4(-I_1 + I_2 - I_3 + I_4) \tag{7.34}$$

$$V = 1/4(-I_1 - I_2 + I_3 + I_4) \tag{7.35}$$

$$W = 1/4(I_1 - I_2 - I_3 + I_4) \tag{7.36}$$

$$N = U\alpha + V\beta + W\alpha\beta \tag{7.37}$$

$$\beta = (N - U\alpha - W\alpha\beta)/V \tag{7.38}$$

The N of equation (7.37) has two unknowns, α and β. Once the local coordinates α, β are known, the control function Q_O is computed as follows:

$$Q_O = \sum M_i Q_i \tag{7.39}$$

In other words, Q_O is the sum of the scaled shape, where Q_i are control nodal values, and M_i is a shape function.

7.10.2 Five Discrete Elements with Eight Nodes

There are five discrete elements which are used with eight nodes placed in the corner and the mid-sides of a quadrilateral element. Each node in an element corresponds to a shape function which has a unit value at that node and 0 at all other nodes. Local coordinates (α, β) are used to define the shape functions.

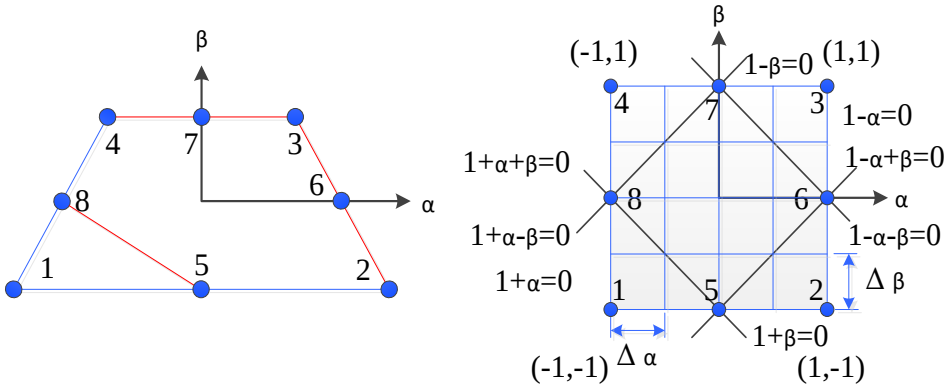


Figure 7-15 Element geometry and local coordinate system for eight node quadrilateral configration

The image on the left side of Figure 7-14 illustrates the element geometry and node configuration of the quadrilateral elements. The image on the right depicts the local coordinate system for an 8 node quadrilateral element. The relationship between the local coordinates (α, β) and global coordinates (E, I) is given by the following equations:

$$E(\alpha,\beta)=\sum_{i=1}^{8}N_iE_i \tag{7.40}$$

$$I(\alpha,\beta)=\sum_{i=1}^{8}N_iI_i \tag{7.41}$$

Writing the sum in expanded form gives

$$E(\alpha,\beta)=N_1E_1+N_2E_2+N_3E_3+N_4E_4+N_5E_5+N_6E_6+N_7E_7+N_8E_8 \tag{7.42}$$

$$I(\alpha,\beta)=N_1I_1+N_2I_2+N_3I_3+N_4I_4+N_5I_5+N_6I_6+N_7I_7+N_8I_8 \tag{7.43}$$

where E and I are the error and integral error anywhere on the element, N_i is the shape function, E_i and I_i are the error and integral error at the node i within the element, and error and integral error nodal values are unknowns which should be determined from the discrete global equation system.

The Lagrange shape functions for an 8-noded quadrilateral element are quadratics. The corresponding shape functions are

$$\left.\begin{aligned}
N_1&=1/4(1-\alpha)(1-\beta)(-\alpha-\beta-1)\\
N_2&=1/4(1+\alpha)(1-\beta)(\alpha-\beta-1)\\
N_3&=1/4(1+\alpha)(1+\beta)(\alpha+\beta-1)\\
N_4&=1/4(1-\alpha)(1+\beta)(-\alpha+\beta-1)\\
N_5&=1/2(1-\alpha^2)(1-\beta)\\
N_6&=1/2(1+\alpha)(1-\beta^2)\\
N_7&=1/2(1-\alpha^2)(1+\beta)\\
N_8&=1/2(1-\alpha)(1-\beta^2)
\end{aligned}\right\} \tag{7.44}$$

Expansion gives

$$E(\alpha,\ \beta) = a_1 + a_2\alpha + a_3\beta + a_4\alpha\beta + a_5\alpha^2 + a_6\beta^2 + a_7\alpha^2\beta + a_8\alpha\beta^2 \quad (7.45)$$

$$I(\alpha,\ \beta) = b_1 + b_2\alpha + b_3\beta + b_4\alpha\beta + b_5\alpha^2 + b_6\beta^2 + b_7\alpha^2\beta + b_8\alpha\beta^2 \quad (7.46)$$

Where a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8, b_1, b_2, b_3, b_4, b_5, b_6, b_7 and b_8 are eight generalized degrees of freedom of the element.

The given shape functions are inserted into the formulae (7.37) and (7.38), and simplified by grouping all like terms together in order to calculate the local coordinates $(\alpha,\ \beta)$.

To solve the challenge under investigation, the following steps of the FECMM are employed. First the full region of the relevant error and integral error variables is discretized into quadrilateral elements. This can be accomplished by many controller techniques, but in the current study the conventional PI controller is used to define the full region. The developed algorithm generates the finite element grid. The grid is made up of several arrays, of which the principle arrays are nodal coordinates. Element connectivity arrays in the counter clockwise direction are used for the assembly process. Second, to interpolate the field variables within the element, designated interpolation functions in polynomials are employed. The polynomial's order is determined according to the number of nodes assigned to the element. Third, a logic statement is used in the developed program to determine in which element the field variable is located. After some manipulation to define all the coefficients required by the used polynomial function, the local coordinates $(\alpha,\ \beta)$ have to be defined.

Finally, in order to determine the control action, all the shape functions at each node must be combined. Iterative methods are used in order to modify and update the local and

global coordinates (α, β), (E, I), because they do not require much computational burden (time) along with fast convergence, and are therefore preferable.

The color contours in figure 7-15 illustrates the geometry of the five element configuration for curve sides with eight nodes. These are employed to develop the algorithm. This could also be beneficial in generating a more complicated map with more degrees of freedom. The three dimensional (3-D) is illustrated in figure 7-16.

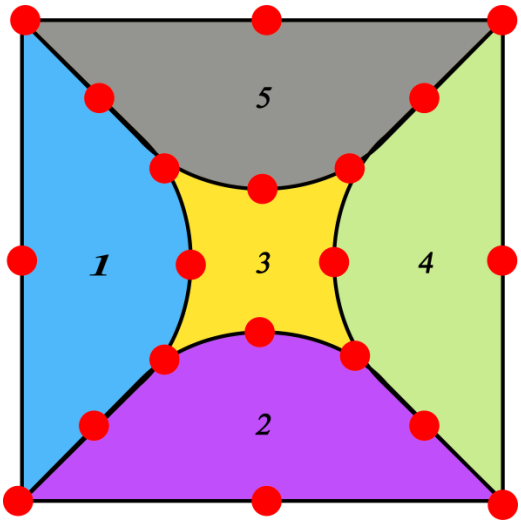


Figure 7-16: Five Element for an eight node curve sides quadrilateral

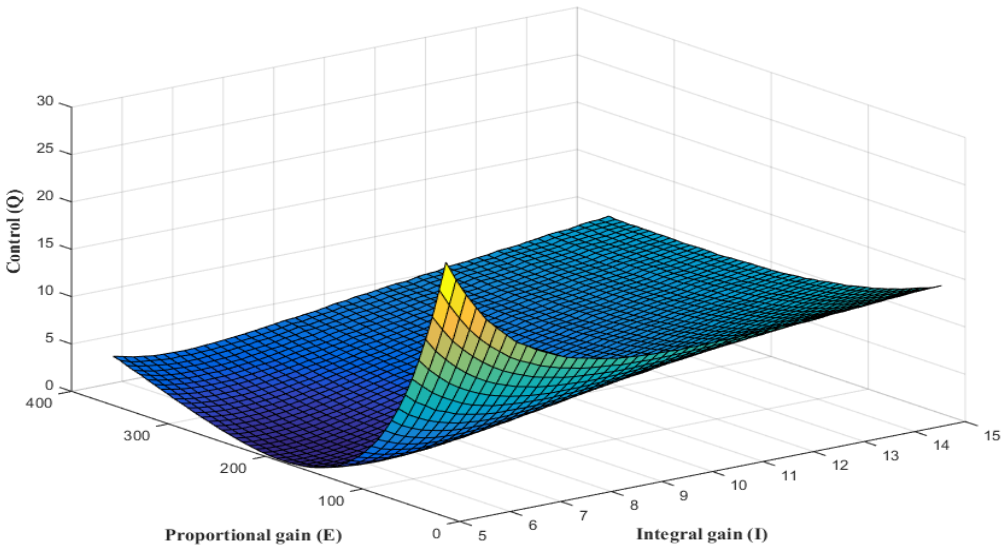


Figure 7-17: Finite element controller map (FECM) with eight nodes.

The FECM is used in the proposed application to control the induction motor drive, and to confirm the validity of efficient algorithm. The proposed algorithm is first used to mimic PI controllers, which require fixed nodes to be constrained to follow precise speeds. Furthermore, numerous simulation tests have been applied to determine the algorithm's ability to lend itself towards being developed as an self-tuning controller. This test is run based on the integrated squared difference over a set period of time. To confirm the self-tuning capacity of this controller, arbitrary values are typically assigned to two nodes located on the corner of the middle element, which have a higher probability of being subjected to error. These nodes have been chosen because they are more sensitive due to the fact that they are connected to two other elements, along with their location in the element which takes place in the middle of the solution region, making them more sensitive to error.

In order to compare tracking speed at fast convergence rates, a PI controller-based IM drive system has also been employed and tested. Additionally, drives at different dynamic operating conditions have been assessed for performance by conducting a series of simulation tests.

In order to validate the efficacy of the proposed approach, simulation and laboratory experimental results are presented and discussed.

Through the use of this proposed technique, high performance speed tracking is achieved. This technique can also be upgraded to develop the algorithm for offline tuning to adjust the element nodes' values for sudden changes in various operating conditions.

7.11 Grid Search Optimization

The finite element controller map method is integrated with the grid search method. In this proposed work, it is used to determine the optimum values of the nodal (Q_i) for different types of speed control and to continually adapt these nodal values automatically in real-time. To minimize learning time and serve as a starting point, initial gains are given to the nodal values corresponding to the minimum value of the objective function over the grid, and then step size is set to arbitrary values for the nodal d_3 and d_6, respectively. Figure 7-17 provides the flowchart of how the program is organized to adjust two nodes.

The algorithm is relatively simple to program compared to other optimization algorithms, and also has the advantages of fast convergence and no requirement of accurate knowledge of the system model. However, it is very inefficient in that it requires extended computational time.

The performance of the proposed grid search method to optimize specified corner nodes with vector control is simulated and compared with that of the conventional fixed gain proportional-integral controller under different speed commands, repetitive operations, parameter variations, and load disturbances. The computer simulation and experimental laboratory results are given to demonstrate the effectiveness of the proposed PID learning controller, showing its superiority to the conventional fixed PI controller.

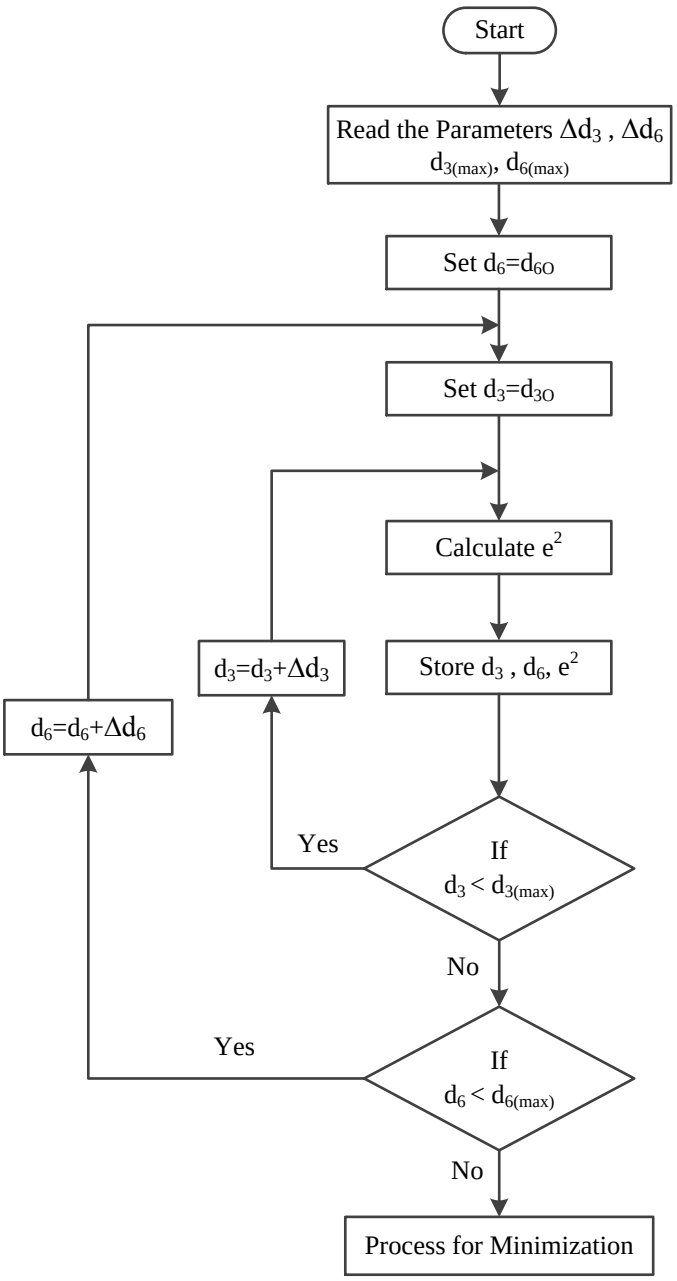


Figure 7-18: Flowchart of adjusting nodes using grid search optimization method

Initial numbers are given to node three $(d3)$ and node six $(d6)$, for instance, with value 1 and 2 going to these nodes respectively. After running the system, the desired response will be compared with the actual output response generated by the five elements controller map method. The resulting error will be squared and integrated through the

objective function, and then stored as data. This process will be repeated numerous times. The eventual data size will be determined by the step size of the grid. For each step, the nodal value of d3 and d6 of the element varies based upon the corresponding amount of error. Once this process has been repeated sufficient times, the nodal will attain a value that results in the least error and makes the actual output converge and track the desired output.

The optimum value, which has been attained from the process previously described, is then plugged into the system, rather than a random value, to test the system response.

Additionally, another test is applied to further examine the system response when it is being run by the five elements. In this test, instead of using a reference system, the input step command is directly fed to the actual system, and the error between the step command and actual system response is again squared and integrated. Then the same sequence described in the flowchart is repeated many times to compare the optimum value obtained by each test.

7.12 Five Element Results with four nodes for each element

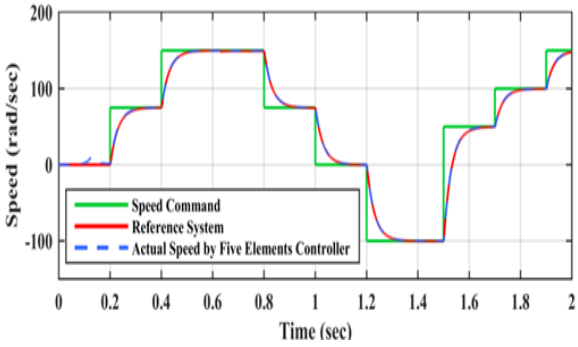


Figure 7-19: Initial speed tracking response with FECMM without load

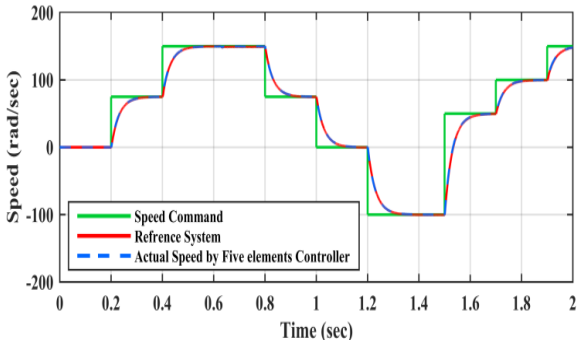


Figure 7-22: Speed tracking response with FECMM controller without load

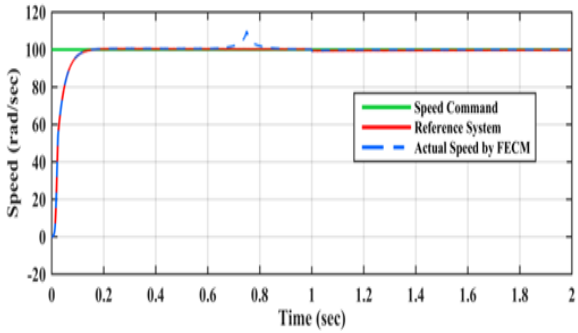


Figure 7-20:. Initial speed tracking response with FECMM for load (2 Nm)

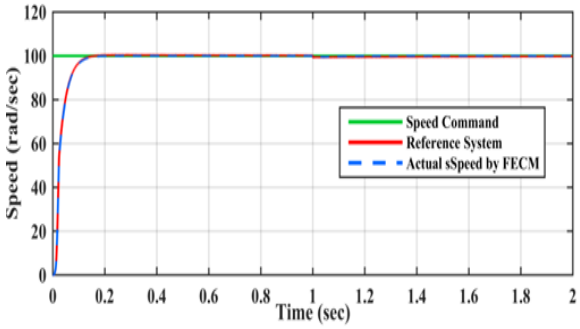


Figure 7-23: Speed tracking response with FECMM controller for load (2 Nm)

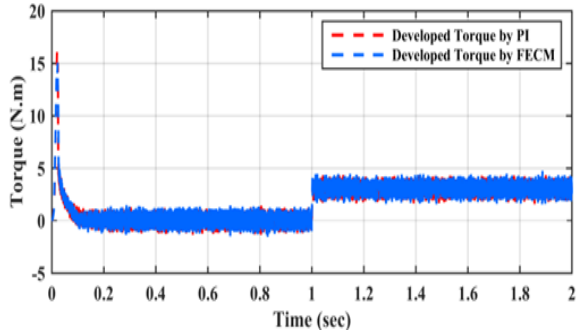


Figure 7-21: Initial motor torque corresponding to the speed given in figure 7-18

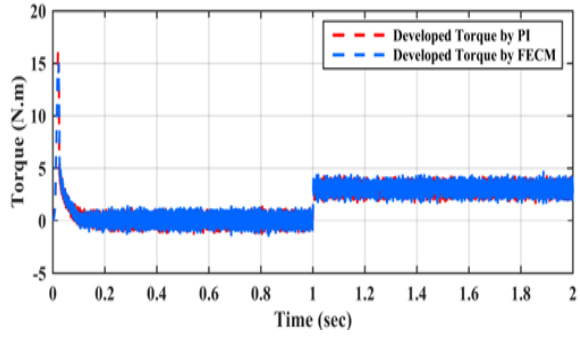


Figure 7-24: Developed motor torque corresponding to the speed given in figure 7-21.

