



University  
of Glasgow

Unar, Mukhtiar Ali (1999) *Ship steering control using feedforward neural networks*.

PhD thesis

<http://theses.gla.ac.uk/4493/>

Copyright and moral rights for this thesis are retained by the author

A copy can be downloaded for personal non-commercial research or study, without prior permission or charge

This thesis cannot be reproduced or quoted extensively from without first obtaining permission in writing from the Author

The content must not be changed in any way or sold commercially in any format or medium without the formal permission of the Author

When referring to this work, full bibliographic details including the author, title, awarding institution and date of the thesis must be given

**SHIP STEERING CONTROL USING  
FEEDFORWARD NEURAL NETWORKS**

Mukhtiar Ali Unar

Department of Electronics and Electrical Engineering  
University of Glasgow

A thesis submitted for the degree of  
Doctor of Philosophy  
to the University of Glasgow

## ABSTRACT

*One significant problem in the design of ship steering control systems is that the dynamics of the vessel change with operating conditions such as the forward speed of the vessel, the depth of water and loading conditions etc. Approaches considered in the past to overcome these difficulties include the use of self adaptive control systems which adjust the control characteristics on continuous basis to suit the current operating conditions.*

*Artificial neural networks have been receiving considerable attention in recent years and have been considered for a variety of applications where the characteristics of the controlled system change significantly with operating conditions or with time. Such networks have a configuration which remains fixed once the training phase is complete. The resulting controlled systems thus have more predictable characteristics than those which are found in many forms of traditional self-adaptive control systems. In particular, stability bounds can be investigated through simulation studies as with any other form of controller having fixed characteristics.*

*Feedforward neural networks have enjoyed many successful applications in the field of systems and control. These networks include two major categories: multilayer perceptrons and radial basis function networks. In this thesis, we explore the applicability of both of these artificial neural network architectures for automatic steering of ships in a course changing mode of operation.*

*The approach that has been adopted involves the training of a single artificial neural network to represent a series of conventional controllers for different operating conditions. The resulting network thus captures, in a*

*nonlinear fashion, the essential characteristics of all of the conventional controllers. Most of the artificial neural network controllers developed in this thesis are trained with the data generated through simulation studies. However, experience is also gained of developing a neuro controller on the basis of real data gathered from an actual scale model of a supply ship.*

*Another important aspect of this work is the applicability of local model networks for modelling the dynamics of a ship. Local model networks can be regarded as a generalized form of radial basis function networks and have already proved their worth in a number of applications involving the modelling of systems in which the dynamic characteristics can vary significantly with the system operating conditions. The work presented in this thesis indicates that these networks are highly suitable for modelling the dynamics of a ship.*

## ACKNOWLEDGEMENTS

My sincere gratitude goes to my supervisor, Professor David J. Murray-Smith, for his guidance, encouragement, inspiration and patience throughout the course of my PhD work. In particular, his vision in, and commitments to, the research and his share of knowledge and time that has made the research a success.

I would like to thank Professor John O'Reilly, Professor S. Beaumont and Professor C. Wilkinson for their valuable comments and suggestions about this work.

My deepest thanks go to Professor A.R. Memon, Vice Chancellor, Mehran University of Engineering and Technology (MUET), Jamshoro, Pakistan, for his encouragement and inspiration. His knowledge and expertise has helped me to extend my abilities in the field of Systems and Control.

I would also like to thank Professor P.J.R. Laybourn, Dr. J.J. Gribble, Dr. M. Hersh, Dr. E. Acha, Dr. S.H. Siddiqui and Mr. Tom O'hara for their encouragement, co-operation and help.

Special acknowledgement and appreciations are due to Professor P.J. Gawthrop, Dr. Gary J. Gray, Dr. R. Murray-Smith and Dr. Henrik Gollee for their assistance in local model networks.

In addition, I would like to thank Dr. E.W. McGookin, who was kind enough to provide me with the real data which he gathered while testing his sliding mode controller on scale model of a supply ship at the Guidance, Navigation and Control Laboratory, Department of Engineering Cybernetics, Norwegian University of Science and Technology, Trondheim.

In respect of funding I would like to thank the Ministry of Education, Government of Pakistan for the scholarship to carryout this research. I am

also thankful to my employer, MUET Jamshoro Pakistan for granting me study leave with pay for this research. Moreover, I would like to thank the Department of Electronics and Electrical Engineering, University of Glasgow, for funding my attendance at the 3<sup>rd</sup> and 4<sup>th</sup> International Conference on Engineering Applications of Neural Networks in Stockholm and Gibraltar respectively.

Special thanks to all my friends in the Department of Electronics and Electrical Engineering, University of Glasgow for their company, friendliness, and sincere help. In particular, Munir Chowdhury, Jay Sigbrandt, Tariq Lodhi and Seelan Sundaralingham.

On a more personal note my deepest feelings of thanks to my parents, brothers, sisters, and other relatives for their love and support during my studies. Special thanks to my wife Sumera for her unceasing and invaluable support, despite her poor health. Deepest love for my little sons Shahroz and Shahmir who made my life happy and enjoyable, though I could not give them time and attention they deserved.

**To my first teacher Pir Bux Unar**

## TABLE OF CONTENTS

<b>1. Introduction.....</b>	<b>1</b>
1.1 Background.....	1
1.2 Thesis Contributions.....	2
1.3 Thesis Structure .....	4
<b>2. Ship Dynamics.....</b>	<b>7</b>
2.1 Basic Equations of Motion.....	8
2.2 Linear Models.....	11
2.2.1 The Model of Davidson and Schiff.....	12
2.2.2 Nomoto's Models.....	14
2.3 Normalization.....	17
2.4 Non-linear Models.....	21
2.4.1 The Model of Bech and Smith.....	22
2.4.2 Norbbin's Model.....	23
2.5 Full Scale Trials.....	24
2.5.1 Turning Circle Manoeuvre.....	24
2.5.2 Zig-Zag Manoeuvre.....	26
2.5.3 Spiral Manoeuvre.....	27
2.5.4 The Pull-Out Manoeuvre.....	28
2.5 Parameter Variations.....	30
2.5.1 Forward Speed Effects.....	30
2.5.2 Effects of the depth of water.....	31
2.5.3 Effects of different loading conditions.....	33
<b>3. Automatic Steering of Ships.....</b>	<b>35</b>
3.1 Ship Steering Control.....	36
3.2 Reference Model.....	36