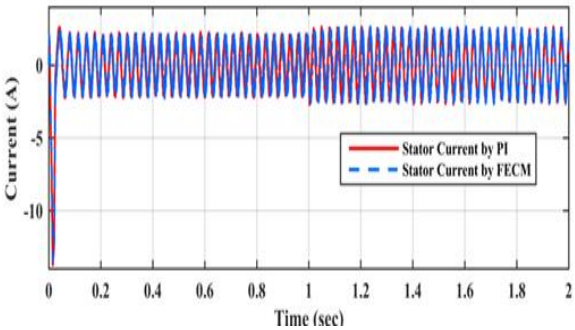


Figure 7-25: Initial stator current corresponding to the speed given in figure 7-18

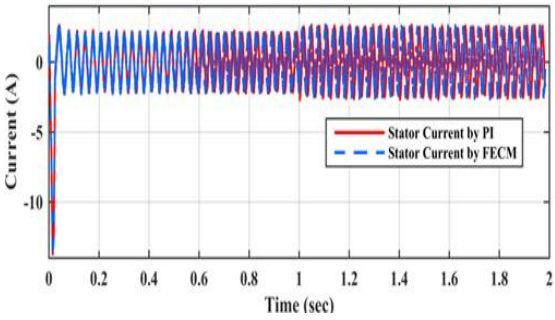


Figure 7-26: Stator current corresponding to the speed given in figure 7-21

In order to evaluate the proposed algorithm, a series of simulation tests was carried out on an indirect vector controlled induction motor drive using both a conventional tuned PI controller and FECMM controller combined with the grid search method to find and automatically adapt the optimum controller corner nodes. The speed and current responses of the induction motor have been observed under different operating conditions, such as sudden change in reference speed and step change in load.

Figures 7-18 and 7-19 show the speed of the proposed technique in the initial stage and have fluctuations. This represents the response of the algorithm at initial start-up, using semi-arbitrary gains and before the controller has had time to execute its self-tuning process. Performance then improves gradually before reaching optimization, as illustrated in figures 7-21, and 7-22, where motor speed is shown to converge to the desired speed. It is also clear from the figures that the algorithm has high precision in tracking the command speed and is able to recover quickly from load disturbances.

Figure 7-20 and 7-23 is the torque response obtained by the developed algorithm. However, these two responses seems almost same. These two torques correspond to the same speed command. However, two nodes were initially set to an arbitrary plus and minus value, after a certain amount of time the system response was gradually improved. That is why the simulation results obtained for both developed torques corresponding to the same speed command were quite close in their response.

Figure 7-24 shows that a small disturbance of stator current occurred at the initial startup of the motor from standstill. At step change, the current response also fluctuated significantly but recovered within a short time, and at the sudden load disturbance there

was also current fluctuation. These disturbances are eliminated after the self-tuning process, as shown in figure 7-25. Along with these figures 7-21 and 7-22 show that after the self-tuning process, the controller has reached a performance standard at least equal to that of an optimally tuned fixed gain PI controller. The GSM based controller has located the optimum proportional and integral gains corresponding to the minimum objective function over the grid for the self-tuning PI controller.

7.13 Five Element Results with Eight Nodes

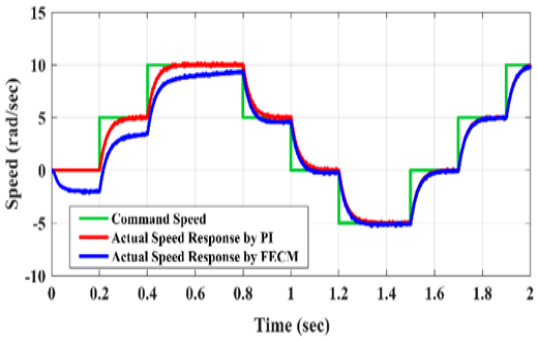


Figure 7-27: Initial start of speed tracking response for PI & FECM controllers without load

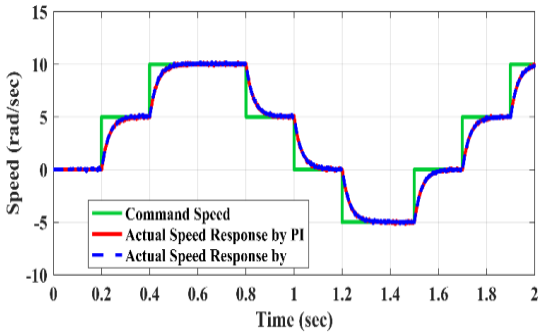


Figure 7-28: Simulation response for speed tracking of PI & FECM controllers at the end of the time period without load

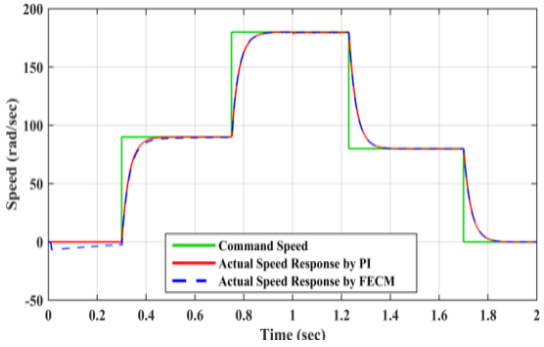


Figure 7-29: Initial start of speed tracking response for PI & FECM controllers with load torque (2 Nm)

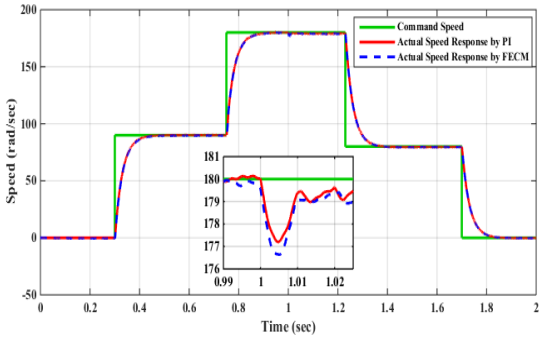


Figure 7-30: response for speed tracking of PI & FECM controllers at the end of the time period with load torque (2 Nm)

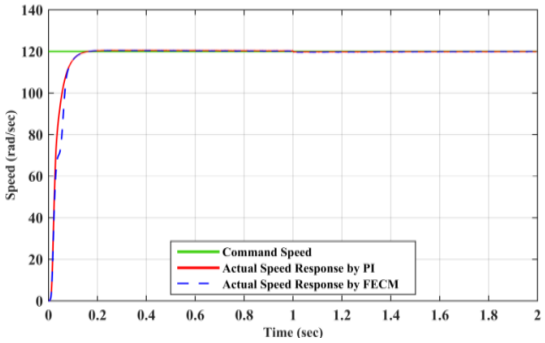


Figure 7-31: Initial speed tracking response for PI & FECM controllers with load torque (1 Nm) of full rated load

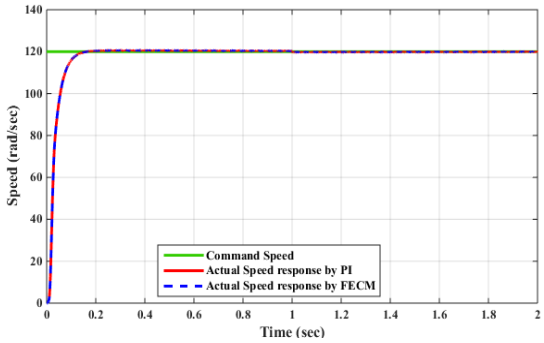


Figure 7-32: Simulation response for speed tracking of PI & FECM controllers at the end of the time period with load torque (1 Nm)

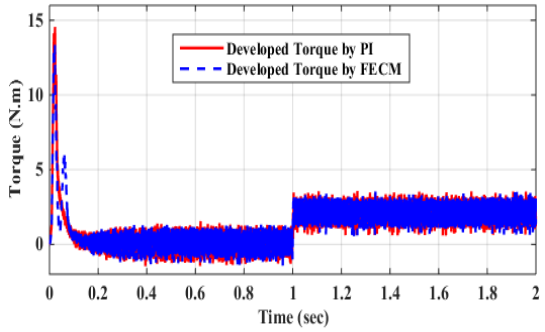


Figure 7-33: Simulated torques of the PI & FECM controllers corresponding to the speed command given in figure 7-30 under the load torque (1 Nm)

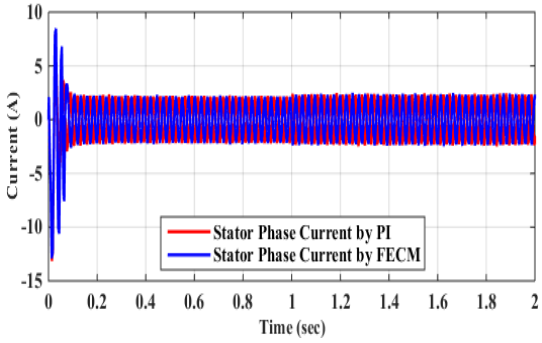


Figure 7-34: Simulated currents of the PI & FECM controllers corresponding to the speed command given in figure 7-30 under the load torque (1 Nm)

The speed control of the vector control in figure 7-4 is achieved using an FECM algorithm. The conventional PI controller is replaced by the FECM and the fixed eight nodes located on corners and mid-sides of a quadrilateral element are assigned based on corresponding PI gains in order to mimic PI control. The aim of using FECMM is to create an intricate map of the output response. To train a controller based on the integrated squared difference over a set period of time, it is preferable to develop a model

of the system and train it before uploading the controller to the actual system. For online training it would be best to select the most sensitive parameters based on simulation and only gradually adjust those parameters step by step in time. With fixed nodes these inputs would seem to produce control which is proportional-like and integral-like, respectively.

For comparative evaluation purposes, a PI controller-based IM drive system has been carried out and tested. Furthermore, extensive simulations have been done in order to evaluate the performances of both the FECM and PI techniques at different operating conditions. The performances of the IM drive for both the conventional and proposed speed controllers are investigated for step changes in command speeds and load torque.

Simulation tests have been conducted on a vector control for an IM drive, using a traditional PI controller and an FECMM controller, to evaluate the effectiveness of the proposed algorithm based speed controller for the IM drive. The IM speed and current response have been investigated under conditions including sudden change in step reference speed and step load change. It is obvious from figure 7-26 that there is no significant difference between the traditional PI controller and the proposed technique during the motor's performance without load, and that both techniques accurately follow the desired speed. Further, when load test is applied to the motor, both controller and the proposed algorithm can track speed commands when rated load is applied to the IM, as seen in figures 7-28, 7-30. There is only a slight dip in the speed performance at the moment of applying load, and the only difference is the time to recover from load disturbance, but both schemes converge to follow the desired speed. To examine the

robustness and adaptability of the developed algorithm, simulation tests are run by giving arbitrary values to two nodes. The results of this test, as shown in figures (7-27) and (7-29) indicate the capability of this technique to converge to the desired speed without issue. Initially, it can be observed from the figure that the algorithm does not discretely converge to the correct speed, and requires a slight extra rise time to do so. This is because an arbitrary value was given to the two nodes, as already mentioned. Figure (7-32) and (7-33) show a developed motor torque and stator phase current corresponding to the step speed command that has been shown in figure (7-29). The preceding simulation results reflect the features of self-tuning the finite element controller map.

From these results, it is clear that the FECMM controller has been successfully tuned, via the off-line PI-referenced method presented in this paper, to exhibit performances equal to those obtained with the conventional PI controller upon which it is based. The results show that under the conditions presented here, the FECMM controller is, in fact, producing control signals in agreement with the reference PI controller.

7.14 Experimental Results and Discussion

To minimize complexity and computational capacity requirements, the output of the control (Q) is the summation of all scaled shape functions used as indicated in equation (7.6) Speed error can be used at each iteration to modify the local and global coordinates α, β, error (E) and integral speed error (IE) to produce the precise command torque to minimize error. The drive signals for the inverter power switches are derived from the output signals of the controller. In the current controller, the three-phase current commands (i^*_{abc}) are compared with the actual stator currents (I_{abc}), and then the

resulting errors are used to switch the PWM inverter. The output of the inverter is supplied to the stator of the induction motor.

To evaluate the effectiveness of the proposed FECM algorithm, a series of simulation tests are conducted and observed under different operating conditions using MATLAB/SIMULINK. A picture of the experimental setup for an IM drive using DSP board DSS1104 is illustrated in figure 7-34. In addition, real time experimentation is carried out to confirm the efficacy of the developed algorithm for high performance industrial ac drives. To evaluate the robustness of the proposed approach, the FECMM is investigated under a sudden change in command speed and then the approach is verified by carrying it out experimentally. It should be noted that the impact of sudden load change on the speed performance of the proposed algorithm based IM drive was determined solely by computer simulation, as illustrated in figures 7-28, and 7-30.

It is obvious from the experimental results demonstrated in figures 7-37 and 7-38 that performance of the proposed algorithm FECMM precisely follows the command speed. However, while it currently experiences a slight delay, the aim is to soon develop a still faster conversion rate. Generally, the results show that under the conditions presented here, the FECMM controller is, in fact, producing satisfactory control signals.

7.15 Experimental Drive Configuration

The control algorithm in this work is executed through the dSPACE DSP board DSS1104, connected into a PC computer. The development software of the algorithm works in a Matlab/Simulink environment. It is composed of two principle components. Real Time interface (RTI) is its implementation software. The RTI is a Simulink toolbox. It enables the configuration of the model by providing model blocks, which permit users to access the dSPACE hardware. The IM parameters are listed in the table in Appendix A. The IM is supplied by a three phase full bridge inverter using six IGBT's and a gate driver board. The control is done using dspace-DS1104 control board, which is interfaced with a personal computer (PC) through the control panel. This panel has several instrument devices such as digital to analog (D/A) converters which are required to read the motor currents, analog-to-digital (A/D) converters to generate PWM pulses that are needed to fire the gate of the IGBT, and position encoder interfaces. It also provides the required digital input/output (I/O) ports and timer function such as input and output captures, and generation of inverter pulses. The inverse Park's transformations are used to attain the three-phase reference motor currents from the reference direct and quadrature axis currents. The motor currents are measured by the current transducers and represent the input to the DSP control board. The hysteresis current controller uses the difference between the actual motor currents and the equivalent reference motor currents to create the three-phase PWM pulses to trigger the bridge inverter IGBTs. The output voltages from the inverter are used to supply the IM with appropriate voltages and frequency corresponding to the operating condition. An incremental encoder position

sensor delivering 2500 pulses per revolution is installed on the IM rotor shaft to detect rotor position and speed. This is then fed back to the DSP board through an encoder interface. The DC voltage required for VSI (VSI) is obtained from a three phase diode bridge rectifier. To reduce voltage ripple, a capacitive filter is connected at the end terminal of the DC link. The Digital to Analog (D/A) channels are used to deliver the desired signals to the oscilloscope. The DC machine can be coupled to the IM to represent a load applied to the motor. The sampling frequency utilized in this application is 15 kHz.

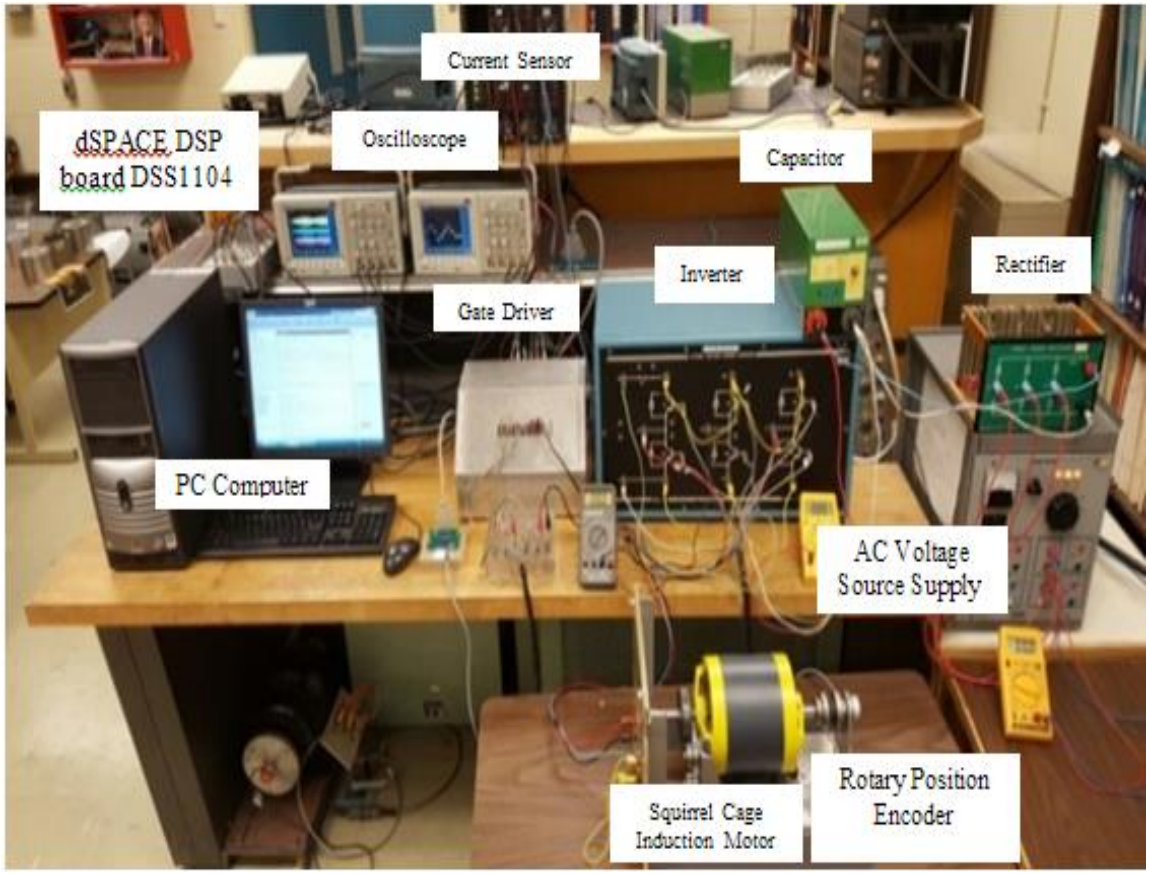


Figure 7-35 Experimental setup for IM drive using DSP board DSS1104

7.16 Experimental Results and Discussion

In order to evaluate the effectiveness of the proposed FECMM algorithm, a series of simulation tests are conducted and observed under different operating conditions using MATLAB/SIMULINK. In addition, real time experimentation is carried out to confirm the efficacy of the developed algorithm for high performance industrial ac drives. Additionally, a comparative performance analysis between the proposed scheme and a traditional PI controller is provided to emphasize the validity and robustness of the algorithm. The results obtained for the FECMM controller with five elements and eight nodes located around each element are shown in figures 7-38, 7-39, and the result for the PI controller is shown in figures 7-36, 7-37. To evaluate the robustness of the proposed approach, FECMM is investigated under a sudden change in command speed and then verified by carrying it out experimentally. It should be noted that, the effect of sudden load change on dynamic speed performance and the evaluation of speed with parameter variation of the proposed algorithm based IM drive were performed solely by computer simulation, as illustrated in Figures 7-29, 7-31. It is obvious from these experimental results that the performance of the FECMM follows the command speed more accurately compared the performance of the PI controller. Additionally, the FECMM has faster convergence to the desired speed and has no overshoot in contrast with the PI controller. Generally, the results show that under the conditions presented here, the FECM controller is, in fact, producing control signals in agreement with the reference PI controller, and better than the PI experimentally.

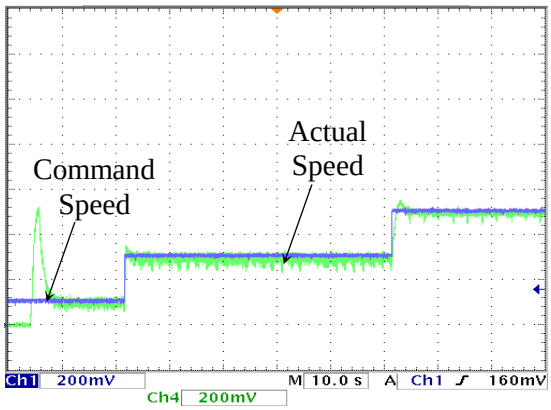


Figure 7-36: Experimental speed response of the drive for step changes of command speed based on PI controller, (div=10 rad/sec)

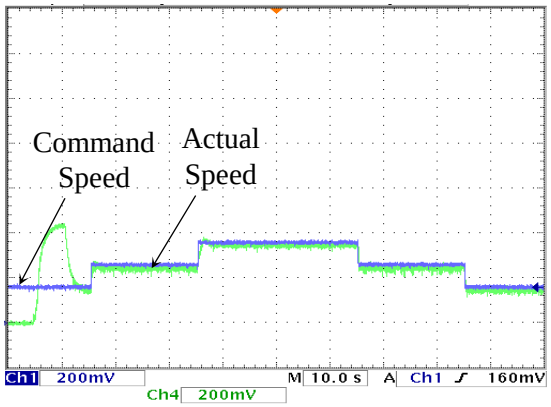


Figure 7-37: Experimental speed response of the drive for the step changes of the command speed based on based on PI controller, (div=10 rad/sec)

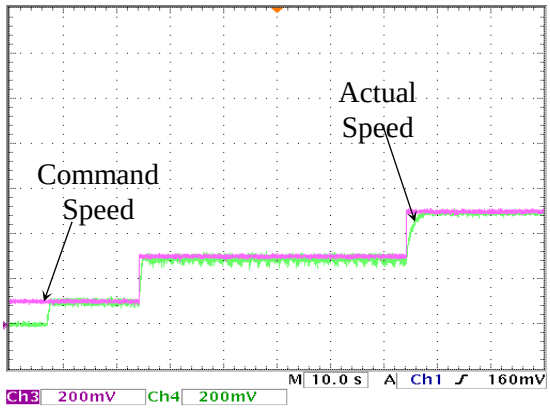


Figure 7-38: Experimental speed response of the drive for step changes of command speed based on FECM controller for five element, (div=10 rad/sec)

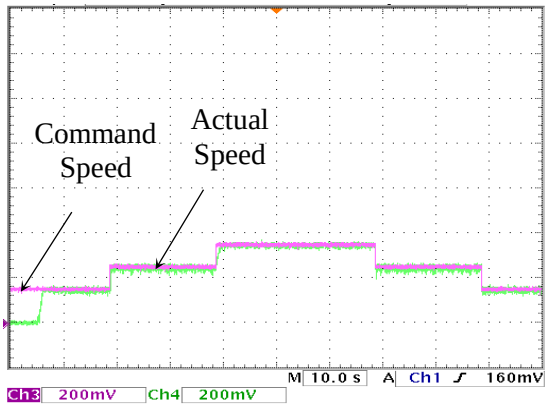


Figure 7-39: Experimental speed response of the drive for the step changes of the command speed based on FECM controller for five element, (div=10 rad/sec)

Other results are obtained when the FECM controller has nine elements and four nodes are placed at the corner of each element. After initializing the required variables, the initial set of nodes (d3, d6) acquired from the knowledge of the system were downloaded. The results obtained for the FECM controller are shown in figures 7-39 and 7-40. This work evaluated the performances of the proposed FECMM based drive system.

The results show that the proposed drive system has the ability to follow the reference speed under no load conditions. Iinitially, there is difference between command and actual speeds, this is delay caused by power was supplied manually to the setup by use of the switches.

It should be mentioned that the effect of sudden load impact on dynamic speed response and the evaluation of speed with parameter variation of the FECMM based IM drive were performed solely by computer simulation, as depicted in figures 7-7, and 7-10. It can be noted for these experimental results that the dynamic performance of the FECMM controller shows that the transient associated with startup reveals a delay that seems to occur before the actual speed begins to track speed command. Then the transient gradually disappears so that there is a good agreement between actual speed and speed command. The disturbance caused by the transient might be the result of motor inertia. The results show that under the conditions presented here, the FECMM controller is, in fact, providing satisfactory performance.

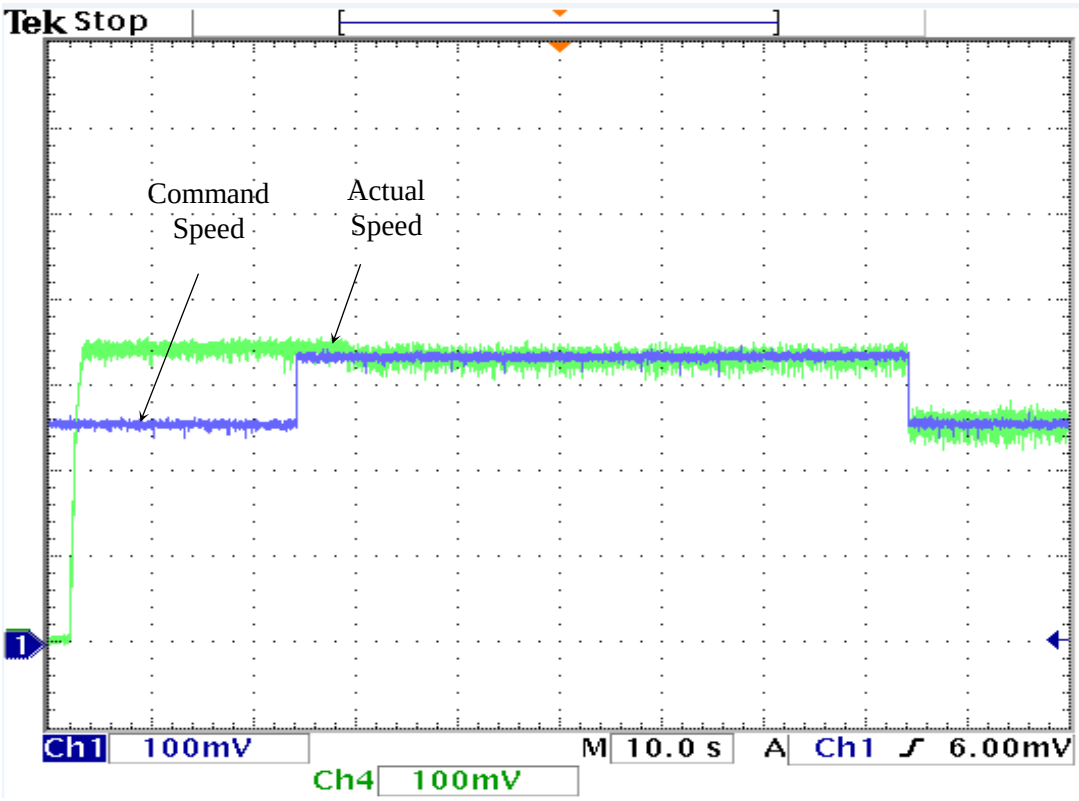


Figure 7-40: Experimental speed response of the drive for the step changes of the command speed based on FECM controller for nine element, (div=20 rad/sec)

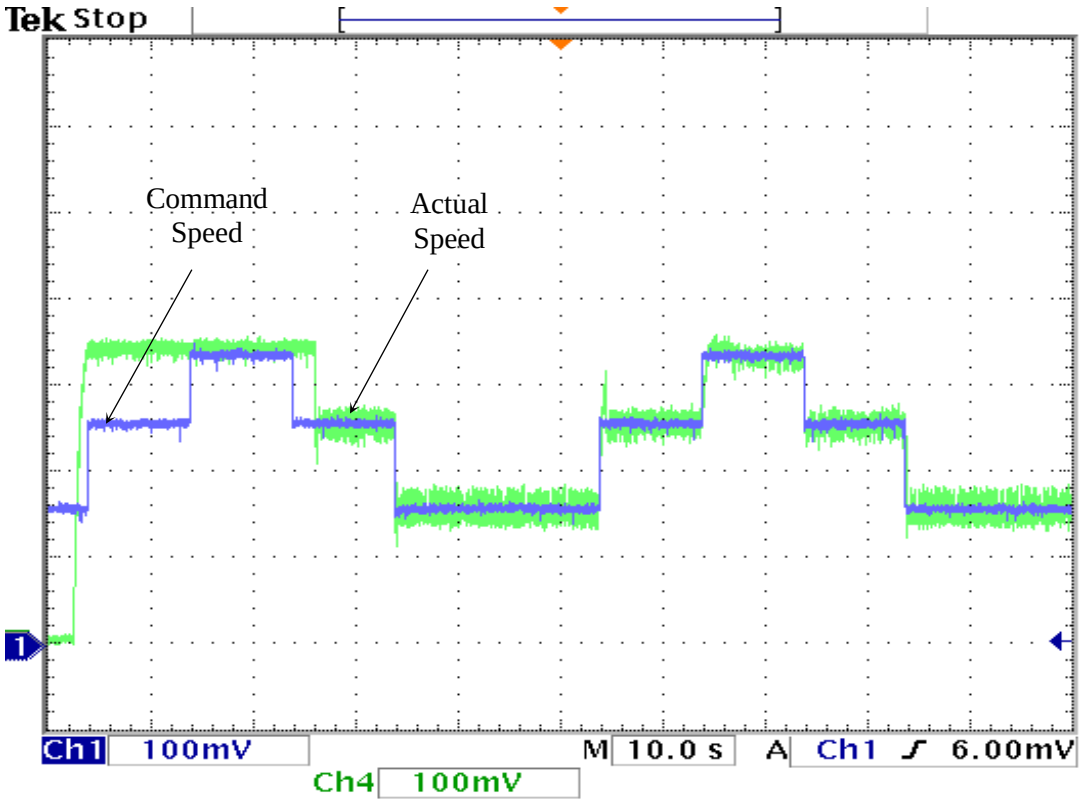


Figure 7-41: Experimental speed response of the drive for the step changes of the command speed based on FECM controller for nine element, (div=20 rad/sec)

7.17 Conclusion

This current work contributes to the development of a novel controller scheme for IM drives based on an algorithm using higher-order polynomial functions. The algorithm creates a Finite Element Controller Map (FECM) of the output response of an IM drive system. The polynomial's order and map complexity is determined according to the number of nodes assigned to the discrete element. The proposed control scheme achieves high accuracy motion-tracking performance in a wide range of speeds, from low to high even in the presence of load disturbance. In addition, the approach provides fast tracking convergence and simple formulation functions. A comparison with the classical PI controller is provided to highlight the superior performance of the proposed approach. Furthermore, because the algorithm creates a complex map, it has the ability to go beyond the PI controller. Consequently, the potential for adaptation is higher, as is the potential for greater efficiency performance, including stability and disturbance load rejection. Both simulation and laboratory test results are given to verify the validity of this method.

7.18 Chapter Summary

This chapter introduces a novel control approach called the Finite Element Control Map. This map is characterized by multiple degrees of freedom and local character. The complexity of the map's shape depends on the number of nodes assigned within the discrete finite elements. An application of these techniques is applied to an industrial induction motor drive as a comparison to using the traditional PI controller. The developed algorithms are applied to the same application by two different strategies. The difference between the two strategies is the number of elements that are employed: one uses nine elements, while the other one uses only five. To evaluate these techniques, a series of simulation tests are conducted under a variety of operating conditions. The simulation results demonstrate the effectiveness and robustness of the proposed algorithm. A comparison with other conventional PI controllers shows that the proposed algorithm has achieved significantly better results for all studied cases. Additionally, these techniques are implemented in the laboratory and achieve efficient performances.

Chapter 8.

Comparative Simulation Results between Conventional PI, Fuzzy logic, Neural Network and Finite Element Controller Map Based IM Drive

As mentioned early in sections 5.6, 6.5.2 and 7.9.1 that simulation results attained using a conventional PI controller and developed controller algorithms (FL, NN, FECM) were very close. However, with unchanged nonlinear IM model and parameters used throughout the comparisons, a series of simulation tests are conducted to check and evaluate the performance of the IM drive when operated by different controllers, which include Conventional PI, Fuzzy logic, Neural Network and Finite Element Controller Map Based IM Drive. The performance response of the IM drive for all four controllers are compared and investigated under step changes in command speeds and load torque. After analyzing figures 8-1, 8-2 and 8-3, it is clear that developed controllers conventional PI, Neural Network and Finite Element Controller succeeded in converging and tracking speed command. In contrast, from the same figures, fuzzy logic controller shows satisfactory simulation results during no load applied to the motor drive, and exhibits steady state errors when the motor speed is stepped down, and this is noticeable when the load is applied.

It was also noted that the difference between the performances achieved through these controllers are rise time, falling time and the fluctuations dip caused by the moment when the load torque is applied.

Firstly, the rise time is defined during the speed response rise from 10% to 90% of the final speed response. Secondly, the rise time is also observed during the speed change from 100 to 180 rad/sec.

Table 8-1 shows the rise time for conventional PI, fuzzy logic, neural network and finite element controllers. Though in this application, there is no considerable variation in the rise time between them,

Table 8-1: Rise time for conventional PI, fuzzy logic, neural network and finite element controllers

Controller Type	Rise Time (sec) 0.0-100	Rise Time (sec) 100-180
Conventional PI	0.0659	0.0644
Fuzzy Logic	0.0677	0.0721
Neural network	0.0657	0.0645
Finite Element Controller Map	0.0659	0.0644

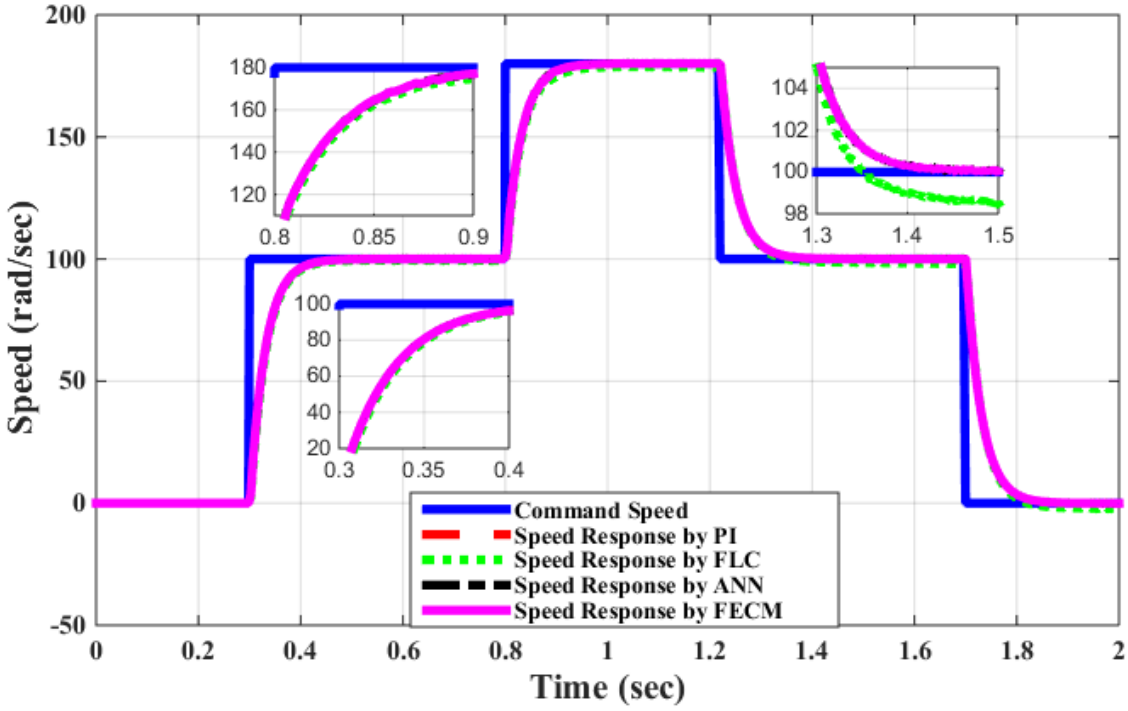


Figure 8-1: Speed tracking responses noload, 180 rad/sec

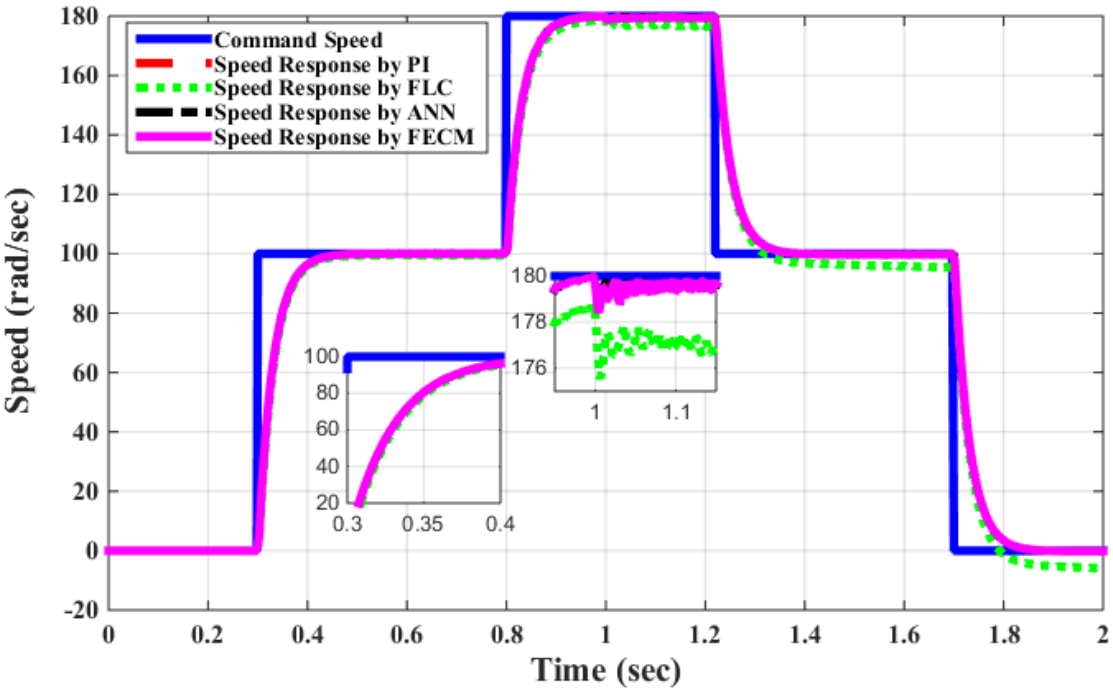


Figure 8-2: Speed tracking responses at applying load 2 N.m, 180 rad/sec

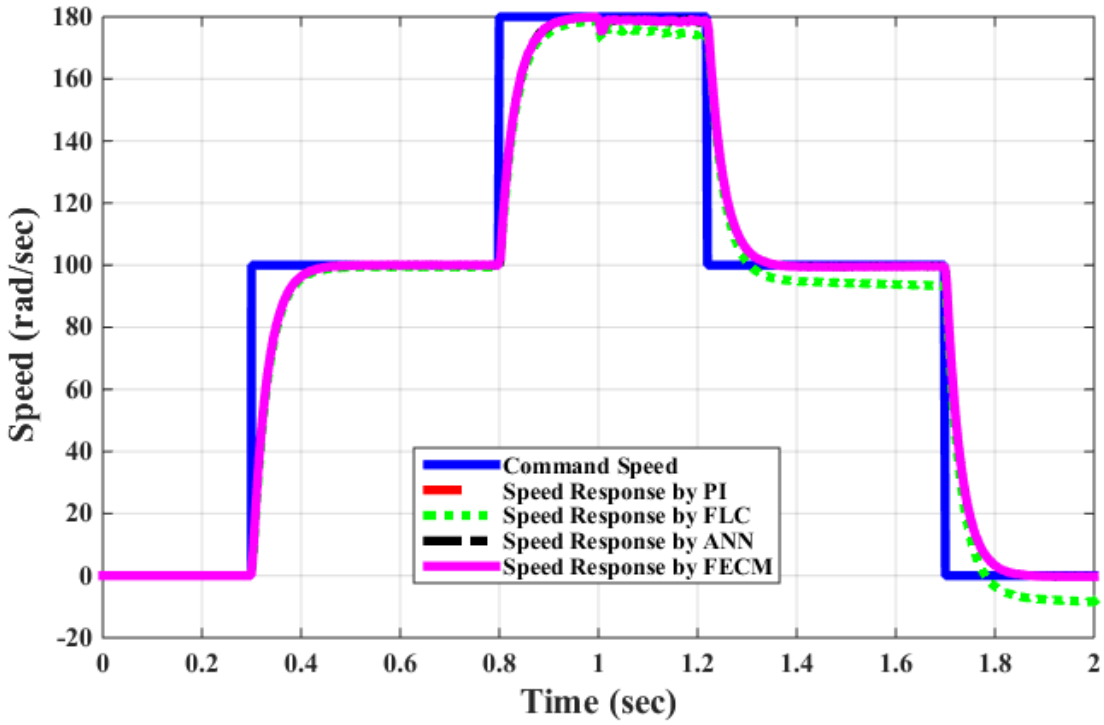


Figure 8-3: Speed tracking responses at applying load 4 N.m, 180 rad/sec

Besides that, the settling time (t_s) is also defined as the time after the actual speed response remains within 5% of the command speed. It is determined first in the period when the speed command changes from 0.0 to 100 and then, defined after from 100-180. During this period the settling time (t_s) for conventional PI, fuzzy logic, neural network and finite element controllers are indicated in the table 8-2.