3.3 Steering Machine .....	38
3.4 Ship Autopilot .....	39
3.5 Very Early Autopilots .....	39
3.6 PD/PID Autopilots .....	40
3.6.1 PID parameters in terms of the damping ratio and the undamped natural frequency .....	42
3.6.2 Internal Model Control Laws .....	45
3.6.3 Controller Design Using Optimal Control Theory.....	47
3.6.4 Pseudo derivative Feedback Algorithm.....	50
3.7 Non-Linear Autopilots .....	50
3.7.1 Autopilot based on feedback linearization control laws.....	51
3.7.2 Sliding Mode Control.....	53
3.8 Adaptive Autopilots.....	54
3.8.1 Model Reference Adaptive Control .....	56
3.9 Intelligent Autopilots .....	59
<b>4. Artificial Neural Networks.....</b>	<b>61</b>
4.1 Historical Motivation .....	61
4.2 Biological Neuron.....	65
4.3 Model of a Single Artificial Neuron.....	67
4.4 ANN Architectures .....	70
4.5 Multilayer Feedforward Neural Networks .....	71
4.6 Multilayer Perceptron Networks .....	73
4.6.1 Error Back-Propagation Algorithm .....	74
4.6.1.1 Derivation .....	75
4.6.2 Improved Back-Propagation .....	82
4.6.3 Back-Propagation with Levenberg-Marquardt Algorithm .....	83
4.6.4 Approximation Capabilities of MLP Networks .....	85
4.7 Radial Basis Function Networks.....	88

4.7.1 Training .....	90
4.7.1.1 Orthogonal Least Squares Method .....	91
4.7.2 Approximation Capabilities of RBF Networks .....	95
4.8 Discussion .....	95
<b>5. Development of Multilayer Perceptron Networks.....</b>	<b>98</b>
5.1 Methodology .....	98
5.2 Assumptions .....	100
5.3 ROV Zeefakkel.....	101
5.4 Mariner Class Merchant Ship .....	109
5.5 210,000 dwt Tanker.....	111
5.6 Reducing the size of the training data set.....	113
5.7 Back-Propagation with Levenberg-Marquardt Algorithm .....	116
5.8 Conclusions .....	118
<b>6. Ship Steering Control Using RBF Networks.....</b>	<b>120</b>
6.1 ROV Zeefakkel.....	121
6.2 Mariner Class Merchant Ship .....	123
6.3 210,000 dwt Tanker.....	124
6.4 Performance of RBF network at different loading conditions.....	125
6.5 Performance of RBF controller at different depths of water .....	129
6.6 Conclusions .....	134
<b>7. An Application Involving Real Data .....</b>	<b>136</b>
7.1 Ship Model Dynamics .....	137
7.2 Thruster Dynamics.....	139
7.3 Sliding Mode Controller Design .....	140
7.4 Development of ANN Controller .....	142
7.5 Summary .....	146

<b>8. Local Model Networks .....</b>	<b>147</b>
8.1 Local Model Networks .....	147
8.2 Literature of Local Model Networks in Learning and Modelling.....	150
8.3 Advantages of LMNs.....	151
8.4 LMNS for Modelling the Ship Dynamics.....	152
8.5 Simulation Studies .....	153
8.6 Conclusions .....	157
<b>9. Conclusions and Further Work.....</b>	<b>159</b>
9.1 Conclusions .....	159
9.1.1 MLP Networks.....	160
9.1.2 Radial Basis Function Networks.....	161
9.1.3 Experience with real data .....	162
9.1.4 Local Model Networks .....	163
9.2 Suggestions for further work.....	163
9.2.1 Development of Neuro Autopilots in presence of disturbances	163
9.2.2 Development of ANNs from real data .....	164
9.2.3 On-Line Training.....	164
9.2.4 Modelling the Ship Dynamics.....	164
9.2.5 Rudder Roll Stabilization .....	165
9.2.6 Track Keeping Autopilots .....	165
<b>References .....</b>	<b>166</b>
<b>Appendices.....</b>	<b>194</b>
Appendix A: Internal Model Control.....	194
Appendix B: Derivation of SM controller.....	198
Appendix C: Levenberg-Marquardt Algorithm.....	202
Appendix D: Supply Ship Model.....	204

## LIST OF TABLES

2.1 Motions Nomenclature	9
2.2 Model parameters for different ships	17
2.3 Prime non-dimensionalized coefficients	18
2.4 Normalization variables used for the Bis system	19
2.5 Parameters and eigen values of the MARINER ship at 7 knots at various depth to draft (H/DT) ratios	32
2.6 The parameters of the tanker TOKYO MARU at 7 knots at different H/DT ratios	33
2.7 Parameters of a tanker at different loading conditions	34
5.1 Parameter variations with respect to the forward speed of the ship	107
6.1 Comparison of MLP1 & MLP2 with RBF network for ROV Zeefakkel	123
6.2 Comparison of RBF network with MLP1 & MLP2 networks for Mariner class merchant ship	123
6.3 Comparison of RBF network with MLP1 and MLP2 networks for 210,000 dwt tanker	124
6.4 Parameters of a tanker at different loading conditions at 8 m/s	125
6.5 Parameters of mariner type ship at a speed of 12 knots	130
7.1 Optimized parameter values of SMC	141

## **List of Figures**

2.1 Ship fixed and Earth fixed axes	8
2.2 Orientation of Earth fixed axes and ship fixed axes. Variables used to describe the motion in the horizontal plane	10
2.3 Turning circle manoeuvre	25
2.4 Zig-zag manoeuvre	27
2.5 (a) Spiral curve for a course stable ship	28
(b) spiral curve for a course unstable ship	28
(c) Bech's reverse spiral curve	28
2.6 Pull-out manoeuvre for (a) stable ship (b) unstable ship	29
2.7 Logarithmic presentation of the pull out manoeuvre	29
3.1 Ship Steering Control System	36
3.2 Simplified diagram of the steering machine	38
3.3 Heading and rudder response of an oil tanker at a speed of 8.1 m/sec for a reference heading of $20^{\circ}$ .	44
3.4 Heading and rudder response of oil tanker at a speed of 8.1 m/sec for a reference heading of $50^{\circ}$ .	45
3.5 Performance of the PID controller tuned using the IMC laws	47
3.6 Performance of the optimal controller of Equation (3.19)	49
3.7 Basic block diagram showing structure of PDF controller	50
3.8 Performance of feedback linearization controller at 5 m/sec.	53
4.1 A simplified view of a biological neuron	65