Although there was a very little difference in settling times between all the controllers, the finite element controller and conventional PI controller are the same settling time,

they take less time than the other controllers. The neural network was next to follow and finally the fuzzy logic.

Table 8-2: Settling time for conventional PI, fuzzy logic, neural network and finite element controllers

Controller Type	Settling Time (sec) 0.0-100	Settling Time (sec) 100-180
Conventional PI	0.390	0.889
Fuzzy Logic	0.395	0.910
Neural network	0.390	0.892
Finite Element Controller Map	0.390	0.889

In addition, the FL controller response under transient load conditions exhibits a large dip fluctuation caused by load unlike the other developed controllers; this is obviously clear in figures 8-2 and 8-3.

Moreover, in order to evaluate between these simulation results, Root Mean Square Error (RMSE) was used. RMSE is often used as a measure of the deviation between sample data attained from the actual motor drive and the data speed response obtained from reference PI. These deviations are called residuals. The calculations are performed over the data sample that was collected for estimation.

$$RMSE = \sqrt{\frac{\sum (R - R_{PI})^2}{n}} \quad (8.1)$$

RMSE is always positive and shows the accuracy of fit to the data. A lower RMSE indicates a higher accuracy.

Table 8-3: Root Mean Square Error for Fuzzy logic, neural network and Finite element controllers

Controller Type	No load	Load Torque 2.0 N.m	Load Torque 4.0 N.m
Fuzzy Logic	1.316	2.062	3.90
Neural network	0.062	0.099	0.11
Finite Element Controller Map	0	0	0

The RMS results show that the error between the Finite Element controller and the PI controller is zero whereas for the Neural Network controller the error is small while for the Fuzzy Logic controller it is large.

A Simulink diagram was created that ran PI, Fuzzy Logic, Neural Network and Finite Element simulations in parallel. Each simulation had the same blocks for the system and the same command and disturbance. The only difference between each simulation was the controller. Each controller had the same saturation limits. With known saturation

limits and known PI gains, the PI equations were used to find the error and the integral of the error where the controller hit the saturation limits.

The control signal equation for a conventional PI controller is

$$Q = K_P E + K_I I \tag{8.2}$$

where

$$I = \int E \, dt \tag{8.3}$$

The error at onset of saturation E_S and integral of error at the onset of saturation I_S

$$E_S = Q_{PS} / K_P \tag{8.4}$$

$$I_S = Q_{IS} / K_I \tag{8.5}$$

where the control signal saturation levels Q_{PS} and Q_{IS} as well as the gains K_P and K_I are known. Plus and minus these PI saturation values were used to set the corner node locations of the center element of a 9 element map. The PI control signal equation was used to set the control signal at these nodes and the nodes of the other elements. This made the FE and PI maps identical.

The FE equation for each element is

$$Q = \Sigma N m \tag{8.6}$$

where N is the shape function for a node and m is the control signal at the node. This gave the control signal throughout the element.

For the center element in a 9 element map with 4 corner nodes, the shape functions for the nodes are

$$N_1 = ¼ \ (1-\alpha) \ (1-\beta) \tag{8.7}$$

$$N_2 = ¼ \ (1+\alpha) \ (1-\beta) \tag{8.8}$$

$$N_3 = ¼ \ (1+\alpha) \ (1+\beta) \tag{8.9}$$

$$N_4 = ¼ \ (1-\alpha) \ (1+\beta) \tag{8.10}$$

The control signal Q throughout the element is

$$Q = N_1Q_1 + N_2Q_2 + N_3Q_3 + N_4Q_4 \tag{8.11}$$

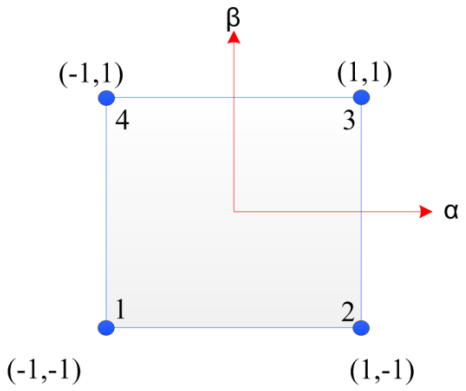


Figure 8-4: Four-node quadrilateral element

Note that at the node corner (N_3) where $\alpha = 1$ and $\beta = 1$

$$Q = 0\,Q_1 + 0\,Q_2 + 1\,Q_3 + 0\,Q_4 = Q_3 \tag{8.12}$$

At the center of the element where $\alpha = 0$ and $\beta = 0$

$$Q = ¼\,(Q_1 + Q_2 + Q_3 + Q_4) \tag{8.13}$$

$$= ¼\,([-Q_{PS} - Q_{IS}] + [+Q_{PS} - Q_{IS}] + [+Q_{PS} + Q_{IS}] + [-Q_{PS} + Q_{IS}]) = 0$$

The MATLAB control functions for PI and FE are attached. In these codes LIMIT is the saturation Q_{PS} and LEVEL is saturation Q_{IS}.

```
% Conventional PI Code

   ONE=KP*ERROR;
   if(ONE>+LIMIT) ONE=+LIMIT; end;
   if(ONE<-LIMIT) ONE=-LIMIT; end;
   TWO=KI*SUM;
   if(TWO>+LEVEL) TWO=+LEVEL; end;
   if(TWO<-LEVEL) TWO=-LEVEL; end;
   CONTROL=ONE+TWO;
```

```
% Finite Element Code

  dd11=-LIMIT-LEVEL; dd12=-LIMIT-LEVEL;
  dd13=+LIMIT-LEVEL; dd14=+LIMIT-LEVEL;
  dd21=-LIMIT-LEVEL; dd22=-LIMIT-LEVEL;
  dd23=+LIMIT-LEVEL; dd24=+LIMIT-LEVEL;
  dd31=-LIMIT+LEVEL; dd32=-LIMIT+LEVEL;
  dd33=+LIMIT+LEVEL; dd34=+LIMIT+LEVEL;
  dd41=-LIMIT+LEVEL; dd42=-LIMIT+LEVEL;
  dd43=+LIMIT+LEVEL; dd44=+LIMIT+LEVEL;
  vv2=-LEVEL/KI; vv1=20*vv2; vv3=+LEVEL/KI; vv4=20*vv3;
  hh2=-LIMIT/KP; hh1=20*hh2; hh3=+LIMIT/KP; hh4=+20*hh3;
  h1=0.0; h2=0.0; v1=0.0; v2=0.0; d1=0.0; d2=0.0; d3=0.0; d4=0.0;
  alpha=0.0; beta=0.0;
  for j=1:3
```

```
for i=1:3
if (j==1) v2=vv2;v1=vv1;end;
if (j==2) v2=vv3;v1=vv2;end;
if (j==3) v2=vv4;v1=vv3;end;
if (i==1) h2=hh2;h1=hh1;end;
if (i==2) h2=hh3;h1=hh2;end;
if (i==3) h2=hh4;h1=hh3;end;
if (error<h2 && error>h1) alpha=2*(error-h1)/(h2-h1)-1;end;
if (sum<v2 && sum>v1) beta=2*(sum-v1)/(v2-v1)-1;end;
if (j==1 && i==1) d1=dd11; d2=dd12; d3=dd21; d4=dd22; end;
if (j==1 && i==2) d1=dd12; d2=dd13; d3=dd22; d4=dd23; end;
if (j==1 && i==3) d1=dd13; d2=dd14; d3=dd23; d4=dd24; end;
if (j==2 && i==1) d1=dd21; d2=dd22; d3=dd31; d4=dd32; end;
if (j==2 && i==2) d1=dd22; d2=dd23; d3=dd32; d4=dd33; end;
if (j==2 && i==3) d1=dd23; d2=dd24; d3=dd33; d4=dd34; end;
if (j==3 && i==1) d1=dd31; d2=dd32; d3=dd41; d4=dd42; end;
if (j==3 && i==2) d1=dd32; d2=dd33; d3=dd42; d4=dd43; end;
if (j==3 && i==3) d1=dd33; d2=dd34; d3=dd43; d4=dd44; end;
if (error<h2 && error>h1 && sum<v2 && sum>v1) break; end;
end
if (error<h2 && error>h1 && sum<v2 && sum>v1) break; end;
end
CONTROL=1/4*(1-alpha)*(1-beta)*d1+1/4*(1+alpha)*(1-beta)*d2 ...
  +1/4*(1-alpha)*(1+beta)*d3 +1/4*(1+alpha)*(1+beta)*d4;
```

The finite element controller had 9 linear elements. The center element had error and integral of error corner node coordinates corresponding to the saturation limits of the PI controller. The PI equations were used to calculate the control signal at each node. So the performance of the FE controller should exactly match that of the PI controller.

The weights of the Neural Network controller were chosen so that its gains at zero error and zero integral of error matched the PI gains and it had the same saturation levels as PI. The only difference between the Neural Network controller and the PI controller is for PI the transition to saturation is abrupt or has a sharp edge where it hits saturation while for NN the transition is gradual because of the S shaped squashing function. So basically the NN controller should be close to the PI controller.

The breakpoints of the Fuzzy Logic controller were chosen to make it approximately match PI. Because of the minimum fuzzy logic rule, each component of the controller does not necessarily contribute to the control signal. So an exact match with PI is not possible. Only 4 rules were used for Fuzzy Logic. More rules would greatly improve its performance.

Chapter 9.

Conclusions, Contributions and Future Work

9.1 Conclusions

Induction machines are widely used in industrial applications. Consequently, much attention has been given to the design and development of induction machine control. Open loop control systems maintain the stator v/f ratio at a predetermined level to establish the desired machine flux. The ratio is satisfied only at low or moderate dynamic requirements. Field orientation technology can provide high performance control of the induction motor by aligning a revolving reference frame with a space vector of selected flux and allowing the induction motor to emulate a separately excited DC machine.

This thesis presents a field oriented control of the induction motor. The method of field oriented control is widely used in controlling electric machines. The theory behind the field oriented control has been described. The application of the theory of the field oriented control on the induction motor has been demonstrated. The model of the induction motor was derived from the "abc" coordinate frame through a conversion into the "d-q" rotating coordinate frame. The demonstration shows that the electromagnetic torque and the magnetizing field can be controlled by the q-axis and the d-axis current, respectively.

A high performance motor drive should possess good command tracking and fast dynamic responses, and these responses should be unaffected by the operating conditions. Uncertainties are usually associated with motor parameter variations, external

load disturbances and nonlinear dynamics of the motor. With the conventional PI speed controller, the limits of the controller gains and the rate at which the gains change have to be appropriately chosen. This makes the PI gains very sensitive to operating conditions, and it is necessary to retune the gains manually at each operating condition to achieve the desired response. Classical adaptive algorithms were applied to automatically adjust the gains of the PI controller in this research in order to make the PI speed controller more robust and responsive to changes in the operating conditions and to avoid manual retuning. Though the use of adaptive controllers has been reported in the literature, this research demonstrates the automated tuning of their gains in order to progress into more advanced controllers, such as artificial neural network (ANN) controllers and the finite element controller map.

An adaptive controller adjusts parameters based on the difference between the desired response of the system and the actual response. Usually an integrated squared difference is calculated and parameters are adjusted to minimize the error. In the basic PID controller, the parameters that can be adjusted are its three gains. Another controller parameter that is usually fixed is the saturation limit, however, there is a limited amount of adjustment that can be done with only three parameters. A nonlinear polynomial type PID can be developed with more parameters that can provide a more complex map, and thus a better fit to the desired response. A piecewise PID can also be used where the gains change when PID inputs cross certain thresholds. When the gains are made a function of disturbance levels, the process is generally referred to as gain scheduling.

In this research ANN is used to develop a complex map that has many degrees of freedom and a more local character. To train a controller based on the integrated squared

difference over a set period of time, it would be best to develop a model of the system, train it and then upload the controller to the actual system. For online training, the most sensitive parameters are selected based on simulation and are adjusted step by step over time.

Most researchers see neural network controllers as fundamentally different from the classical PI controller. However, this study has shown that at the lowest level they are basically the same. This is a new contribution. The network has one neuron for each input. For PI, saturation is abrupt. For NN, it is gradual. This is really the only difference between conventional PI and lowest level NN control.

The Finite Element Controller Map (FECM) has also been included in this research, which is considered an original and main contribution. This type of controller has a complex map like the ANN, but has more degrees of freedom and a more local character than ANN which is ideal during real-time implementation when a large variation occurs in the system. This work is considered to be new. There is no mention of FECM in the available literature.

The quality of system control depends on the complexity of the controller map. This thesis proposes to use finite element shape functions to construct a PI map. This is unique and can be found nowhere else in the literature. The output of the map at finite element nodes can be adjusted, as can the locations of the finite element nodes in the PI plane. This makes it possible to construct maps with localized features that have very few degrees of freedom.

Initially, work was done with conventional PI controllers to gain experience making a controller self-tuning. Then neural network squashing functions were used to create a more complex PI map, which then was made self-tuning. Finally, a finite element controller map was developed and made self-tuning. In each case, the map was trained offline using a model of the system and made self-tuning by adjusting only the most sensitive parameters. The maps were tested in a motor speed control system. To minimize calculations, the work completed to date has used very simple linear shape functions in the finite element map.

The thesis developed several intelligent controllers for induction motors, these controllers were shown to mimic the conventional PI controller. This shows that the coding for the intelligent controllers was developed properly. Each controller was made self-tuning. The performance of each controller was checked experimentally.

9.2 Contributions

1- Developed a simple fuzzy logic controller for an induction motor. Showed that the break points in the controller determine the effective gains and saturations of the controller.

2- Developed a simple neural network controller for an induction motor. In this case, the weights in the controller determine the effective gains and saturations of the controller.

3- Developed a novel finite element controller for an induction motor. In this situation, the information at the nodes in the controller determines the effective gains and saturations of the controller.

4- Simulated the controllers using Matlab SIMULINK. Parameters were chosen to make them mimic PI. Root mean square error (RMSE) was Zero for FECM and Small for NN and Large for FL. These results were expected.

5- Developed a simple self-tuning scheme for each controller based on a few parameters that allowed each controller to be used and easy implemented.

6- Tested each controller in a lab induction motor.

Other researchers have used Matlab tool box for Fuzzy Logic and Artificial Neural Networks controllers. No other researchers have used finite element controllers which can have local character and could give robust control. The simple structure of the intelligent controllers made them easier to make self-tuning.

9.3 Future Work

1- Make controllers adaptive. Presently, only a few parameters are varied to make the controllers self-tuning, in future work, more parameters will be allowed to be varied online. This will make the controller adaptive and more robust.

2- Develop faster self-tuning algorithm. Presently, a simple a grid search technique is used to make the controller self-tuning, in future work, a faster technique such as Newton Raphson will be used to obtain an accurate and faster response.

3- Develop better optimum performance. Currently, the conventional PI controller with optimum gain is used as the optimum performance, in future work, a more realistic optimum performance will be developed.

4- Make controller maps more complex. Presently, the intelligent controllers are made to mimic the conventional PI controller; this is a very simple, almost planer map. In future work, parameters will be adjusted to allow for more complex maps with local features such as jumps up or down.

5- Explore potential of asymmetric maps. Currently, the intelligent controllers are symmetric in the sense that for every positive control signal, there is a corresponding a negative control signals having the same magnitude. In future work, the potential of maps that have different positive and negative magnitude will be explored.

6- Apply controller to other motors, currently the controller was applied to the induction motor. In future work, they would be applied to motors like IPM.

References

[1] T. Kume and T. Iwakane, “High-Performance vector controlled AC motor drives, Application and New Technologies,” *IEEE Transactions on Industry Applications,* Vol. IA-23, no. 5, pp. 872-880, September/October 1987.

[2] G. Kaufman, L. Garces, and G. Gallagher, “High performance servo drives for machine tool applications using ac motors”, proceedings IEEE- Industry Applications Society Annual Meeting Conference Record, San Francisco, USA 1982, pp. 604-609, 4-7th October, 1982.

[3] B. K. Bose, “Power electronics and motor drives recent progress and perspective,” IEEE Transactions on Industrial Electronics, Vol. 56, no. 2, pp. 581–588, February 2009.

[4] B. K. Bose, “Neural Network Applications in Power Electronics and Motor Drives-An Introduction and Perspective,” IEEE Transactions on Industrial Electronics, Vol. 54, no.1, pp 14-33, February 2007.

[5] T.S. Radwan, M.N Uddin, M.A. Rahman, "A new and simple structure of fuzzy logic based indirect field oriented control of induction motor drives," Proceedings of IEEE Power Electronics Specialists Conference, 35th Annual meeting conference, Aachen, Germany, Vol. 5, no. 20, pp.3290-3294, 25 June 2004.

[6] M. A. Khan, M. N. Uddin, and M. A. Rahman, “Real-time performance investigation of an intelligent controller based speed control of induction motor drives Proceedings of IEEE International Electric Machines & Drives Conference, Niagara Falls, Canada, pp. 164–169, August 2011.

[7] M. A. Rahman, R. M. Milasi, C. Lucas, B. N. Araabi, and T. S. Radwan, "Implementation of emotional controller for interior permanent-magnet synchronous

motor drive," IEEE Transactions on Industry Applications, Vol. 44, no. 5, pp. 1466-1476, September/October 2008.

[8] V.M.Panchade, R.H.Chile, B.M.Patre, “A survey on sliding mode control strategies for induction motors,” ELSEVIER Annual Reviews in Control Vol. 37, No. 2, , pp. 289-307, India, December 2013.

[9] R. Lorenz, “Turning of Field-Oriented Induction Motor Controllers for High-Performance Applications,” IEEE Transactions on Industry Applications, Vol. IA-22, no. 2, pp. 293-297, March/April 1986.

[10] Fizatul Aini Patakor, Marizan Sulaiman, Zulkifilie Ibrahim “Comparison performance of induction motor using SVPWM and hysteresis current controller” Article in Journal of Theoretical and Applied Information Technology, Vol. 30, no.1, August 2011.

[11] M.H.N Talib, Z. Ibrahim, N. Abd. Rahim, A.S.A. Hasim, “Comparison Analysis of Indirect FOC Induction Motor Drive using PI, Anti-Windup and Pre Filter Schemes” Article in International Journal of Power Electronics and Drive Systems· Vol. 5, no. 2, pp. 219-229, October 2014.

[12] R. Krishnan and A. S. Bharadwaj, “A Review of parameter sensitivity and adaptation in indirect vector controlled induction motor drive systems,” IEEE Transactions on Power Electronics. Vol.6, no. 4, pp.695-703, October 1991.

[13] R. D. Lorenz and D. B. Lawson, “A Simplified approach to continuous on-line tuning of field oriented Induction motor Drives,” IEEE Transactions on Industry Applications, Vol. 26, no. 3, pp. 420-424, May/June 1990.

[14] T. Rowan, R. Kerkman and D. Leggate, “A Simple on-line adaptation for indirect field orientation of an induction machine,” IEEE Transactions on Industry Applications, Vol. 27, no. 4, pp.720-727, July/August 1991.

[15] R.Seliger, B. Köppen, P.M. Frank, M.A. El-Sharkawi, “Self-Tuning Adaptive Control for Field-Oriented Controlled Induction Motors - a simulation Study”, Elsevier IFAC Proceedings, Vol. 25, no. 15, PP 505-510, July 1992.