4.2 Model of a single neuron	68
4.3 Some choices of activation functions	69
4.4 An alternative model of a single (artificial) neuron	70
4.5 A simple three layer feedforward neural network	72
4.6 Dynamics of output neuron $j$ of an MLP network	76
4.7 Hidden neuron $j$ and output neuron $k$ of an MLP network	80
5.1 Supervised learning of an MLP network	99
5.2 MLP network for ROV Zeefakkel	102
5.3 Matching with training data	104
5.4 Performance of MLP network for ROV Zeefakkel at 5m/s	104
5.5 Performance of MLP network for ROV Zeefakkel at 10m/s	105
5.6 Performance of MLP network for ROV Zeefakkel at 8m/s	106
5.7 Performance of MLP network for ROV Zeefakkel at 3m/s	108
5.8 Performance of MLP network for ROV Zeefakkel at 15m/s	108
5.9 MLP network for mariner class merchant ship	109
5.10 Performance of MLP controller for Marine Class Merchant ship at a speed of 5m/s	110
5.11 Performance of MLP controller for mariner class merchant ship at a speed of 12m/s	111
5.12 Performance of MLP controller at 6m/s for 210,000 dwt tanker	112
5.13 Performance of MLP controller at 10 m/s for 210,000 dwt tanker	112
5.14 Performance of MLP network at 5m/s for ROV Zeefakkel with reduced data for training	115
5.15 Comparison of back-propagation (BP) with adaptive learning rate and momentum and BP with L-M algorithm for ROV Zeefakkel at 10m/s.	117

5.16 Comparison of the back-propagation algorithm with adaptive learning rate and momentum, and the back-propagation with the L-M algorithm for 210,000 dwt tanker.	117
6.1 Comparison of MLP and RBF network at a speed of 7m/sec. for ROV Zeefakkel	122
6.2 Comparison of MLP1 and RBF network at 12 m/sec for ROV Zeefakkel	122
6.3 Comparison of RBF and MLP1 at 5 m/sec. for mariner class merchant ship	124
6.4 Comparison of RBF and MLP1 network for 210,000 dwt tanker at a speed of 5m/sec.	125
6.5 Performance of RBF controller at various speeds under loading condition OC1	126
6.6 Performance of RBF controller at loading condition OC2	127
6.7 Performance of RBF controller at operating condition OC3	128
6.8 Performance of RBF controller under loading condition OC4	129
6.9 Performance of RBF controller at $H/DT = 1.93$	131
6.10 Performance of RBF controller at $H/DT = 1.5$	132
6.11 Performance of RBF controller at $H/DT = 2.5$	133
6.12 Performance of RBF controller at $H/DT = \infty$	134
7.1 CyberShip1	137
7.2 Thruster configuration	139
7.3 Ship Model and Course Changing Controller Configuration	142
7.4 Matching of data	143
7.5 Performance of RBF controller for CyberShip1 for a $20^0/-20^0$ manoeuvre	144
7.6 Performance of RBF controller for a $40^0/-40^0$ manoeuvre	145
7.7 Performance of RBF controller for a $-5^0/+15^0$ manoeuvre	145

8.1 General architecture of LMN	148
8.2 Closed loop system when ship is represented by an LMN	155
8.3 Performance of LMN for ROV Zeefakkel	155
8.4 Comparison of LMN with Nomoto's second order (conventional) model	156
8.5 Comparison of LMN & Nomoto's second order model for 210,000 dwt tanker	157

## ABBREVIATIONS

ADALINE	Adaptive Linear Element
AI	Artificial Intelligence
ANN	Artificial Neural Network
BP	Backpropagation
DPK	Dynamic Position Keeping
FB	Feedback Linearization
GNC	Guidance, Navigation and Control
IMC	Internal Model Control
INNS	International Neural Network Society
LM	Least Squares
LMN	Local Model Network
LMS	Least Mean Squares
L-M	Levenberg-Marquardt
MLP	Multilayer Perceptron
MRAC	Model Reference Adaptive Control
OC	Operating Condition
OLS	Orthogonal Least Squares
PD	Proportional-Derivative
PDF	Pseudo Derivative Feedback
PDP	Parallel Distributing Process
PID	Proportional-Integral-Derivative
RBF	Radial Basis Function
ROV	Remotely Operated Vehicle
RRS	Rudder Roll Stabilization
SMC	Sliding Mode Control

## *Chapter 1*

### **INTRODUCTION**

#### **1.1 Background:**

Although the history of ships and sailing is spread over centuries, the history of ship autopilots is not more than 77 years old. Minorsky's work [Minorsky, 1922] on automatic ship steering was one of the principal contributions to the early literature in the general field of automatic control. In the same year, Sperry [Sperry, 1922] introduced the first automatic steering control system for ships. These early autopilots were purely mechanical in construction and they provided a very simple steering action, the rudder demand being proportional to the heading error. To prevent oscillatory behaviour, a low gain was selected which rendered the device useful only in the course keeping mode, where there was no significant desire for a high degree of accuracy in the response. When proportional-integral-derivative (PID) controllers became commercially available, they greatly improved the performance and until the 1980s almost all makes of autopilots were based on these controllers. The main disadvantage of PID controllers is that they require manual adjustments to compensate for changes of operating conditions as well as environmental conditions, but these controller settings are in any case usually not optimal for the ship. The adjustments are time consuming and tedious. Furthermore, the PID autopilots can cause difficulties when the ship makes large manoeuvres involving nonlinear dynamic behaviour.

To avoid these problems of fixed structure PID controllers, adaptive autopilots were introduced in the 1970s and have remained a major area of

research until recently [Honderd and Winkelman, 1972; Oldenburgh, 1975; Ohtsu and Kitagawa, 1978; Sugimoto and Kojima, 1978; Källström, 1979; Åström, 1980; Van Amerongen, 1982; Arie et al. 1986; Katebi and Byrne, 1988; Fossen and Paulsen, 1993; Garcia and Castelo, 1995]. It is because of their significant benefits such as improved fuel economy, increased speed of the vessel, and reduced manual settings to compensate for changes in operating and environmental conditions that they are attractive for such applications. However, there is some concern about potential instabilities and other problems associated with adaptive system behaviour.

Artificial neural networks (ANNs) appear to offer some advantages over other forms of control for ship steering. This is because of the ability of neural networks to handle variations of plant dynamics without the element of unpredictability that may cause concern when adaptive control is considered for safety critical applications. An ANN controller can be trained so that it has the best properties of both of a constant parameter controller as well as of an adaptive controller. The reason is that once a neural network is trained the parameters are fixed. On the other hand, the network can be trained for a range of operating conditions so that the network's behaviour changes just like an adaptive controller. Witt et al. [1994, 1995] report that a neuro controller can improve the profit margin of the vessel and contribute to the safety of the vessel by: (i) reducing manual levels required on the bridge (ii) achieving a fuel saving by allowing the vessel to stay on course with little deviation and (iii) providing accurate steering in an environment of increased traffic density and close proximity of obstacles.