[16] A. B. Plunkett and D. L. Plette, “Inverter-induction motor drive for transit cars”, IEEE Transactions on Industry Applications, Vol. IA-13, no. 1, pp. 26-37, January 1977.

[17] A. B. Plunkett, “A Current Controlled PWM Transistor Inverter Drive”, IEEE Industry Applications Society, Proceeding of Annual Meeting Conference Record, Cleveland, Ohio, USA, pp. 785-792, October 1979.

[18] E. P. Cornell and T. A. Lipo, “Modelling and design of controlled current induction motor drive system”, IEEE Transactions on Industry Applications, Vol. 13, no. 4, pp. 321-330, July 1977.

[19] H.-B. Shin and J.-G. Park, “Anti-windup PID controller with integral state predictor for variable-speed motor drives,” IEEE Transactions on Industrial Electronics , Vol. 59, no. 3, pp. 1509–1516, March 2012.

[20] Y. Agrebi Zorgani, Y. Koubaa, M. Boussak, “Simultaneous estimation of speed and rotor resistance in sensorless ISFOC induction motor drive based on MRAS scheme”, proceedings of 19th International Conference on Electrical Machines, ICEM 2010, Rome, Italy, 6-8 September 2010.

[21] S. A. Mir, D. S. Zinger, M. E. Elbuluk, “Fuzzy Controller for Inverter Fed Induction Machines”, in *IEEE IAS Annu. Rec.*, 1992, pp. 464-471.

[22] L. A. Zadeh, “Fuzzy sets”, ELSEVIER Information and Control, Vol. 8, no. 3, pp. 338-353, June 1965.

[23] L. A. Zadeh, “Outline of a newa to the analysis of complex system and decision processes”, IEEE Transactions on Systems, Man and Cybernetics, Vol. SMC-3, pp. 28-44, 1973.

[24] B. K. Bose, “Expert systems, fuzzy logic, and neural network applications in power electronics and motion control”, Proceedings IEEE, Vol. 82, pp. 1303- 1323, August 1994.

[25] P. Vas, Artificial-Intelligence-Based Electric Machines and Drives. New York: Oxford University Press, 1999.

[26] J. O. P. Pinto, B. K. Bose, L. E. Borges, M. P. Kazmierkovski, “A Neural network-based space-vector PWM controller for voltage-fed inverter induction motor drive”, proceedings of IEEE Industry Applications Society Annual Meeting, Rome, Italy, pp. 1605-1612, July 2000.

[27] Y. Tang and L. Xu, “Fuzzy logic application for intelligent control of a variable speed drive”, IEEE Transactions on Energy Conversion, Vol. 9, no. 4, pp. 679- 685, December 1994.

[28] Li Zhen and L. Xu, “Fuzzy logic enhanced speed control of an indirect field-oriented induction motor drive”, IEEE Transactions on control systems technology, Vol. 8, no. 2, pp. 270-278, March 2000.

[29] J. Fonseca, I.E. Afonso, J. S. Martins, and C. Couto, “Fuzzy logic speed control of an induction motor”, Microprocessor and Microsystem. Vol. 22, pp. 523-534, 1999.

[30] M. A. Rahman and M. A. Hoque, “On-Line Self-Tuning ANN Based Speed Control o f a PM DC Motor”, IEEE/ASME Transactions on Mechatronics, vol. 2, no. 3, , pp. 169-178, September 1997.

[31] Krishnan, R. ‘Electric Motor Drives, Modeling Analysis and Control’, Prentice Hall, Englewood Cliffs (2001).

[32] B. K. Bose, “Modern Power Electronics and Motor Drives’, Advances and Trends

[33] B. K. Bose, “Modern Power Electronics and AC drives”, Pearson Prentice Hall International, London (2002).

[34] B. K. Bose, (Edited), ‘Adjustable speed A.C. drive systems’, IEEE Press, New York, 1981.

[35] Seung-Ki Sul, ‘Control of Electric Machine Drive Systems’, IEEE Press, Published by John Wiley & Sons, Inc., Hoboken, New Jersey.

[36] A. Saghafinia, H. Ping, M. Nasir Uddin, Khalaf Salloum Gaeid, ‘Adaptive Fuzzy Sliding-Mode Control Into Chattering-Free IM Drive’, IEEE Transactions on Industry Applications, Vol. 51, no. 1, January, February 2015.

[37] M. Nasir Uddin, Zhi Rui Huang, A. B. M. Siddique Hossain, ‘Development and Implementation of a Simplified Self-Tuned Neuro–Fuzzy-Based IM Drive’, IEEE Transactions on Industry Applications, Vol. 50, no. 1, January/February 2014.

[38] M. N. Uddin, T. S. Radwan, and M. A. Rahman, “Performances of fuzzy-logic-based indirect vector control for induction motor drive,” IEEE Transactions Industry Applications, Vol. 38, no. 5, pp. 1219–1225, September/October 2002.

[39] M. Nasir Uddin, T. S. Radwan, M. A. Rahman, 'Performance Analysis of a Four Switch 3-Phase Inverter Fed IM Drives' Large Engineering Systems Conference on Power Engineering', Pages: 36 – 40, 2004.

[40] Bin Wu, 'High-power converters and ac drives' John Wiley Wiley.

[41] W. Leonhard, 'Control of Electrical Drives', Springer-Verlag, 1985.

[42] Shaahin Filizadeh, 'Electric Machines and Drives: Principles, Control, Modeling, and Simulation' CRC Press; 1 edition (20 Feburary 2013).

[43] Boldea, I., Nasar, S.A.: Electric Drive. CRC Press, Boca Raton (1999).

[44] G. Slemon, 'Electric Machines and Drives', Addison-Wesley, 1992'

[45] Mukhtar Ahmad, 'High Performance AC Drives, Modelling Analysis and Control', London : Springer.

[46] A. Chakraborty, "Advancements in power electronics and drives in interface with growing renewable energy resources," ELSEVIER Renewable and Sustainable Energy Reviews, Vol. 15, no. 4, pp. 1816-1827, May 2011.

[47] Saidur Rahman, S. Mekhilef, M.B Ali, A. Safari, H.A. Mohammed, " Applications of variable speed drive (VSD) in electrical motors energy savings," ELSEVIER Renewable and Sustainable Energy Reviews, Vol. 16, no. 1, pp. 543-550, January 2012.

[48] Saidur Rahman, "A review on electrical motors energy use and energy savings," ELSEVIER Renewable and Sustainable Energy Reviews, Vol. 14, no. 3, pp. 877-898, April 2010.

[49] I. Al-Bahadly, "Energy saving with variable speed drives in industry applications," Proceedings of the WSEAS Int. Conference on Circuits, Systems, Signal and Telecommunications, Gold Coast, Australia, pp. 53-58, 17-19 January 2007.

[50] Mohammad Jannati, Sajad Abdollahzadeh Anbaran, Seyed Hesam Asgari, Wee Yen Goh, Ali Monadi, Mohd Junaidi Abdul Aziz, Nik Rumzi Nik Idris, 'A Review on Variable Speed Control techniques for efficient control of Single-Phase Induction Motors: Evolution, classification, comparison', ELSEVIER Renewable and Sustainable Energy Reviews, Vol. 75, pp. 1306–1319, Malaysia, August 2017.

[51] M.A. Hannana, Jamal A. Alib, Azah Mohamedc, Aini Hussain,"Optimization techniques to enhance the performance of induction motor drives: A review," ELSEVIER Renewable and Sustainable Energy Reviews, Vol. 81, part 2, pp. 1611-1626, Malaysia, January 2018.

[52] Ibrahim M. Alsofyani, N.R.N. Idris, "A review on sensorless techniques for sustainable reliablity and efficient variable frequency drives of induction motors," ELSEVIER Renewable and Sustainable Energy Reviews, Vol. 24, pp. 111–121, Malaysia, August 2013.

[53] Rashid MH. Power electronics handbook. Canada: Academic press; 2001.

[54] Dino Zorbas, "Electric Machines Principles, Applications and Control Schematics", second edition.

[55] Charles I. Hubert, 'Electric Machines: Theory, Operation, Applications, Adjustment, and Control', February 1, 1991.

[56] A. Mu˜noz-Garcia, T. A. Lipo, and D. W. Novotny, "A new induction motor V/f control method capable of high-performance regulation at low speeds", IEEE Transactions on Industry Applications, Vol. 34, no. 4, pp. 813–821, July/August 1998.

[57] A. Smith, S. Gadoue, M. Armstrong, and J. Finch, “Improved method for the scalar control of induction motor drives,” IET Electric Power Applications, Vol. 7, No. 6, pp. 487-498, 22 July 2013.

[58] Suetake, M., da Silva, I.N., Goedtel, A.: ‘Embedded DSP-based compact fuzzy system and its application for induction-motor V/f speed control’, IEEE Transactions on Industrial Electronics , Vol. 58, no. 3, pp. 750–760. March 2011.

[59] Paul C. Krause ; Oleg Wasynczuk ; Scott D. Sudhoff, ‘Analysis of Electric Machinery and Drive Systems’, Second Edition, Copyright Year: 2002, WILEY-IEEE PRESS.

[60] Tole Sutikno, Nik Rumzi Nik Idris, Auzani Jidin, “A Review of direct torque control of induction motors for sustainable reliability and energy efficient drives,” Elsevier Renewable and Sustainable Energy Reviews, Vol. 32, pp. 548–558, Malaysia, April 2014.

[61] C.M.F.S. Reza, Md. Didarul Islam, Saad Mekhilef, “A review of reliable and energy efficient direct torque controlled induction motor drives”, Elsevier Renewable and Sustainable Energy Reviews, Vol. 37, pp. 919–932, 2014.

[62] Eun-Chul Shin, Tae-Sik Park, Won-Hyun Oh, Ji-Yoon Yoo ‘A design method of PI controller for an induction motor with parameter variation”, Industrial Electronics Society, IECON '03. The 29th Annual Conference of the IEEE, 2-6 Nov. 2003, Roanoke, VA, USA.

[63] P. Vas. Vector Control of AC Machines. Oxford University Press, 1990.

[64] C. Ogbuka, C. Nwosu, M. Agu, ‘A New high speed induction motor drive based on field orientation and hysteresis current comparison’, Journal of Electrical

Engineering, Vol. 67, no. 5, pp. 334-342, University of Nigeria, Nsukka, Nigeria, 02 November 2016.

[65] S.K. Panda, J.M.S. Lim, P.K. Bash, K.S. Lock, "Gain scheduled PI speed controller for PMSM drive," proceedings of IEEE Industrial Electronics Society International Conference IECON 97, pp. 925–930, New Orleans, USA , November 1997.

[66] B. Singh, A.H.N. Reddy, S.S. Murthy, Gain scheduling control of permanent magnet brushless DC motor, IE (I) J.-EL 84 pp. 52–62, 2003.

[67] W.D. Chang, R.C. Hwang, J.G. Hsieh, A Self-tuning PID control for a class of nonlinear systems based on the Lyapunov approach,"ELSEVIER Journal of Process Control, Vol. 12, no. 2, pp. 233–242. Februray 2002.

[68] O. Lequin, M. Gevers, M. Mossberg, E. Bosmans, L. Triest, "Iterative feedback tuning of PID parameters: comparison with classical tuning rules," Control Engineering Practice, Vol. 11, No. 9, pp. 1023–1033. September 2003.

[69] K.J. Astrom, T. Hagglund, "Revisiting the Ziegler–Nichols step response method for PID control," Journal of Process Control, Vol. 14, no. 6, pp. 635–650. September 2004.

[70] B. Kristiansson, B. Lennartson, "Evaluation and simple tuning of PID controllers with high-frequency robustness," Journal of Process Control, Vol. 16, no. 2, pp. 91–102. February 2006.

[71] D. G. Holmes, B. P. McGrath, S. G. Parker, "A Comparative evaluation of high performance current regulation strategies for vector controlled induction motor drives," ISIE, 2010 IEEE International Symposium pp. 3707-3714. 2010.

[72] Mohamed S. Zaky and Ehab M. Ismaeil, " Gain Scheduling Adaptive PI Controller for a Field Oriented Control of Hybrid Stepper Motor Drives," Proceedings of 13th International Middle-East Power Systems Conference (MEPCON),. Cairo University, Egypt, pp. 85-91, 19-21 December, 2010.

[73] Mohamed S. Zaky, 'A self-tuning PI controller for the speed control of electrical motor drives', Electric Power Systems Research 119 (2015) 293–303.

[74] Model Reference Adaptive System using Rotor Flux and Back Emf Techniques for Speed Estimation of an Induction Motor operated in Vector Control Mode: A Comparative Study', proceedings of IEEE Uttar Pradesh Section International Conference on Electrical, Computer and Electronics Engineering (UPCON), Varanasi, India, December 2016.

[75] S. Maiti, C. Chakraborty, Y. Hori, and C.M Ta, "Model reference adaptive controller-based rotor resistance and speed estimation techniques for vector controlled induction motor drive utilizing reactive power", IEEE Transactions on Industrial Electronics, Vol. 55, no. 2, pp. 594–601, February 2008.

[76] R. Kumar, S. Das, P. Syam and A. K. Chattopadhyay, "Review on model reference adaptive system for sensorless vector control of induction motor drives," Institution of Engineering and Technology, IET Electric Power Applications, Vol. 9, no. 7, pp. 496-511, July 2015.

[77] M. M. Khater, M. S. Zaky, H. Yasin, S. S. Shokralla, and A. El-Sabbe, "A Comparative study of sliding mode and model reference adaptive speed observers for induction motor drives," Proceedings of IEEE-Eleventh International Middle-East Power

Systems Conference (IEEE-MEPCON'2006), El-Minia, Egypt, Vol. 2, pp. 434-440, December 19-21, 2006.

[78] Vadim I. Utkin, "Sliding mode control design principles and applications to electric drives," IEEE Transactions on Industrial Electronics, Vol 40, pp. 23–36, Februry1993.

[79] Asif Sabanovic, "Variable Structure Systems With Sliding Modes in Motion Control—A Survey,", IEEE Transactions on Industrial informatics, Vol. 7, no. 2, pp. 212–223, May 2011.

[80] K. Jamoussi, M. Ouali, H. Charradi, "A Sliding Mode Speed Control of an Induction Motor," American Journal of Applied Sciences, Vol. 4, No. 12, pp 987-994, Science Publications 2007.

[81] Hoon Lee, Vadim I. Utkin, "Chattering suppression methods in sliding mode control systems," ELSEVIER Annual Reviews in Control Vol. 31, pp. 179-188, October 2007.

[82] Wilfrid Perruquetti, Jean-Pierre Barbot, "Sliding Mode Control in Engineering", CRC Press, January 2002.

[83] F. J. Lin, P. H. Shen, and S. P. Hsu, "Adaptive backstepping sliding mode control for linear induction motor drive," Proceedings Institution of Electrical Engineering, Electric Power Applications, Vol. 149, no. 3, pp. 184–194, August 2002.

[84] Ibtissem Bakhti, Souad Chaouch and Abdesselam Maakouf, "High performance backstepping control of induction motor with adaptive sliding mode observer", Archives of Control Sciences Vol. 21, no. 3, pp. 331–344, January 2011.

[85] F. Mehazzem, A. Reama, Y. Hamam, and H. Benalla, “Real time implementation of backstepping controller in indirect field oriented control of induction motor drive”, Proceedings of IEEE second International Conference on Power Engineering, Energy and Electrical Drives, Lisbon, Portugal, 18-20 March 2009.

[86] B. Sahu, K. B. Mohanty, and S. Pati, "A Comparative study on fuzzy and PI speed controllers for field-oriented induction motor drive," IEEE in Industrial Electronics, Control & Robotics (IECR), 2010 International Conference, pp. 191-196. Wroclaw, Poland, September 2010.

[87] M. N. Uddin, T. S. Radwan, and M. A. Rahman, "Performances of fuzzy-logic-based indirect vector control for induction motor drive,", IEEE Transactions on Industry Applications, Vol. 38, no. 5, pp. 1219-1225, September/October 2002.

[88] R.Arulmozhiyal and D. K.Baskaran, “Speed Control of Induction Motor using Fuzzy PI and Optimized using GA”, International Journal of Recent Trends in Engineering, Vol 2, no. 5, November 2009.

[89] M.A. Hannana, Jamal A. Ali, Azah Mohamed, Aini Hussain, “Optimization techniques to enhance the performance of induction motor drives: A review”, Renewable and Sustainable Energy Reviews, Volume 81, Part 2, pp. 1611-1626, January 2018.

[90] C. B. Butt, M. Ashraful Hoque, and M. A. Rahman, “Simplified fuzzy-logic-based MTPA speed control of IPMSM drive,” IEEE Transactions on Industry Applications, vol. 40, no. 6, pp. 1529-1535, November./December 2004.

[91] V. Chitra, and R. S. Prabhakar, “Induction motor speed control using fuzzy logic controller”, World Academy of Science, Engineering and Technology, 2006.

[92] Rajesh Kumar, R.A. Gupta and S.V. Bhangale, “Microprocessor/digital control and artificial intelligent vector control techniques for induction motor drive”, IETECH Journal of Electrical Analysis, 2008.

[93] C.B. Butt and M. A. Rahman, "Intelligent Speed Control of Interior Permanent Magnet Motor Drives Using a Single Untrained Artificial Neuron," IEEE Transactions on Industry Applications, vol. 49, no. 4, pp. 1836-1843, July-August 2013.

[94] Bose, B.K.: Neural Network Applications in Power Electronics and Motor Drives— An Introduction and Perspective. IEEE Transaction on Power Electronics 54, 14–33 (2007).

[95] J. O. P. Pinto, B. K. Bose, L. E. B. daSilva, and M. P. Kazmierkowski, “A neural network based space vector PWM controller for voltage-fed inverter induction motor drive,” IEEE Trans. Ind. Appl., vol. 36, no. 6, pp. 1628–1636, November/December 2000.

[96] S. Mondal, J. O. P. Pinto, and B. K. Bose, “A neural network based space vector PWM controller for a three-level voltage-fed inverter induction motor drive,” IEEE Trans. Ind. Appl., vol. 38, no. 3, pp. 660–669, May/June 2002.

[97] J. Zhao and B. K. Bose, “Neural network based waveform processing and delayless filtering in power electronics and ac drives,” IEEE Transactions on Industrial Electronics, vol. 51, no. 5, pp. 981–991, October 2004.

[98] P. Vas, “Artificial-Intelligence-Based Electrical Machines and Drives Application of Fuzzy, Neural, Fuzzy-Neural and Genetic Algorithm Based Techniques”. New York: Oxford University Press, 1999.

[99] W. S. Oh, Y. T. Kim, C. S. Kim, T. S. Kwon and H. J. Kim, "Speed Control of Induction Motor Using Genetic Algorithm Based Fuzzy Controller," Annual Conference on Industrial Electronics Society, pp. 625-629, 1999.

[100] M'hamed Chebre, Abdelkader Meroufel, Yessema Bendaha, "Speed Control of Induction Motor Using Genetic Algorithm-based PI Controller", Acta. Polytechnica Hungarica, Vol. 8, no. 6, 2011.

[101] H. Razik, C. Defranoux, A. Rezzoug, Identification of Induction Motor using a Genetic Algorithm and a Quasi-Newton Algorithm, CIEP, pp. 65–70, Acapulco, Mexico, October 15–19, 2000.

[102] C. M. Liaw and F.J. Lin, "A discrete adaptive induction position servo system," IEEE Transactions on Energy Conversion, Vol. 8, no. 3, pp. 350-356, September 1993.

[103] N.S.Gehlot and P.J.Alsina, "A discrete model of induction motors for real time control applications," IEEE Transactions on Industrial Electronics, Vol. 40, no. 3, pp. 317-325, June 1993.

[104] D.C. Lee, S.K. Sul and M.H. Park "High performance current regulator for a field-oriented controlled induction motor drive," IEEE Transactions on Industry Applications, Vol. 30, no. 5, pp. 1247-1257, September/October 1994.