## **1.2 Thesis Contributions:**

When this Ph.D. project started in October 1995, the investigations on the applicability of artificial neural networks to a ship steering control system were in an embryonic state, and only some initial investigations had been

carried-out by a few researches [e.g. Endo et al., 1989; Richter and Burns, 1993; Witt et al., 1994; Simensen and Murray-Smith, 1995]. The situation remains almost the same even today, though some more papers have been published during the past three years. All of these papers have made use of a multilayer perceptron (MLP) architecture of feedforward neural networks and have trained the networks by using the well known back-propagation learning algorithm. To the best of our knowledge, no author has investigated any other architecture of artificial neural networks for this specific application. Moreover, every author has investigated the potential of ANNs for only one particular ship. Are neuro controllers suitable for all small and large ships? The literature fails to provide a confident answer to this question. In this thesis, we have investigated the potential of artificial neural networks for a number of ships, ranging from a small ferry of length 45 m to large tankers of length more than 300 m. We have not only developed ANNs on the basis of data generated by means of simulation studies, but also on the basis of real data gathered from a physical scale model of a ship. This is probably the first work which shows that ANNs can provide satisfactory performance for different types of ships, whether they are small or large, or whether they are rudder driven ships or thruster driven ships.

As mentioned above, the previous authors have trained their MLP networks by using the conventional back-propagation learning algorithm. The work presented in this thesis is different from the earlier research in a number of ways.

1. We have not only trained MLP networks by using the back-propagation learning algorithm with adaptive learning rate and momentum, but also with what is widely regarded as the fastest approach - the back-propagation incorporating the Levenberg-Marquardt algorithm. No one

had previously used this version of the back-propagation algorithm for this type of application.

2. This is the first thesis, which has investigated the potential of radial basis function (RBF) networks for this application. Because of their distinctive properties of best approximation, simple network structure and efficient learning procedure, these networks are more powerful than the MLP networks.
3. A significant aspect of this work is the applicability of ANNs for modelling the ship dynamics. For this purpose, we have investigated the potential of local model networks for the application. Local model networks or operating regime models, have an architecture which relates closely to ANNs. Such networks have already proved to be of value in other applications involving the modelling of systems in which the dynamic characteristics can vary significantly with the system operating condition. To the best of our knowledge, no one has investigated the potential of any architecture of ANNs for modelling ship dynamics. From this point of view, our work is significantly original and pioneering.

### **1.3 Thesis Structure:**

Chapter 2 describes ship dynamics in some detail. A number of linear and non-linear ship models are reviewed with emphasis on those models which are particularly suitable for control purposes. The Chapter also describes the variations of ship parameters with operating conditions. The operating conditions include the forward speed of the ship, the depth of water and loading conditions. The issues relating to the normalization of ship parameters are also discussed.

Chapter 3 gives an overview of a ship steering control system with emphasis on controller (i.e. autopilot) design. Developments in the design of ship autopilots are discussed. These developments include the design of very early autopilots (i.e. proportional controllers), PD/PID autopilots, Non-linear autopilots, and adaptive autopilots. The limitations/disadvantages of these autopilots are pointed out. Finally, advantages of intelligent autopilots are listed which make them superior to self adaptive control systems.

Chapter 4 serves as an introduction to artificial neural networks. Basic concepts of the theory are introduced and a brief history of ANNs is presented. The Chapter is mainly concerned with feedforward neural networks. Both MLP as well as RBF networks are described in detail. The training issues, function approximation capabilities and other features of these networks are discussed. The back-propagation algorithm is derived for the training of MLP networks. It is also explained how the training speed of the algorithm can be improved by introducing adaptive learning rate and momentum. The incorporation of the Levenberg-Marquardt algorithm into the conventional back-propagation algorithm is also covered. The orthogonal least squares algorithm is derived for the training of RBF networks. This Chapter is especially suitable for those who have little or no knowledge of ANNs.

Chapter 5 discusses the procedure involved in the development of MLP networks for ship steering control systems. The networks are developed for three different ships of length 45 m, 161 m and 310 m. The networks have been developed in such a way that they provide satisfactory performance even when the ship dynamics change with the forward speed of the ship. The networks are developed by using the back-propagation algorithm with adaptive learning rate and momentum. They are also developed by using the back-propagation algorithm with the incorporation of Levenberg-Marquardt

theorem. It is demonstrated that MLP networks with only one hidden layer are sufficient to allow satisfactory performance to be achieved.

The applicability of RBF networks is investigated in Chapter 6. The RBF controllers are developed and their performance is compared with that achieved with MLP networks. It is demonstrated that RBF networks can provide satisfactory performance when the depth of water or loading conditions change along with the forward speed of the ship.

In Chapter 7, an RBF controller is developed by using the real data which were gathered while testing a sliding mode controller on an actual scale model of a supply ship in the Guidance, Navigation and Control Laboratory at the Norwegian University of Science and Technology, Trondheim. The dynamics of supply ships, which are different from those of rudder steered ships, are described and it is explained how a sliding mode controller can be designed for such a ship. Then, a radial basis function network is developed which mimics the dynamics of the sliding mode controller. The performance of the network is illustrated by means of simulation studies.

Chapter 8 is devoted to local model networks (LMNs). The main objective of the Chapter is to explore the worth of LMNs for modelling the ship dynamics. A brief overview of LMNs is presented and major advantages of LMNs over other architectures of ANNs are discussed. The Chapter contains two simulation studies which show that these networks are an effective way to model the ship dynamics which vary with the forward speed of the ship.

Finally the thesis ends with Chapter 9 which presents the conclusions drawn from this work and suggests additional work which could be used to further develop key aspects of this research.

## *Chapter 2*

### **SHIP DYNAMICS**

A central element in engineering analysis and design is the determination of an appropriate mathematical model of the physical system under consideration. A model, in the context of a study of the dynamics and control of plant, may be defined as, “the characterization of the plant behaviour in terms of mathematical equations”. A model may be a very detailed representation or it may be only a very simple form indicating approximate behaviour or behaviour within a limited operating condition. The level of complexity appropriate for a particular case depends on the application.

As far as ship steering control systems are concerned, one possible approach involves equations of motion containing hydrodynamic derivatives which are themselves derived from a Taylor’s series expansion of the force/moment balance equations [Davidson, and Schiff, 1946; Abkowitz, 1964; Chislett and Strøm-Tejsen, 1965a, 1965b] . These complex models which are popular among naval architects can accurately predict the motion of a ship in a sea way. However, the determination of the large number of parameters in these models must be repeated for different ships and the techniques rely heavily on scale model testing which undoubtedly creates further problems with regard to scaling. An alternative approach is to refine the model into a form recognisable by control engineers [Nomoto et al, 1957; Norrbin, 1963; Bech and Smith, 1969; Van Leeuwen, 1972; Bech, 1972]. These models are usually of reduced complexity, however, they must be able to describe rather precisely the structure of the system to be controlled, e.g. the dynamic responses of the ship’s motions caused by rudder deflections.

The main purpose of this work is to develop intelligent controllers for the heading control of a ship. It is therefore appropriate to use the simple models preferred by control engineers, rather than the complex models, wherever possible. This Chapter reviews such models and explains the influence of various operating conditions on the parameters of these models. Both linear and non-linear models are reviewed. The layout of the Chapter is as follows: Section 2.1 provides a basic formulation of the ship steering equations. In Section 2.2 some well known linear models are reviewed. Section 2.3 discusses and presents methods for parameter normalization. The importance of normalization of ship parameters is also discussed in this Section. Two non-linear ship models are reviewed in Section 2.4. Full scale trials are briefly described in Section 2.5 and the influence of the forward speed of the ship, the depth of water and the loading on the ship parameters is discussed in Section 2.6.

## 2.1 Basic Equations of Motion

The equations describing motion of a ship are derived using Newton's laws of motion by taking two co-ordinate systems into account: an inertial system  $X_0O_0Y_0Z_0$  (Earth fixed) and a ship fixed system  $XOYZ$  as is indicated in Figure 2.1.

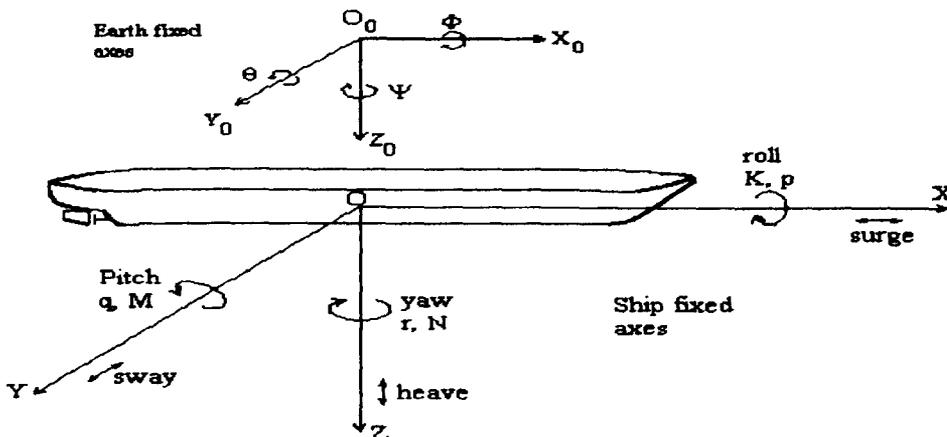


Figure 2.1: Ship fixed and Earth fixed axes

The equations of motion can be described completely by the general six degrees of freedom of rigid body motion, namely three translations (surge, sway and heave) and three rotations (roll, pitch and yaw). The physical definitions and the nomenclature associated with states and motions are shown in Figure 2.1 and summarized in Table 2.1

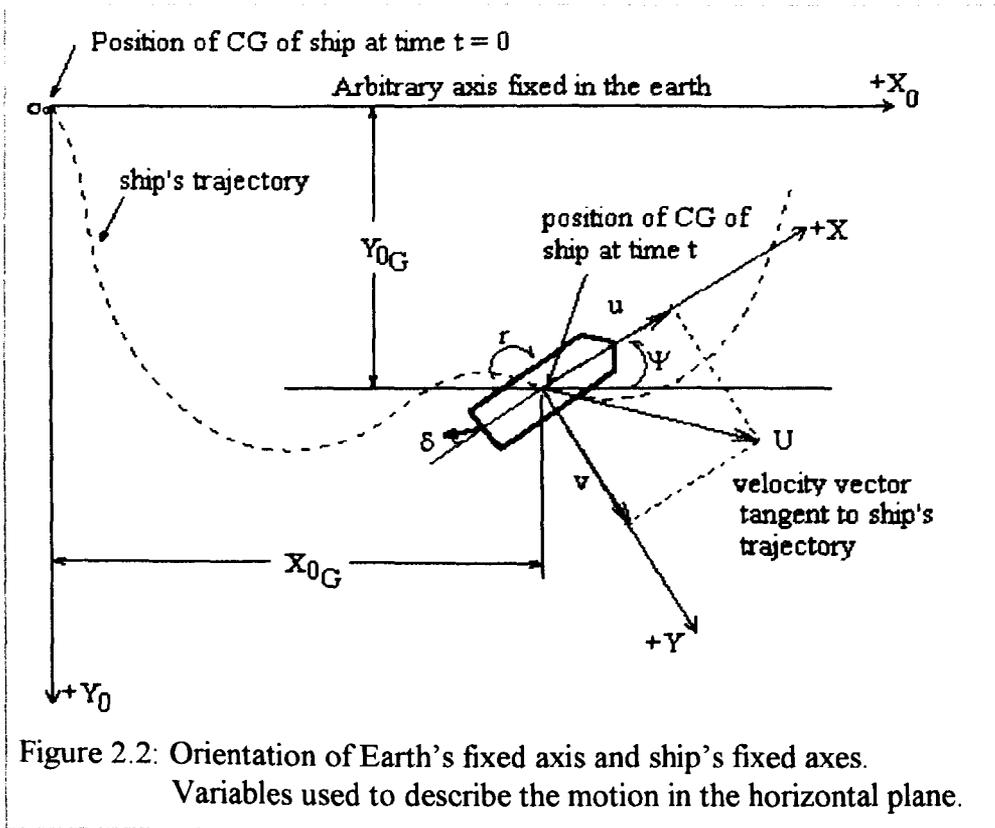
For surface ships, only horizontal motion is usually considered and the heave, roll and pitch modes are neglected. This reduces the number of degrees of freedom to three viz. surge translation in the x-direction, sway translation in the y-direction and the yaw rotation about the z-axis.

Throughout this thesis, only the horizontal motions surge, sway and yaw are considered. The remaining motions such as roll, pitch and heave are assumed to be negligible.