[105] W. Leonhard, "Controlled AC drives, a successful transfer from ideas to industrial practice", Control Engineering Practice, Vol. 4, no. 7, pp. 897-908, 1996.

[106] X . Nian ,J. Wang , W. Gui , J. Huang , Z. Huang ,"A Constant Gain Adaptive Observer for Speed and Resistances Identification", Conference Record of the 2006 IEEE Industry Applications Conference Forty-First IAS Annual Meeting, pp.712-718 Tampa, FL, USA, 8-12 October 2006.

[107] Anno Yoo, Sang-Heon Han, Young Ik Son, Young-Doo Yoon, and Chanook Hong, “Gain design of an adaptive full-order observer using a pole placement technique for speed sensorless induction motor drives” Journal of Power Electronics, Vol. 16, no. 4, pp. 1346-1354, July 2016.

[108] S. Bolognani and M. Zigliotto, "Hardware and software effective configurations for multi-input fuzzy logic controllers," IEEE Transactions on Fuzzy Systems, Vol. 6, no. 1, pp. 173-179, February 1998.

[109] S. N. Mat Isa, Z. Ibrahim, J. M. Lazi, M. H. N. Talib, N. M. Yaakop, and A. S. Abu Hasim, "dSPACE DSP based implementation of simplified fuzzy logic speed controller for vector controlled PMSM drives," IEEE International Conference on Power and Energy (PECon), pp. 898-903, Kota Kinabalu, Malaysia, 2-5 December. 2012,.

[110] M. H. N. Talib, Z. Ibrahim, N. A. Rahim, A. S. Abu Hasim and H. Zainuddin, "Performance improvement of induction motor drive using simplified FLC method," 2014 16th International Power Electronics and Motion Control Conference and Exposition (PEMC), pp. 707-712. Antalya, Turkey, 21-24 September 2014.

[111] M. N. Uddin, T. S. Radwan and M. A. Rahman, "Performances of fuzzy-logic-based indirect vector control for induction motor drive," IEEE Transactions on Industry Applications, Vol. 38, no. 5, pp. 1219-1225, September/October 2002.

[112] Z. Ibrahim and E. Levi, "A detailed comparative analysis of fuzzy logic and PI speed control in high performance drives," Seventh International Conference on Power Electronics and Variable Speed Drives (IEE Conf. Publ. No. 456), pp. 638-643.London, UK, 21-23 September 1998.

[113] Z. Ibrahim and E. Levi, "A comparative analysis of fuzzy logic and PI speed control in high-performance AC drives using experimental approach," IEEE Transactions on Industry Applications, Vol. 38, no. 5, pp. 1210-1218, November 2002.

[114] R. J. Wai, "Fuzzy Sliding-Mode Control Using Adaptive Tuning Technique," in IEEE Transactions on Industrial Electronics, Vol. 54, no. 1, pp. 586-594, February 2007.

[115] K. Bose and N. R. P. a. K. Rajashekara, "A Neuro-Fuzzy Based On-Line Efficiency Optimization Control of a Stator Flux-Oriented Direct Vector Controlled Induction Motor Drive," IEEE Trans. on Industrial Electronics, Vol. 44, no. 2, pp. 270-273, April 1997.

[116] T. Banerjee, S. Choudhuri and K. Das Sharma, "Speed tracking scheme for FOC based induction motor by fuzzy controller," Proceedings of IEEE 2014 International Conference on Control, Instrumentation, Energy and Communication (CIEC), Calcutta, India, pp. 71-75, January/February 2014.

[117] C. B. Butt, M. A. Hoque and M. A. Rahman, "Simplified fuzzy-logic-based MTPA speed control of IPMSM drive," IEEE Transactions on Industry Applications, Vol. 40, no. 6, pp. 1529-1535, November/December 2004.

[118] E. Cerruto, A. Consoli, A. Raciti and A. Testa, "Fuzzy adaptive vector control of induction motor drives," IEEE Transactions on Power Electronics, Vol. 12, no. 6, pp. 1028-1040, November 1997.

[119] B. Heber, X. Longya, and Y. Tang, "Fuzzy logic enhanced speed control of an indirect field-oriented induction machine drive," IEEE Transactions on Power Electronics, Vol. 12, no. 5, pp. 772-778, September 1997.

[120] M. Masiala, B. Vafakhah, J. Salmon and A. M. Knight, "Fuzzy Self-Tuning Speed Control of an Indirect Field-Oriented Control Induction Motor Drive," IEEE Transactions on Industry Applications, Vol. 44, no. 6, pp. 1732-1740, November/December 2008.

[121] L. Han-Xiong and H. B. Gatland, "A new methodology for designing a fuzzy logic controller," IEEE Transactions on Systems, Man and Cybernetics, Vol. 25, no. 3 pp. 505-512, March 1995.

[122] Mike Hinchey, "Experimental Methods Engineering ," EN 9211 Notes, Mechanical Engineering, Memorial University of Newfoundland St. John's, NL Canada, 2012.

[123] Amit Mishra, Zaheeruddin, "Design of Speed Controller for Squirrel-cage Induction Motor using Fuzzy logic based Techniques, "International Journal of Computer Applications, Vol. 58, no. 22, November 2012.

[124] M. Masiala, B. Vafakhah, A. Knight and J. Salmon, "Performances of PI and Fuzzy-Logic Speed Control of Field-Oriented Induction Machine Drives," IEEE Canadian Conference on Electrical and Computer Engineering, pp. 397-400, Vancouver, BC, Canada, 22-26 April 2007.

[125] W. Shun-Chung and L. Yi-Hua, "A Modified PI-Like Fuzzy Logic Controller for Switched Reluctance Motor Drives," IEEE Transactions on Industrial Electronics, Vol. 58, no. 5, pp. 1812-1825, May 2011.

[126] S. Chakraverty, Susmita Mall, “Artificial Neural Networks for Engineers and Scientists, Solving Ordinary Differential Equations”, CRC Press, 2017.

[127] Maurizio Cirrincione, Marcello Pucci, Gianpaolo Vitale, “Power Converters and AC Electrical Drives with Linear Neural Networks”, CRC Press, 2012.

[128] B. K. Bose, “Neural Network Applications in Power Electronics and Motor Drives-An Introduction and Perspective”, IEEE Transactions on Industrial Electronics , Vol. 54, no. 1, pp 14-33, 05 February 2007.

[129] B. Yegnanarayana, “Artificial Neural Networks”, Prentice Hall of India, New Delhi, 2006.

[130] M. A. Rahman and M.A. Hoque, "On-Line Adaptive Artificial Neural Network Based Vector Control of Permanent Magnet Synchronous Motors," IEEE Transactions on Energy Conversion, Vol. 13, no. 4, pp. 311-318, January/February 1998.

[131] A. Ba-Razzouk, A. Cheriti, G. Olivier, P. Sicard, "Field-oriented control of induction motors using neural-network decouplers ,", IEEE Transactions on Power Electronics, Vol. 12, no. 4, pp. 752-763, July 1997.

[132] Marian P. Kazmierkowski, Ramu Krishnan, Frede Blaabjerg, “Control in Power Electronics: Selected Problems” Academic Press, 2002.

[133] B. Karanayil, M. F. Rahman and C. Grantham, "Online stator and rotor resistance estimation scheme using artificial neural networks for vector controlled speed sensorless induction motor drive", IEEE Transaction on Industrial Electronics, Vol. 54, no. 1, pp. 167-176, 05 February 2007.

[134] C.B. Butt and M. A. Rahman, "Intelligent Speed Control of Interior Permanent Magnet Motor Drives Using a Single Untrained Artificial Neuron," IEEE Transactions on Industry Applications, Vol. 49, no. 4, pp. 1836-1843, July/August 2013.

[135] Apoorva Saxena, Sayak Dutta, A. Chitra,"Artificial Neural Network Controller for Vector Controlled Induction Motor Drive", International Journal of Computer Applications, Vol. 46, no. 14, pp. 35-40, May 2012.

[136] M. Hinchey, "ENGI 7934 Finite Element Analysis course notes", Mechanical Engineering, Memorial University of Newfoundland, St. John's, Canada.

[137] Carl T. F. Ross," Advanced Applied Finite Element Methods," 1st Edition, eBook ISBN: 9780857099754, Hardcover ISBN: 9781898563518, Imprint: Woodhead Publishing, Published Date: 1st September 1998.

[138] T. L. Eldho, Y. M. Desai, "Finite Element Method with Applications in Engineering," Publisher: Pearson India, Release Date: January 2011, ISBN: 9788131724644.

[139] Gouri Dhatt, Emmanuel Lefrancois, Gilbert Touzot, "Finite Element Method," ISBN: 978-1-84821-368-5, 624 pages, Nov. 2012, Wiley-ISTE.

[140] Prof. Dr. B. N. Rao, "Finite Element Analysis," Department of Civil Engineering, Lecture – 36.

[141] T. L. Eldho, Y. M. Desai, "Finite Element Method with Applications in Engineering," Publisher: Pearson India, Release Date: January 2011, ISBN: 9788131724644.

[142] Faa-Jeng Lin, Wen-Der Chou, "An induction motor servo drive using sliding-mode controller with genetic algorithm", Elsevier, Electric Power Systems Research 64 (2003) 93-108.

[143] R. Palm, "Sliding mode fuzzy control", 1992, pp. 519-526.

[144] G. C. D. Sousa and B. K. Bose, "Fuzzy logic applications to power electronics and drives-an overview," in Industrial Electronics, Control, and Instrumentation, 1995., Proceedings of the 1995 IEEE IECON 21st International Conference on, 1995, pp. 57-62 vol.1.

[145] N. Islam, M. Haider, M. B. Uddin, and Ieee, "Fuzzy logic enhanced speed control system of a VSI-fed three phase induction motor," in 2nd International Conference on Electrical and Electronics Engineering (ICEEE 2005), Mexico City, MEXICO, 2005, pp. 296-301.

[146] S. Bolognani and M. Zigliotto, "Hardware and software effective configurations for multi-input fuzzy logic controllers," Fuzzy Systems, IEEE Transactions on, vol. 6, pp.173-179, 1998.

[147] M. Singh and A. Chandra, "Application of adaptive network-based fuzzy inference system for sensorless control of PMSG-based wind turbine with nonlinear-load compensation capabilities," Power Electronics, IEEE Transactions on, vol. 26, pp. 165-175, 2011.

[148] S. Y. Wang, C. L. Tseng, and C. J. Chiu, "Design of adaptive TSK-fuzzy observer for vector control induction motor drives," 2011, pp. 5220-5223.

[149] L. Li and Z. Xizheng, "Regular paper Indirect Adaptive Fuzzy Sliding-mode Control for Induction Motor Drive," J. Electrical Systems, vol. 7, pp. 412-422, 2011.

[150] A. Ibaliden and P. Goureau, “Fuzzy robust speed control of induction motor”, in Proc. IC EM ’96, Pt. Ill, Vigo, Spain, 1996, pp. 168-173.

[151] M. F. Lai, M. Nakano, G. C. Hsieh, “Application of fuzzy logic in the phase-locked loop speed control of induction motor drive”, IEEE Ttrans. on Ind. Electr., vol. 43, no. 6, pp. 630-639, 1996.

[152] B. Heber, L. Xu, and Y. Tang, “Fuzzy logic enhanced speed control of an indirect field-oriented induction motor drive”, IEEE Trans, on Power Electr., vol. 12, no. 5, pp. 772-778, 1997.

[153] F. Barrero, A. González, A. Torralba, E. Galván, and L. G. Franquelo, “Speed Control of Induction Motors Using a Novel Fuzzy Sliding-Mode Structure”, IEEE Transactions on Fuzzy Systems, Vol. 10, No. 3, pp. 375-383, JUNE 2002.

[154] Yang Liyong, Li Yinghong ; Chen Yaai ; Li Zhengxi “A Novel Fuzzy Logic Controller for Indirect Vector Control Induction Motor Drive” , IEEE World Congress on Intelligent Control and Automation, Chongqing, China, 25-27 June 2008.

[155] T. Mishra, A. Devanshu, N. Kumar, A. Kulkarni “Comparative Analysis of Hysteresis Current Control and SVPWM on Fuzzy Logic based Vector Controlled Induction Motor Drive”, IEEE 1st International Conference on Power Electronics, Intelligent Control and Energy Systems (ICPEICES), Delhi, India, 4-6 July 2016.

[156] M. Zaky , M. Metwaly, “A Performance Investigation of a Four-Switch Three-Phase Inverter-Fed IM Drives at Low Speeds Using Fuzzy Logic and PI Controllers”, IEEE Transactions on Power Electronics,Vol. 32 , No. 5, pp. 3741 – 3753, May 2017.

[157] H. Sathishkumar, S. S. Parthasarathy, “A novel neural network intelligent controller for vector controlled induction motor drive”, International Conference on A alternative Energy in Developing Countries an d Emerging Economies, Elsevier, Science Direct, Energy Procedia 138 , pp. 692-697, Bangkok, Thailan, 25-26 May 2017.

[158] A. Bohari, W. Utomo, Z. Haron, N. Zin, S. Sim, R. Ariff , “Improved FOC of induction motor with online neural network” WSEAS Transactions on Power Systems, Vol. 9, pp. 136-142, Malaysia, January 2014.

[159] Mohamadian, M.; Nowicki, E.; Ashrafzadeh, F.; Chu, A.; Sachdeva, R.; Evanik, E., "A Novel Neural Network Controller And Its Efficient DSP Implementation For VectorControlled Induction Motor Drives," IEEE Transactions on Industry Applications,

Vol.39, No.6, pp.1622-1629, Nov.-Dec. 2003.

[160] A. Saxena , S. Dutta , A. Chitra, "Artificial Neural Network Controller for Vector Controlled Induction Motor Drive", International Journal of Computer Applications, Vol. 46, no. 14,pp. 34-40, May 2012.

[161] A. Ba-razzouk, A. Cheriti, G. Olivier, "A Neural Networks Based Field Oriented Control Scheme for Induction Motors" IEEE Industry Applications Society Annual Meeting , pp. 804-811, New Orleans, Louisiana, October 5-9, 1997.

[162] F. Blaschke, "The Principle of Field Orientation as Applied to the New Transvector Closed-Loop Control System for Rotating Field Machines," Siemens Review, Vol. 34, pp. 217-220, May 1972.

[163] K. Hasse, "Zur dynamik drehzahlgeregelter antriebe mit stromrichtergespeisten asynchron-kurzschlussläufermaschinen," Dissertation Techn. Hochsch., Darmstadt, West Germany, 1969.

[164] Michael Zuercher- Martinson, "Harmonics and Noise in Photovoltaic (PV) Inverter and the Mitigation Strategies", Solectria Renewables. Soonwook Hong, Ph. D., White paper.

[165] D.J.Leith, WE.Leithead, "Survey of Gain-Scheduling Analysis & Design".

Appendices

Appendix A Parameters of Induction Motors

The parameters of of the 3Φ, 208 V, four poles, 60 Hz squirrel-cage induction motor used for the simulation studies and experimental work in the thesis are listed below.

Rated power	$P_{rat} = 0.25$ Hp
Rated stator voltage	$V_{rat\ L-L} = 208$ V
Rated frequency	$f = 60$ Hz
Rated speed	$\omega_r = 1670$ rev./min
Rated current	1.2 A
Stator resistance	$R_s = 12.5\ \Omega$
Stator leakage inductance	$L_{ls} = 0.03611$
Rotor resistance	$R_r = 3.833\ \Omega$
Rotor leakage inductance	$L_{lr} = 0.03611$H
Mutual inductance	$L_m = 0.4955H$
Number of pole pairs	$P = 2$
Moment of inertia of the rotor	$J = 0.05$ kgm^2
Coefficient of friction	$B = 0.001$

Appendix B List of Technical Papers

List of technical papers that have been written and published, related to this work

1. F. Lftisi, G.H. George, A. Aktaibi, C.B, Butt, M. A. Rahman, "Artificial Neural Network Based Speed Controller for Induction Motors," IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society, pp: 2708 – 2713, 2016.
2. Fuzi Lftisi; M. A. Rahman, "A Novel Finite Element Controller Map for Intelligent Control of Induction Motors",8th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON), PP: 18 – 24, 2017.
3. F. Lftisi, G.H. George, M.A. Rahman, "An Application of a Finite Element Controller Map for Speed Control for Saturated Induction Motors" IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society Pages: 8756 – 8762, 2017.
4. F. Lftisi; G. H. George; C. B. Butt; A. Aktaibi; M. A. Rahman, "Grid search optimization techniques for the indirect vector-controlled induction motor drives", IEEE Electrical Power and Energy Conference (EPEC), Pages: 1 – 6, 2017.
5. F. Lftisi, G.H. George, C.B. Butt and M. A. Rahman, "A Fast Fuzzy Logic Controller Based on Vector Control for Three Phases Induction Motor Drives", in Newfoundland Electrical and Computer Engineering Conference (NECEC), 2016, Newfoundland, Canada.
6. F. Lftisi, and M. A. Rahman, "Direct Torque Controlled Induction Motor Drive Using Hysteresis Comparators" in Newfoundland Electrical and computer Engineering Conference (NECEC), 2014, Newfoundland, Canada.

7. F. Lftisi, and M. A. Rahman, Advanced Control for Induction Motor Drives", in Newfoundland Electrical and computer Engineering Conference (NECEC), 2013, Newfoundland, Canada.

Appendix C

C.1 Command Current

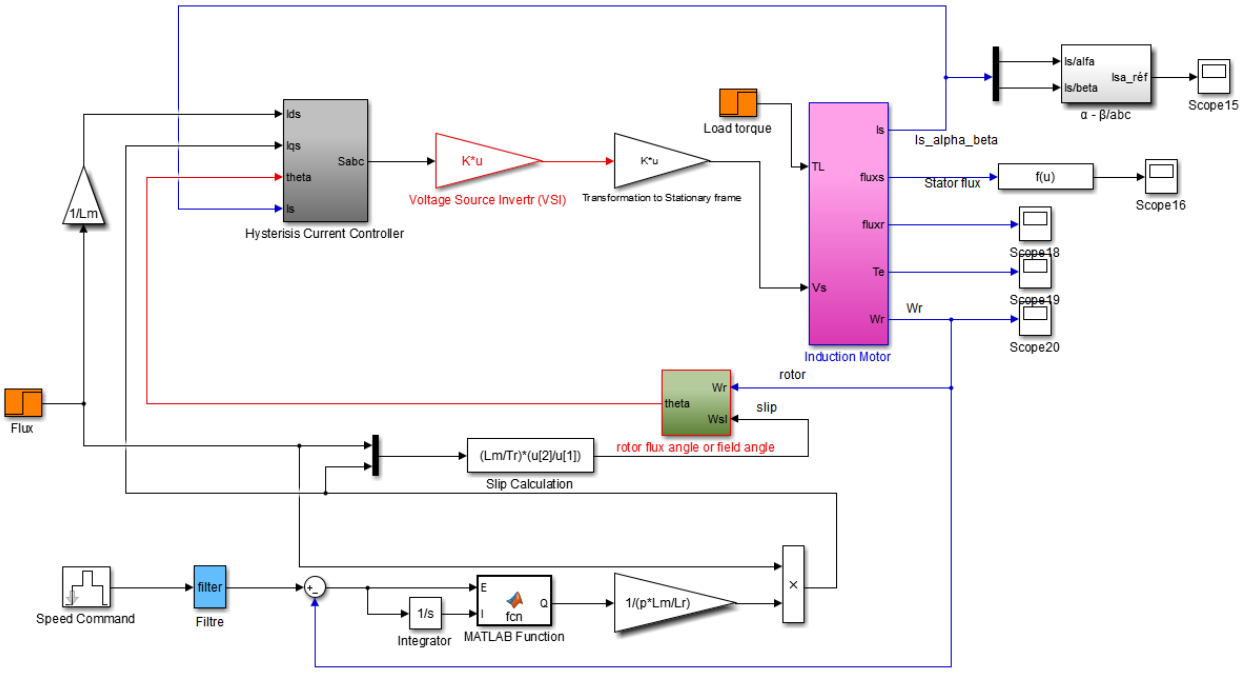


Figure C-9-1Schematic diagram of indirect field oriented control

The Command Current block generates the command currents necessary to maintain the desired rotor speed of the motor. To do this, rotor speed and rotor angular position are necessary inputs. The schematic of the Command Current Generator block is shown in figures C-1 (a) and C-1(b).

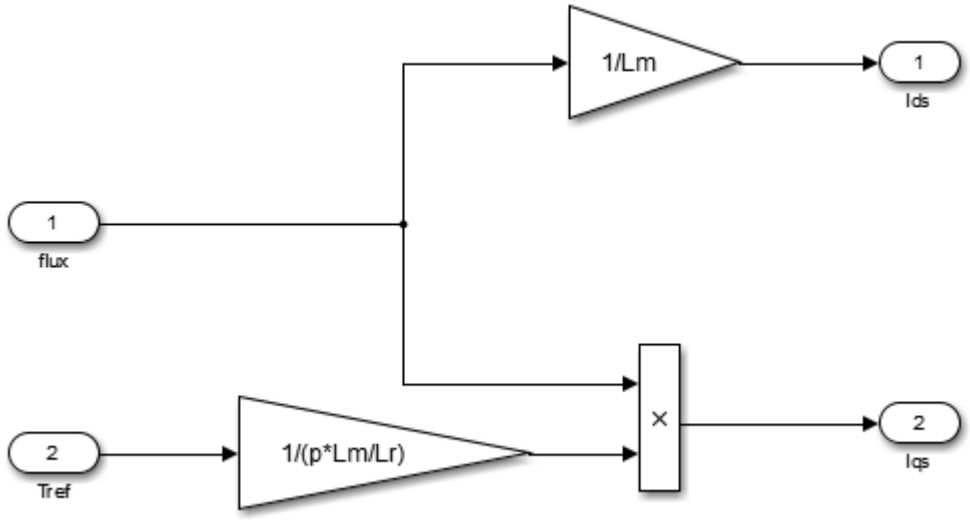


Figure C-1 (a) Current flux and torque components

The command torque necessary to achieve the desired rotor speed is calculated as explained in chapter 3. This command torque is then used to obtain i_{qs}^{*} and ω_{sl} , equation (3.23), (3.25). Then, i_{qs}^{*}, i_{ds}^{*} and i_{a}^{*}, i_{b}^{*}, and i_{c}^{*} are calculated. This block is shown in figures C-1(a) and C-1(b).

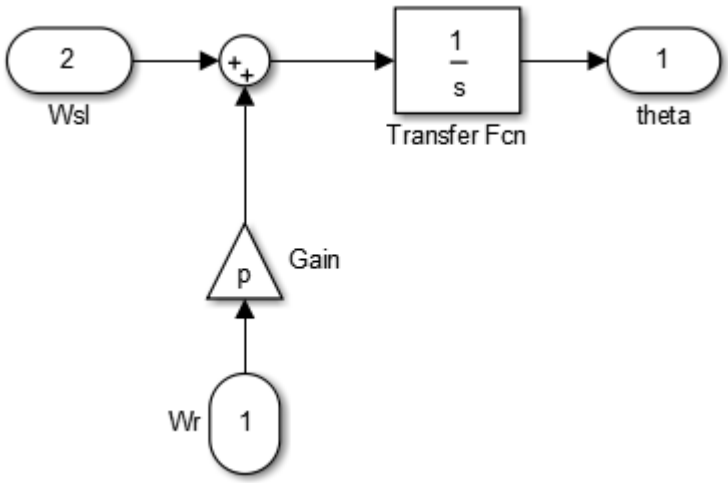


Figure C-1 (b) Rotor flux angle

C.2 Current Controller

The next block in the model represents the controller for the voltage source inverter.

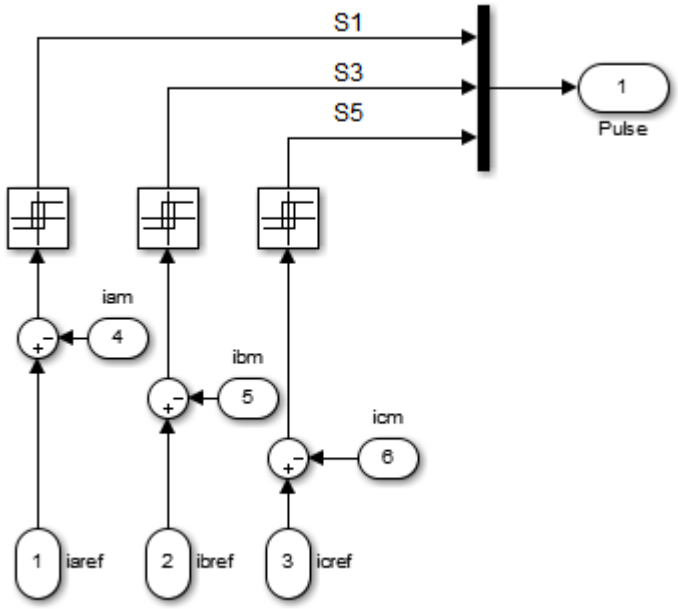


Figure C-2 Hysteresis current controller

C.3 Voltage Source Inverter

The next block in the model represents the voltage source inverter which designed according to the following equation, as indicated in the equation (C.1) and illustrated in figure C-3.

$$\begin{bmatrix} V_{an} \\ V_{bn} \\ V_{cn} \end{bmatrix} = \frac{V_{dc}}{3} \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix} \begin{bmatrix} NA \\ NB \\ NC \end{bmatrix}$$

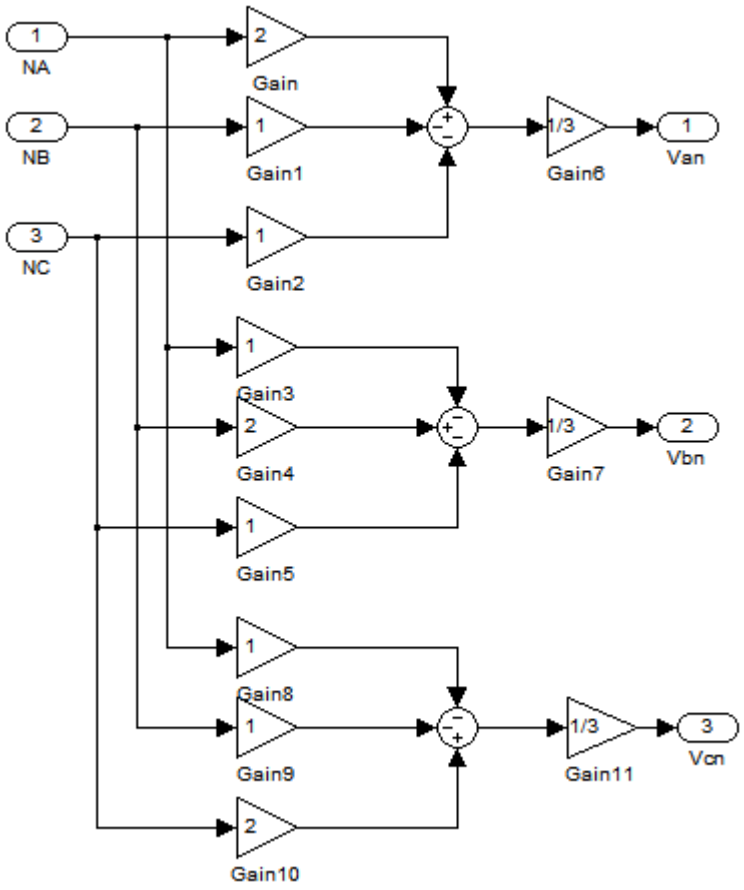


Figure C-3 voltage source inverter

C.4 Transformation to Stationary Frame

The IM model equations require $V_{\alpha s}$ and $V_{\beta s}$ to give the outputs ω_r and θ_r . Therefore, V_{an} , V_{bn} and V_{cn} , as obtained from the VSI, must be transformed to the stationary reference frame (also known as α - β axes) . This is done using equation (C.2).

$$\begin{bmatrix} v_\alpha \\ v_\beta \end{bmatrix} = \frac{2}{3}\begin{bmatrix} 1 & -1/2 & -1/2 \\ 0 & \sqrt{3}/2 & -\sqrt{3}/2 \end{bmatrix}\begin{bmatrix} v_a \\ v_b \\ v_c \end{bmatrix}$$

Appendix D

D.1 Fuzzy logic controller for four rules

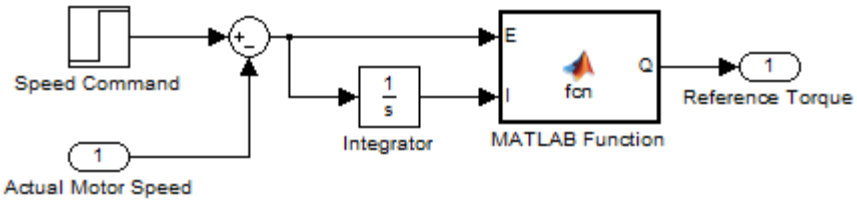


Figure D-1: Matlab function which generate the Fuzzy logic Program

```
function Q = fcn(E,I)
NE=0; NI=0; PI=0; PE=0; NN=0; NP=0; PN=0; PP=0;
% Fuzzy Logic
% PI Controller
% 4 Fuzzy Rules
% BE Error Break Point
% BI Integral Break Point
% Signal Break Points
% BQ1 BQ2
% Error E Integral D Control Q
% IF E IS N && I IS N THEN Q IS NN
% IF E IS N && I IS P THEN Q IS NP
% IF E IS P && I IS N THEN Q IS PN
% IF E IS P && I IS P THEN Q IS PP
% Data
BE=15; BI=0.5;
BQ1=10; BQ2=40;
GQ1=BQ1; GQ2=BQ2-BQ1;
HQ1=BQ1/2;HQ2=BQ1+(BQ2-BQ1)/2;
% Calculations
% Input Membership Values
if(E<-BE) NE=1.0; PE=0.0; end;
if(E>+BE) NE=0.0; PE=1.0; end;
if(E>=-BE && E<=+BE)
NE=1-(E+BE)/(2*BE);
PE=(E+BE)/(2*BE);
end;
if(I<-BI) NI=1.0; PI=0.0; end;
if(I>+BI) NI=0.0; PI=1.0; end;
if(I>=-BI && I<=+BI)
NI=1-(I+BI)/(2*BI);
PI=(I+BI)/(2*BI);
end;
% Output Membership Values
if(NE<NI) NN=NE; end;
if(NE>=NI) NN=NI; end;
if(NE<PI) NP=NE; end;
```

```
38 if(NE>=PI) NP=PI; end;
39 if(PE<NI) PN=PE; end;
40 if(PE>=NI) PN=NI; end;
41 if(PE<PI) PP=PE; end;
42 if(PE>=PI) PP=PI; end;
43 % Defuzzification
44 Q=(-NN*GQ2*HQ2-NP*GQ1*HQ1+PN*GQ1*HQ1+PP*GQ2*HQ2)/(NN*GQ2+NP*GQ1 ...
45 +PN*GQ1+PP*GQ2)
```

D.2 Fuzzy logic controller with nine rules used instead of PI controller

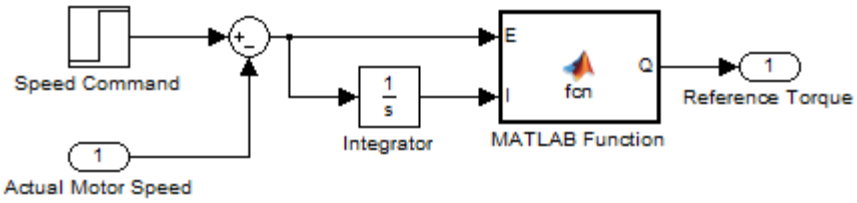


Figure D-2: Matlab function which generate 9 rules Fuzzy logic Program

```
1 function Q = fcn(E,I)
2 NE=0; NI=0; PI=0; PE=0;NZ=0;ZN=0;PZ=0;ZP=0;ZZ=0; NN=0; NP=0; PN=0;
PP=0;
3 % Fuzzy Logic
4 % PI Controller
5 % 9 Fuzzy Rules
6 NE=0;ZE=0;ZI=0; NI=0; PI=0; PE=0; NN=0; NP=0; PN=0; PP=0;
7 % BE Error Break Point
8 % BI Integral Break Point
9 % Signal Break Points
10% BQ1 BQ2 NQ3 BQ4 BQ5
11% Error E  Integral I  Control Q
12% IF E IS N && I IS N THEN Q IS NN
13% IF E IS N && I IS Z THEN Q IS NZ
14% IF E IS N && I IS P THEN Q IS NP
15% IF E IS Z && I IS N THEN Q IS ZN
16% IF E IS Z && I IS Z THEN Q IS ZZ
17% IF E IS Z && I IS P THEN Q IS ZP
18% IF E IS P && I IS N THEN Q IS PN
19% IF E IS P && I IS Z THEN Q IS PZ
20% IF E IS P && I IS P THEN Q IS PP
21% Data
22  BE=10; BI=1;
23  BQ1=10.0; BQ2=20.0; BQ3=30.0; BQ4=40.0; BQ5=50.0;
24  GQ1=BQ1*2; GQ2=BQ2-BQ1;
25  GQ3=BQ3-BQ2; GQ4=BQ4-BQ3;
26  GQ5=BQ5-BQ4; HQ1=0.0; HQ2=BQ1+(BQ2-BQ1)/2;
27  HQ3=BQ2+(BQ3-BQ2)/2; HQ4=BQ3+(BQ4-BQ3)/2;
28  HQ5=BQ4+(BQ5-BQ4)/2;
29  % Calculations
30% Input Membership Values
31  if(E<-BE) NE=1.0; ZE=0.0; PE=0.0; end;
32  if(E>+BE) NE=0.0; ZE=0.0; PE=1.0; end;
33  if(E>=-BE && E<=0.0)
34    NE=1-(E+BE)/BE;
35    ZE=(E+BE)/BE;
36    PE=0.0;
```

```
37  end;
38  if(E>0.0 && E<=+BE)
39    NE=0.0;
40    ZE=(BE-E)/BE;
41    PE=1-(BE-E)/BE;
42  end;
43  if(I<-BI) NI=1.0; ZI=0.0; PI=0.0; end;
44  if(I>+BI) NI=0.0; ZI=0.0; PI=1.0; end;
45  if(I>=-BI && I<=0.0)
46    NI=1-(I+BI)/BI;
47    ZI=(I+BI)/BI;
48    PI=0.0;
49  end;
50  if(I>0.0 && I<=+BI)
51    NI=0.0;
52    ZI=(BI-I)/BI;
53    PI=1-(BI-I)/BI;
54  end;
55% Output Membership Values
56  if(NE<NI) NN=NE; end;
57  if(NE>=NI) NN=NI; end;
58  if(NE<ZI) NZ=NE; end;
59  if(NE>=ZI) NZ=ZI; end;
60  if(NE<PI) NP=NE; end;
61  if(NE>=PI) NP=PI; end;
62  if(ZE<NI) ZN=ZE; end;
63  if(ZE>=NI) ZN=NI; end;
64  if(ZE<ZI) ZZ=ZE; end;
65  if(ZE>=ZI) ZZ=ZI; end;
66  if(ZE<PI) ZP=ZE; end;
67  if(ZE>=PI) ZP=PI; end;
68  if(PE<NI) PN=PE; end;
69  if(PE>=NI) PN=NI; end;
70  if(PE<ZI) PZ=PE; end;
71  if(PE>=ZI) PZ=ZI; end;
72  if(PE<PI) PP=PE; end;
73  if(PE>=PI) PP=PI; end;
74% Defuzzification
75  AREA=NN*GQ5+NZ*GQ4+NP*GQ3+ZN*GQ2+ZZ*GQ1+ZP*GQ2+PN*GQ3+PZ*GQ4+PP*GQ5;
76  MOMENT=-NN*GQ5*HQ5-NZ*GQ4*HQ4 ...
77   -NP*GQ3*HQ3-ZN*GQ2*HQ2 ...
78   +ZP*GQ2*HQ2+PN*GQ3*HQ3 ...
79   +PZ*GQ4*HQ4+PP*GQ5*HQ5;
80  Q=MOMENT/AREA
```

Appendix E

Neural Network Controller

```
% This is the code for Neural Network Controller used to imitate PI Controller
function o = fcn(i,j)
wbo=-100; wbx=0.0;
wby=0.000; wbz=0.0;
wix=0.128; wjx=0.0;
wiy=0.0; wjy=0.144;
wiz=0.0; wjz=0.0;
wxo=100.0; wyo=100.0;
wzo=0.0;
x=wix*i+wjx*j+wbx;
y=wiy*i+wjy*j+wby;
z=wiz*i+wjz*j+wbz;
xx=1.0/(1.0+exp(-x));
yy=1.0/(1.0+exp(-y));
zz=1.0/(1.0+exp(-z));
aa=wxo*xx;
bb=wyo*yy;
cc=wzo*zz;
o=aa+bb+cc+wbo;
```

Appendix F

F.1 Finite Element Controller Map for Nine Elements with Four Nodes

```
function control = fcn(error,sum)
LIMIT=50.0;
LEVEL=50.0;
KP=3.2; KI=3.6;
  %dd11=-149.6; dd12=-53.6; dd13=10.4; dd14=106.4;
  %dd21=-135.2; dd22=-39.2; dd23=24.8; dd24=120.8;
  %dd31=-120.8; dd32=-24.8; dd33=39.2; dd34=135.2;
  %dd41=-106.4; dd42=-10.4; dd43=53.6; dd44=149.6;
  %vv1=-6; vv2=-2; vv3=2; vv4=6;
  %hh1=-40; hh2=-10; hh3=+10; hh4=+40;
  dd11=-LIMIT-LEVEL; dd12=-LIMIT-LEVEL; dd13=+LIMIT-LEVEL;
dd14=+LIMIT-LEVEL;
  dd21=-LIMIT-LEVEL; dd22=-LIMIT-LEVEL; dd23=+LIMIT-LEVEL;
dd24=+LIMIT-LEVEL;
  dd31=-LIMIT+LEVEL; dd32=-LIMIT+LEVEL; dd33=+LIMIT+LEVEL;
dd34=+LIMIT+LEVEL;
  dd41=-LIMIT+LEVEL; dd42=-LIMIT+LEVEL; dd43=+LIMIT+LEVEL;
dd44=+LIMIT+LEVEL;
  vv2=-LEVEL/KI; vv1=20*vv2; vv3=+LEVEL/KI; vv4=20*vv3;
  hh2=-LIMIT/KP; hh1=20*hh2; hh3=+LIMIT/KP; hh4=+20*hh3;
  h1=0.0; h2=0.0; v1=0.0; v2=0.0;
  d1=0.0; d2=0.0; d3=0.0; d4=0.0;
  alpha=0.0; beta=0.0;
  for j=1:3
  for i=1:3
  if (j==1) v2=vv2;v1=vv1;end;
  if (j==2) v2=vv3;v1=vv2;end;
  if (j==3) v2=vv4;v1=vv3;end;
  if (i==1) h2=hh2;h1=hh1;end;
  if (i==2) h2=hh3;h1=hh2;end;
  if (i==3) h2=hh4;h1=hh3;end;
  if (error<h2 & error>h1) ...
      alpha=2*(error-h1)/ ...
      (h2-h1)-1;end;
  if (sum<v2 & sum>v1) ...
      beta=2*(sum-v1)/ ...
      (v2-v1)-1;end;
  if (j==1 & i==1) d1=dd11; ...
   d2=dd12; d3=dd21; d4=dd22; end;
  if (j==1 & i==2) d1=dd12; ...
   d2=dd13; d3=dd22; d4=dd23; end;
  if (j==1 & i==3) d1=dd13; ...
   d2=dd14; d3=dd23; d4=dd24; end;
  if (j==2 & i==1)  d1=dd21; ...
   d2=dd22; d3=dd31; d4=dd32; end;
  if (j==2 & i==2) d1=dd22; ...
   d2=dd23; d3=dd32; d4=dd33; end;
  if (j==2 & i==3) d1=dd23; ...
   d2=dd24; d3=dd33; d4=dd34; end;
  if (j==3 & i==1) d1=dd31; ...
   d2=dd32; d3=dd41; d4=dd42; end;
```

```
48  if (j==3 & i==2) d1=dd32; ...
49   d2=dd33; d3=dd42; d4=dd43; end;
50  if (j==3 & i==3) d1=dd33; ...
51   d2=dd34; d3=dd43; d4=dd44; end;
52  if (error<h2 & error>h1 & ...
53   sum<v2 & sum>v1) break; end;
54  end
55  if (error<h2 & error>h1 & ...
56   sum<v2 & sum>v1) break; end;
57  end
58  control=1/4*(1-alpha)*(1-beta)*d1 ...
59    +1/4*(1+alpha)*(1-beta)*d2 ...
60    +1/4*(1-alpha)*(1+beta)*d3 ...
61    +1/4*(1+alpha)*(1+beta)*d4;
```