Table 2.1: Motions Nomenclature

axis	X	Y	Z
<b>Translation</b>	surge	sway	heave
position	x	y	z
velocity	u	v	w
force	X	Y	N
+ direction	forward	star board	downward
<b>Rotation</b>	roll	pitch	heave
angle	$\Phi$	$\theta$	$\Psi$
rate	p	q	r
torque	K	M	N
+ direction	star board	aft down	right turn

The variables used to describe the horizontal motion of a ship are sketched in Figure 2.2. The components of the total ship's velocity  $U$  on the  $X$  and  $Y$  axes are the surge velocity  $u$  and sway velocity  $v$  respectively.



By using the simplifications described above, the equations of motion can be derived as follows [Abkowitz, 1964; Crane et al., 1989; Fossen, 1994]:

$$\begin{aligned}
 m(\dot{u} - vr - x_G r^2) &= X \\
 m(\dot{v} + ur + x_G \dot{r}) &= Y \\
 I_z \dot{r} + m x_G (\dot{v} + ur) &= N
 \end{aligned}
 \tag{2.1}$$

where  $m$  is the mass of the ship,  $I_z$  is the moment of inertia about the  $z$ -axis and  $x_G$  is the  $x$  co-ordinate of the centre of gravity.  $X$  and  $Y$  are the hydrodynamic forces and  $N$  is the hydrodynamic moment. The main difficulty in modelling ship dynamics is to find suitable expressions for  $X$ ,  $Y$  and  $N$ . These are complicated functions of the ship motion relative to the sea.

Various functional forms of these have been suggested in the literature. For example, Abkowitz [1964] suggested the following functional form for X, Y and N:

$$\begin{aligned} X &= f(u, v, r, \delta, \dot{u}, \dot{v}, \dot{r}) \\ Y &= f(u, v, r, \delta, \dot{u}, \dot{v}, \dot{r}) \\ N &= f(u, v, r, \delta, \dot{u}, \dot{v}, \dot{r}) \end{aligned} \tag{2.2}$$

and approximated the functions with Taylor series expansions about the steady-state condition  $u = u_0$ ,  $v = r = \delta = \dot{u} = \dot{v} = \dot{r} = 0$ , where  $\delta$  is the rudder angle and  $u_0$  is the mean forward speed.

Other functional forms can be found in [Norrbin, 1970; Blanke, 1981 etc.]. The derivatives of X, Y and N are called *hydrodynamic derivatives*. These derivatives depend on many factors, among others on loading, trim and the depth of water. The hydrodynamic derivatives can be determined approximately from hydrodynamic theory [Comstock 1967; Norrbin 1970] or from experiments using scale models [Ström-Tejsen and Chislett 1966; Comstock 1967; Matora, 1972]. However, a more appropriate approach is to determine these derivatives from experiments on ships using system identification methods [Abkowitz, 1975, 1980; Åström and Källström 1976; Tiano, 1976; Källström, 1979; Åström 1980, Källström and Åström, 1981; Van Amerongen, 1982; Flobakk, 1983; Kaasen, 1986; Holzhüter, 1989].

## 2.2 Linear Models

In this Section, we review some well known linear ship models that have been proposed in the literature.

### 2.2.1 The Model of Davidson and Schiff

The first linear model of a ship steering system was put forward by Davidson and Schiff [1946]. This model can be derived by linearizing (2.1) around the stationary conditions  $u = u_0$ ,  $v = r = 0$  and is given by

$$\begin{aligned} m(\dot{v} + u_0 r + x_G \dot{r}) &= Y \\ I_z \dot{r} + m x_G (\dot{v} + u_0 r) &= N \end{aligned} \quad (2.3)$$

where  $u_0$  is assumed to be a constant.

Davidson and Schiff modelled the hydrodynamic force and moment as follows:

$$\begin{aligned} Y &= Y_v \dot{v} + Y_r \dot{r} + Y_v v + Y_r r + Y_\delta \delta \\ N &= N_v \dot{v} + N_r \dot{r} + N_v v + N_r r + N_\delta \delta \end{aligned} \quad (2.4)$$

where  $Y_v = \frac{\partial Y}{\partial v}$ ,  $N_v = \frac{\partial N}{\partial v}$  and so on for the other coefficients.

Combining (2.3) and (2.4) we can write

$$M\dot{x} = Nx + bu \quad (2.5)$$

where

$$\begin{aligned} M &= \begin{bmatrix} m - Y_v & m x_G - Y_r \\ m x_G - N_v & I_z - N_r \end{bmatrix}, \quad N = \begin{bmatrix} Y_v & Y_r - m u_0 \\ N_v & N_r - m x_G u_0 \end{bmatrix}, \quad b = \begin{bmatrix} Y_\delta \\ N_\delta \end{bmatrix}, \\ x &= \begin{bmatrix} v \\ r \end{bmatrix} \text{ and } u = \delta. \end{aligned}$$

In state space form,

$$\dot{x} = Ax + Bu \quad (2.6)$$

where

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}, \quad B = \begin{bmatrix} b_{11} \\ b_{21} \end{bmatrix}.$$

The coefficients are defined as

$$\begin{aligned} a_{11} &= \frac{(I_z - N_{\dot{r}})Y_v - (mx_G - Y_{\dot{r}})N_v}{|M|} \\ a_{12} &= \frac{(I_z - N_{\dot{r}})(Y_r - mu_0) - (mx_G - Y_{\dot{r}})(N_r - mx_G u_0)}{|M|} \\ a_{21} &= \frac{(m - Y_{\dot{v}})N_v - (mx_G - N_{\dot{v}})Y_v}{|M|} \\ a_{22} &= \frac{(m - Y_{\dot{v}})(N_r - mx_G u_0) - (mx_G - N_{\dot{v}})(Y_r - mu_0)}{|M|} \\ b_{11} &= \frac{(I_z - N_{\dot{r}})Y_{\delta} - (mx_G - Y_{\dot{r}})N_{\delta}}{|M|} \\ b_{21} &= \frac{(m - Y_{\dot{v}})N_{\delta} - (mx_G - N_{\dot{v}})Y_{\delta}}{|M|} \end{aligned} \quad (2.7)$$

where  $|M|$  is the determinant of  $M$ .

Since the ultimate aim of modelling is to identify the relationship between rudder angle  $\delta$  and the yaw angle  $\Psi$ , it is convenient to introduce  $\Psi$  as an additional variable in (2.6). The state-space representation now becomes

$$\dot{x}_1 = A_1 x_1 + B_1 u \quad (2.8)$$

where

$$A_1 = \begin{bmatrix} a_{11} & a_{12} & 0 \\ a_{21} & a_{22} & 0 \\ 0 & 1 & 0 \end{bmatrix}, \quad B_1 = \begin{bmatrix} B \\ 0 \end{bmatrix}, \quad \text{and } x_1 = [v \quad r \quad \Psi]^T$$

### 2.2.2 Nomoto's Models

Nomoto et al. [1957] have proposed two simple linear transfer functions based on the model of Davidson and Schiff. Nomoto's models have been used extensively by control engineers for analysis and design of ship autopilots. The major contributions include [Koyama, 1972; Nomoto, 1972; Källström, 1979; Ohtsu et al. 1979; Åström, 1980; Arie et al., 1986; Lu and Zhang, 1987; Katebi and Byrnes, 1988; Endo et al., 1989; Richter and Burns, 1993; Balasuriya and Hoole, 1995; Lauvdal and Fossen, 1995; Holzhüter and Schultze, 1996]. These models give a reasonably accurate description of the steering behaviour for a large class of ships [Van Amerongen, 1982].

The Nomoto's second order model can be derived from (2.8) as follows:

$$\frac{\Psi(s)}{\delta(s)} = C(SI - A_1)^{-1} B_1, \quad C = [0 \ 0 \ 1]$$

or

$$\frac{\dot{\Psi}(s)}{\delta(s)} = \frac{b_1 s + b_2}{(s^2 + a_1 s + a_2)} \quad (2.9)$$

where

$$\begin{aligned} a_1 &= -a_{11} - a_{22} \\ a_2 &= a_{11}a_{22} - a_{12}a_{21} \\ b_1 &= b_{21} \\ b_2 &= a_{21}b_{11} - a_{11}b_{21} \end{aligned} \quad (2.10)$$

The above transfer function is commonly represented as

$$\frac{\dot{\Psi}(s)}{\delta(s)} = \frac{K(T_3s+1)}{(T_1s+1)(T_2s+1)} \quad (2.11)$$

This is known as the Nomoto's second order model. In the time domain, the model can be expressed as

$$T_1T_2\ddot{\Psi} + (T_1 + T_2)\dot{\Psi} + \Psi = K(\delta + T_3\dot{\delta}) \quad (2.12)$$

The parameters  $T_1, T_2, T_3$  and  $K$  depend on the operating conditions and are usually referred to as *steering quality indices* [Nomoto et al. 1957; Clarke, 1987; Crane et al. 1989].  $T_1$  and  $T_2$  are generally positive, however,  $K$  and  $T_1$  can have positive or negative values.  $T_1$  is positive for course stable ships and it is negative for course unstable ships [Koyama, 1972; Crane et al. 1989; Fossen, 1994].

An approximation of (2.11) can be obtained by setting  $T = T_1 + T_2 - T_3$ :

$$\frac{\dot{\Psi}(s)}{\delta(s)} = \frac{K}{(Ts+1)} \quad (2.13)$$

This is known as the Nomoto's first order model. In the time domain:

$$T\dot{\Psi} + \Psi = K\delta \quad (2.14)$$

The parameters  $K, T_1, T_2, T_3$  and hence  $K$  and  $T$  can be computed using (2.6) through (2.11) if the exact values of the hydrodynamic derivatives are known. This means that the determination of  $K, T_1, T_2$  and  $T_3$  is equivalent to the determination of the model of Davidson and Schiff. Both of them define the governing equation of motion and consequently provide a complete

expression of steering quality of a ship. The former procedure has, however, the advantage that it does not require experimental procedures as lengthy as the latter [Nomoto, et al., 1957]. As the values of these parameters for the ships considered in this thesis are already available in the literature, the issues related to the computation/identification of these parameters will not be discussed. The values of these parameters for some of the ships are given in Table 2.2. The Table shows that the parameters  $T_1$ ,  $T_2$ ,  $T_3$  and  $K$  of the Nomoto's Second order model and the parameters  $T$  and  $K$  of the Nomoto's first order model are significantly different for different ships. Åström [1980] suggests that it is advantageous to re-write Nomoto's first-order model of (2.13) as follows:

$$\frac{\dot{\Psi}(s)}{\delta(s)} = \frac{b}{s + a} \quad (2.15)$$

where  $b = \frac{K}{T}$  and  $a = \frac{1}{T}$ .

The reason is that the parameter  $b$  varies less than  $K$ . For most of the ships, this parameter changes only by a factor of 3 over the operating conditions [Åström, 1980]. The parameters  $a$  and  $b$  are also given in Table 2.2. The gain  $b$  can be expressed approximately as follows:

$$b = c \frac{AL}{D} \quad (2.16)$$

where  $A$  is the rudder area in square meters,  $D$  is the displacement in cubic meters and  $c$  is a parameter whose value is approximately 0.5. [Åström and Wittenmark, 1995]. The parameter  $a$  depends on trim, forward speed and loading. Its sign may change with the operating conditions.

Table 2.2: Model parameters for different ships

Ship	Mine Sweeper	Merchant ship Series 60	Merchant ship MARINER	Tanker 190000dwt	Tanker 210000dwt
Length (m)	55	160	161	305	310
Speed U (m/sec)	4.0	7.8	7.7	8.2	4.1
Speed (knots)	7.8	15.2	15.0	15.9	8.0
K	0.365	-0.0536	-0.187	0.0914	-0.0105
T <sub>1</sub>	-112.3	56.61	119.4	-382.0	1058
T <sub>2</sub>	3.988	6.358	7.737	14.13	37.8
T <sub>3</sub>	8.938	14.56	18.61	34.96	84.68
T	-117.25	48.408	108.527	-402.83	1011.12
b	-0.003113	-0.0011	-0.001723	-0.000227	-1.038e-5
a	-0.008528	0.02066	0.00922	-0.00248	9.891e-4
Reference	Goclowski & Gelb [1966]	Zuidweg [1970]	Chislett & Strøm-Tejsen [1965b]	Van Berlekom & Goddard [1972]	Ekdahl & Henriksson [1970]

## 2.3 Normalization

It is customary to normalize models of ships by introducing dimension-free quantities. One commonly used system for normalization of ship models is the *prime system* of SNAME [1950]. In this system, the length unit is taken

as the length of the ship,  $L$ , the unit of time is  $L/U$ , and the unit of mass is  $\frac{1}{2}\rho L^3$  where  $\rho$  is the mass density of water.