F.2 Finite Element Controller Map for five Elements with Four Nodes

```
1 % this code is Finite elements controller map for 5 elements with 4
nodes
2
3 function control = fcn(error,sum)
4 d1=-149.6; d2=106.4;
5 d3=-39.2;d4=24.8;
6 d5=-24.8;d6=39.2;
7 d7=-106.4; d8=149.6;
8 v1=-6; v2=-2; v3=2; v4=6;
9 h1=-40; h2=-10; h3=10; h4=40;
10
11 %element=1;E0=0; I0=0; E1=0; E2=0; E3=0; E4=0;
12 %I1=0; I2=0; I3=0; I4=0;
13 i=error;
14 j=sum;
15
16 element=0;
17 % Slope between Elements 1#2 (v3-v1)/(h3-h1)=(vb-v1)/(h-h1);
18 h=error;
19 vb=v1+((v2-v1)*(h-h1)/(h2-h1));
20 % Slope between Elements 2#5 (v5-v7)/(h5-h7)=(vt-v7)/(h-h7);
21 vt=v4+((v3-v4)*(h-h1)/(h2-h1));
22 % Slope between Elements 1#4 (v4-v2)/(h4-h2)=(vB-v2)/(h-h2);
23 vB=v1+((v2-v1)*(h-h4)/(h3-h4));
24 % Slope between Elements 4#5 (v8-v6)/(h8-h6)=(vT-v6)/(h-h6);
25 vT=v3+((v4-v3)*(h-h3)/(h4-h3));
26 % if logical statement for finding Element #
27 if(error>h1 && error<h2)
28 if(sum>vb && sum<vt)
29 element=2;
30 end
31 if(sum<vb)
32 element=1;
33 end
34 if(sum>vt)
35 element=5;
```

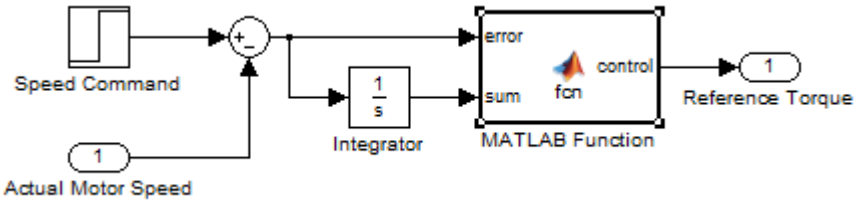
```
end
end

if(error>h2 && error<h3)
if(sum<v2)
element=1;
end
if(sum>v2 && sum<v3)
element=3;
end
if(sum>v3)
element=5;
end
end

if(error>h3 && error<h4)
if(sum<vB)
element=1;
end
```



```
if(sum>vB && sum<vT)
element=4;
end
if(sum>vT)
element=5;
end
end

% FINITE ELEMENTS
E0=0; E1=0; E2=0; E3=0; E4=0;
I0=0; I1=0; I2=0; I3=0; I4=0;
alpha=0; beta=0;

if (element == 1)
E1=h1; E2=h4; E3=h3; E4=h2;
I1=v1; I2=v1; I3=v2; I4=v2;
E0=error; I0=sum;
end

if (element == 2)
E1=h1; E2=h2; E3=h2; E4=h1;
I1=v1; I2=v2; I3=v3; I4=v4;
E0=error; I0=sum;
end

if (element == 3)
E1=h2; E2=h3; E3=h3; E4=h2;
I1=v2; I2=v2; I3=v3; I4=v3;
E0=error; I0=sum;
end

if (element == 4)
E1=h3; E2=h4; E3=h4; E4=h3;
I1=v2; I2=v1; I3=v4; I4=v3;
E0=error; I0=sum;
```

```
end

if(element == 5)
E1=h2; E2=h3; E3=h4; E4=h1;
I1=v3; I2=v3; I3=v4; I4=v4;
E0=error; I0=sum;
end
M=E0-1/4*(E1+E2+E3+E4);
X=1/4*(-E1+E2+E3-E4);
Y=1/4*(-E1-E2+E3+E4);
Z=1/4*(E1-E2+E3-E4);
N=I0-1/4*(I1+I2+I3+I4);
U=1/4*(-I1+I2+I3-I4);
V=1/4*(-I1-I2+I3+I4);
W=1/4*(I1-I2+I3-I4);

for IT=1:60
%
alpha=(M-Y*beta-Z*alpha*beta)/X;
beta=(N-U*alpha-W*alpha*beta)/V;
end

D=[d1 d2 d4 d3;d1 d3 d5 d7;d3 d4 d6 d5;d4 d2 d8 d6;d5 d6 d8 d7];

control=1/4*(1-alpha)*(1-beta)*D(element,1) ...
+1/4*(1+alpha)*(1-beta)*D(element,2) ...
+1/4*(1+alpha)*(1+beta)*D(element,3) ...
+1/4*(1-alpha)*(1+beta)*D(element,4);
```

F.3 Finite Element Controller for Five Elements with Eight Nodes

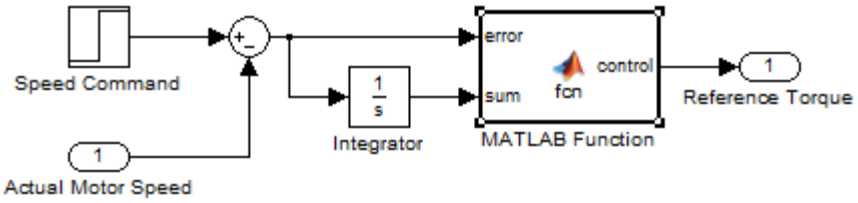



```
% This program is Finte Element Controller Map for Five Elements with
streight eight
nodes
function control = fcn(error,sum)
d1=-149.6; d2=-21.6;d3=106.4; d4=65.6;
d5=24.8; d6=-7.2; d7=-39.2; d8=-94.4;
d9=128; d10=149.6; d11=94.4;d12=39.2;
d13=32; d14=7.2;d15=-24.8; d16=-32;
d17=21.6; d18=-106.4; d19=-65.6; d20=-128;

v1=-6; v2=-4; v3=-2; v4=0;v5=2; v6=4; v7=6;
h1=-40; h2=-25; h3=-10; h4=0; h5=10; h6=25; h7=40;

element=0;
% Slope between Elements 1#2 (v3-v1)/(h3-h1)=(vb-v1)/(h-h1);
h=error;
vb=v1+((v3-v1)*(h-h1)/(h3-h1));
% Slope between Elements 2#5 (v5-v7)/(h5-h7)=(vt-v7)/(h-h7);
vt=v7+((v5-v7)*(h-h1)/(h3-h1));
% Slope between Elements 1#4 (v4-v2)/(h4-h2)=(vB-v2)/(h-h2);
vB=v1+((v3-v1)*(h-h7)/(h5-h7));
% Slope between Elements 4#5 (v8-v6)/(h8-h6)=(vT-v6)/(h-h6);
vT=v5+((v7-v5)*(h-h5)/(h7-h5));
% if logical statement for finding Element #
if(error>h1 && error<h3)
if(sum>vb && sum<vt)
element=2;
end
if(sum<vb)
element=1;
end
if(sum>vt)
```

```
element=5;
end
end

if(error>h3 && error<h5)
if(sum<v3)
element=1;
end
if(sum>v3 && sum<v5)
element=3;
end
if(sum>v5)
element=5;
end
end

if(error>h5 && error<h7)
if(sum<vB)
element=1;
end
if(sum>vB && sum<vT)
element=4;
end
if(sum>vT)
element=5;
end
end

% FINITE ELEMENTS
E0=0; E1=0; E2=0; E3=07; E4=0; E5=0; E6=0; E7=0; E8=0;
I0=0; I1=0; I2=0; I3=0; I4=0; I5=0; I6=0; I7=0; I8=0;
alpha=0; beta=0;

if (element == 1)
E1=h1; E2=h4; E3=h7; E4=h6; E5=h5; E6=h4; E7=h3; E8=h2;
I1=v1; I2=v1; I3=v1; I4=v2; I5=v3; I6=v3; I7=v3; I8=v2;
E0=error; I0=sum;
end

if (element == 2)
E1=h1; E2=h2; E3=h3; E4=h3; E5=h3; E6=h2; E7=h1; E8=h1;
I1=v1; I2=v2; I3=v3; I4=v4; I5=v5; I6=v6; I7=v7; I8=v4;
E0=error; I0=sum;
end

if (element == 3)
E1=h3; E2=h4; E3=h5; E4=h5; E5=h5; E6=h4; E7=h3; E8=h3;
I1=v3; I2=v3; I3=v3; I4=v4; I5=v5; I6=v5; I7=v5; I8=v4;
E0=error; I0=sum;
end

if (element == 4)
E1=h5; E2=h6; E3=h7; E4=h7; E5=h7; E6=h6; E7=h5; E8=h5;
I1=v3; I2=v2; I3=v1; I4=v4; I5=v7; I6=v6; I7=v5; I8=v4;
E0=error; I0=sum;
end
```

```
87
88 if(element == 5)
89 E1=h3; E2=h4; E3=h5; E4=h6; E5=h7; E6=h4; E7=h1; E8=h2;
90 I1=v5; I2=v5; I3=v5; I4=v6; I5=v7; I6=v7; I7=v7; I8=v6;
91 E0=error; I0=sum;
92 end
93
94 M=E0+1/4*(E1+E3+E5+E7)-1/2*(E2+E4+E6+E8);
95 C2=1/2*(E4-E8);
96 C3=1/2*(-E2+E6);
97 C4=1/4*(E1-E3+E5-E7);
98 C5=1/4*(E1+E3+E5+E7)-1/2*(E2+E6);
99 C6=1/4*(E1+E3+E5+E7)-1/2*(E4+E8);
100 C7=-1/4*(E1+E3-E5-E7)+1/2*(E2-E6);
101 C8=-1/4*(E1-E3-E5+E7)-1/2*(E4-E8);
102
103 N=I0+1/4*(I1+I3+I5+I7)-1/2*(I2+I4+I6+I8);
104 B2=1/2*(I4-I8);
105 B3=1/2*(-I2+I6);
106 B4=1/4*(I1-I3+I5-I7);
107 B5=1/4*(I1+I3+I5+I7)-1/2*(I2+I6);
108 B6=1/4*(I1+I3+I5+I7)-1/2*(I4+I8);
109 B7=-1/4*(I1+I3-I5-I7)+1/2*(I2-I6);
110 B8=-1/4*(I1-I3-I5+I7)-1/2*(I4-I8);
111
112
113 for IT=1:36
114 %
115 alpha=(M-C3*beta-C4*alpha*beta-C5*(alpha^2)-C6*(beta^2)...
116 -C7*beta*(alpha^2)-C8*alpha*(beta^2))/C2;
117
118 beta=(N-B2*alpha-B4*alpha*beta-B5*(alpha^2)-B6*(beta^2)...
119 -B7*beta*(alpha^2)-B8*alpha*(beta^2))/B3;
120
121 end
122
123 D=[d1 d2 d3 d4 d5 d6 d7 d8;d1 d8 d7 d16 d15 d19 d18 d20;
124 d7 d6 d5 d13 d12 d14 d15 d16;d5 d4 d3 d9 d10 d11 d12 d13;
125 d15 d14 d12 d11 d10 d17 d18 d19];
126
127 control=-1/4*(1-alpha)*(1-beta)*(1+alpha+beta)*D(element,1) ...
128 +1/2*(1-alpha)*(1+alpha)*(1-beta)*D(element,2) ...
129 -1/4*(1+alpha)*(1-beta)*(1-alpha+beta)*D(element,3) ...
130 +1/2*(1+alpha)*(1+beta)*(1-beta)*D(element,4)...
131 -1/4*(1+alpha)*(1+beta)*(1-alpha-beta)*D(element,5)...
132 +1/2*(1-alpha)*(1+alpha)*(1+beta)*D(element,6)...
133 -1/4*(1-alpha)*(1+beta)*(1+alpha-beta)*D(element,7)...
134 +1/2*(1-alpha)*(1+beta)*(1-beta)*D(element,8);
```

F.4 Finite Element Controller Map for Five Elements with Eight Nodes when the middle element takes a parabolic shape




```
% This is the Code for Finite % Element Controller Map for Five elements with eight
nodes when the middle element takes a parabolic shape.
function control = fcn(error,sum)
d1=-149.6; d2=-21.6;d3=106.4; d4=65.6; d5=24.8; d6=-3.6; d7=-39.2;
d8=-94.4;d9=128; d10=149.6; d11=94.4; d12=39.2; d13=28.8; d14=3.6;
d15=-24.8; d16=-28.8; d17=21.6; d18=-106.4; d19=-65.6; d20=-128;

v1=-6; v2=-4; v3=-2; v4=-1;v5=0; v6=1; v7=2; v8=4; v9=6;
h1=-40; h2=-25; h3=-10; h4=-9; h5=0; h6=9; h7=10; h8=25; h9=40;

element=0;

% Slope between Elements 1#2 (v3-v1)/(h3-h1)=(vb-v1)/(h-h1);
h=error;
vb=v1+((v3-v1)*(h-h1)/(h3-h1));
% Slope between Elements 2#5 (v5-v7)/(h5-h7)=(vt-v7)/(h-h7);
vt=v9+((v7-v9)*(h-h1)/(h3-h1));
% Slope between Elements 1#4 (v4-v2)/(h4-h2)=(vB-v2)/(h-h2);
vB=v1+((v3-v1)*(h-h9)/(h7-h9));
% Slope between Elements 4#5 (v8-v6)/(h8-h6)=(vT-v6)/(h-h6);
vT=v9+((v7-v9)*(h-h9)/(h7-h9));
% if logical statement for finding Element #
if(error>h1 && error<h4)
if(sum>vb && sum<vt)
element=2;
end
if(sum<vb)
element=1;
end
if(sum>vt)
element=5;
end
end

if(error>h4 && error<h6)
if(sum<v4)
element=1;
end
if(sum>v4 && sum<v6)
element=3;
end
```

```
41 if(sum>v6)
42 element=5;
43 end
44 end
45
46 if(error>h6 && error<h9)
47 if(sum<vB)
48 element=1;
49 end
50 if(sum>vB && sum<vT)
51 element=4;
52 end
53 if(sum>vT)
54 element=5;
55 end
56 end
57
58 % FINITE ELEMENTS
59 E0=0; E1=0; E2=0; E3=07; E4=0; E5=0; E6=0; E7=0; E8=0;
60 I0=0; I1=0; I2=0; I3=0; I4=0; I5=0; I6=0; I7=0; I8=0;
61 alpha=0; beta=0;
62
63 if (element == 1)
64 E1=h1; E2=h5; E3=h9; E4=h8; E5=h7; E6=h5; E7=h3; E8=h2;
65 I1=v1; I2=v1; I3=v1; I4=v2; I5=v3; I6=v4; I7=v3; I8=v2;
66 E0=error; I0=sum;
67 end
68
69 if (element == 2)
70 E1=h1; E2=h2; E3=h3; E4=h4; E5=h3; E6=h2; E7=h1; E8=h1;
71 I1=v1; I2=v2; I3=v3; I4=v5; I5=v7; I6=v8; I7=v9; I8=v5;
72 E0=error; I0=sum;
73 end
74
75 if (element == 3)
76 E1=h3; E2=h5; E3=h7; E4=h6; E5=h7; E6=h5; E7=h3; E8=h4;
77 I1=v3; I2=v4; I3=v3; I4=v5; I5=v7; I6=v6; I7=v7; I8=v5;
78 E0=error; I0=sum;
79 end
80
81 if (element == 4)
82 E1=h7; E2=h8; E3=h9; E4=h9; E5=h9; E6=h8; E7=h7; E8=h6;
83 I1=v3; I2=v2; I3=v1; I4=v5; I5=v9; I6=v8; I7=v7; I8=v5;
84 E0=error; I0=sum;
85 end
86
87 if(element == 5)
88 E1=h3; E2=h5; E3=h7; E4=h8; E5=h9; E6=h5; E7=h1; E8=h2;
89 I1=v7; I2=v6; I3=v7; I4=v8; I5=v9; I6=v9; I7=v9; I8=v8;
90 E0=error; I0=sum;
91 end
92
93 M=E0+1/4*(E1+E3+E5+E7)-1/2*(E2+E4+E6+E8);
94 C2=1/2*(E4-E8);
95 C3=1/2*(-E2+E6);
96 C4=1/4*(E1-E3+E5-E7);
```

```
C5=1/4*(E1+E3+E5+E7)-1/2*(E2+E6);
C6=1/4*(E1+E3+E5+E7)-1/2*(E4+E8);
C7=-1/4*(E1+E3-E5-E7)+1/2*(E2-E6);
C8=-1/4*(E1-E3-E5+E7)-1/2*(E4-E8);

N=I0+1/4*(I1+I3+I5+I7)-1/2*(I2+I4+I6+I8);
B2=1/2*(I4-I8);
B3=1/2*(-I2+I6);
B4=1/4*(I1-I3+I5-I7);
B5=1/4*(I1+I3+I5+I7)-1/2*(I2+I6);
B6=1/4*(I1+I3+I5+I7)-1/2*(I4+I8);
B7=-1/4*(I1+I3-I5-I7)+1/2*(I2-I6);
B8=-1/4*(I1-I3-I5+I7)-1/2*(I4-I8);

for IT=1:36
%
alpha=(M-C3*beta-C4*alpha*beta-C5*(alpha^2)...
-C6*(beta^2)-C7*beta*(alpha^2)-C8*alpha*(beta^2))/C2;

beta=(N-B2*alpha-B4*alpha*beta-B5*(alpha^2)...
-B6*(beta^2)-B7*beta*(alpha^2)-B8*alpha*(beta^2))/B3;

end

D=[d1 d2 d3 d4 d5 d6 d7 d8;d1 d8 d7 d16 d15 d19 d18 d20;
d7 d6 d5 d13 d12 d14 d15 d16;d5 d4 d3 d9 d10 d11 d12 d13;
d15 d14 d12 d11 d10 d17 d18 d19];

control=-1/4*(1-alpha)*(1-beta)*(1+alpha+beta)*D(element,1) ...
+1/2*(1-alpha)*(1+alpha)*(1-beta)*D(element,2) ...
-1/4*(1+alpha)*(1-beta)*(1-alpha+beta)*D(element,3) ...
+1/2*(1+alpha)*(1+beta)*(1-beta)*D(element,4)...
-1/4*(1+alpha)*(1+beta)*(1-alpha-beta)*D(element,5)...
+1/2*(1-alpha)*(1+alpha)*(1+beta)*D(element,6)...
-1/4*(1-alpha)*(1+beta)*(1+alpha-beta)*D(element,7)...
+1/2*(1-alpha)*(1+beta)*(1-beta)*D(element,8);
```