Prime non-dimensional forces, moments, velocities and accelerations (see Table 2.3) are defined accordingly in the following way:

- Non-dimensional force  $= \frac{\text{Force}}{\frac{1}{2}\rho L^2 U^2}$
- Non-dimensional moment  $= \frac{\text{Moment}}{\frac{1}{2}L^3 U^2}$
- Non-dimensional angular velocity  $= \frac{\text{Angular velocity}}{U} L$
- Non-dimensional acceleration  $= \frac{\text{Acceleration}}{U^2} L$
- Non-dimensional angular acceleration  $= \frac{\text{Angular acceleration}}{U^2} L^2$

Table 2.3: Prime non-dimensionalized coefficients

$m' = m / \left( \frac{1}{2} \rho L^3 \right)$	$t' = t / \left( \frac{L}{U} \right)$	$x'_G = x_G / L$	$u' = u / U$
$v' = v / U$	$\dot{v}' = \dot{v} / \left( \frac{U^2}{L} \right)$	$r' = r / \left( \frac{U}{L} \right)$	$\dot{r}' = \dot{r} / \left( \frac{U^2}{L^2} \right)$
$Y'_v = Y_v / \left( \frac{1}{2} \rho L^3 \right)$	$Y'_{\dot{v}} = Y_{\dot{v}} / \left( \frac{1}{2} \rho L^3 \right)$	$Y'_r = Y_r / \left( \frac{1}{2} \rho L^3 U \right)$	$Y'_i = Y_i / \left( \frac{1}{2} \rho L^3 U \right)$
$N'_v = N_v / \left( \frac{1}{2} \rho L^3 U \right)$	$N'_{\dot{v}} = N_{\dot{v}} / \left( \frac{1}{2} \rho L^4 \right)$	$N'_r = N_r / \left( \frac{1}{2} \rho L^4 U \right)$	$N'_i = N_i / \left( \frac{1}{2} \rho L^5 \right)$
$Y'_{vv} = Y_{vv} / \left( \frac{1}{2} \rho L^2 \right)$	$Y'_{vr} = Y_{vr} / \left( \frac{1}{2} \rho L^3 \right)$	$N'_{vv} = N_{vv} / \left( \frac{1}{2} \rho L^3 \right)$	$N'_{vr} = N_{vr} / \left( \frac{1}{2} \rho L^4 \right)$
$Y'_\delta = Y_\delta / \left( \frac{1}{2} \rho L^2 U^2 \right)$	$N'_{\delta} = N_{\delta} / \left( \frac{1}{2} \rho L^3 U^2 \right)$	$I'_x = I_x / \left( \frac{1}{2} \rho L^5 \right)$	$I'_z = I_z / \left( \frac{1}{2} \rho L^5 \right)$

Another normalization system, the *bis system*, was proposed by Norrbin [1970]. In this system the length unit is  $L$ , the time unit is  $\sqrt{L/g}$ , where  $g$  is the acceleration of gravity, and the mass unit is  $m$ .

The non-dimensionalized variables used in the *bis* system are given in Table 2.4 where  $\nabla$  is the hull contour displacement and  $\mu = m/\rho\nabla$  is the body mass density ratio. The non-dimensionalized variables in the *bis* system are usually denoted by a double prime ( $\cdot$ )'. For example a non-dimensionalized variable  $m$  in this system will be represented as  $m''$ .

Table 2.4: Normalization variables used for the Bis system

Unit	Bis-system
Length	L
Mass	$\mu\rho\nabla$
Inertia moment	$\mu\rho\nabla L^2$
Time	$\sqrt{L/g}$
Reference area	$\mu \frac{2\nabla}{L}$
Position	L
Angle	1
Linear velocity	$\sqrt{Lg}$
Angular velocity	$\sqrt{\frac{g}{L}}$
Linear acceleration	g
Angular acceleration	$\frac{g}{L}$
Force	$\mu\rho g\nabla$
Moment	$\mu\rho g\nabla L$

The model of Davidson and Schiff in its normalization form (prime system) can be expressed as

$$M' \dot{x}' = N' x' + b' \delta' \quad (2.17)$$

where

$$M' = \begin{bmatrix} m' - Y'_{\dot{v}} & m' x'_G - Y'_{\dot{r}} \\ m' x'_G - N'_{\dot{v}} & I'_z - N'_{\dot{r}} \end{bmatrix}, N' = \begin{bmatrix} Y'_v & Y'_r - m' u'_0 \\ N'_v & N'_r - m' x'_G u'_0 \end{bmatrix},$$

$$b' = \begin{bmatrix} Y'_{\delta} \\ N'_{\delta} \end{bmatrix} \text{ and } x' = \begin{bmatrix} v' \\ r' \end{bmatrix}$$

It is also possible to obtain a model structure in terms of the actual state variables and the non-dimensionalized model parameters. For example, the Davidson and Schiff's model can also be expressed as:

$$M' \dot{x} = N' x + b' u \quad (2.18)$$

or

$$\begin{bmatrix} \frac{L}{U^2} (m' - Y'_{\dot{v}}) & \frac{L^2}{U^2} (m' x'_G - Y'_{\dot{v}}) \\ \frac{L}{U^2} (m' x'_G - N'_{\dot{v}}) & \frac{L^2}{U^2} (I'_z - N'_{\dot{r}}) \end{bmatrix} \begin{bmatrix} \dot{v} \\ \dot{r} \end{bmatrix} \\ = \begin{bmatrix} \frac{1}{U} Y'_v & \frac{L}{U} (Y'_r - m' u'_0) \\ \frac{1}{U} N'_v & \frac{L}{U} (N'_r - m' x'_G u'_0) \end{bmatrix} \begin{bmatrix} v \\ r \end{bmatrix} + \begin{bmatrix} Y'_{\delta} \\ N'_{\delta} \end{bmatrix} \delta$$

similarly Nomoto's second-order and first-order models can be re-written as:

Nomoto's second order model:

$$\left(\frac{L}{U}\right)^2 T_1 T_2 \ddot{\Psi} + \left(\frac{L}{U}\right) (T_1 + T_2) \dot{\Psi} + \Psi = \left(\frac{U}{L}\right) K' \delta + K' T_3 \dot{\delta} \quad (2.19)$$

where  $T'_j = T_j \left(\frac{U}{L}\right)$ ,  $j = 1, 2, 3$  and  $K' = \left(\frac{L}{U}\right) K$

Nomoto's first order model:

$$\left(\frac{L}{U}\right)T'\ddot{\Psi} + \dot{\Psi} = \left(\frac{U}{L}\right)K'\delta \quad (2.20)$$

where

$$K' = \left(\frac{L}{U}\right)K; \quad T' = \left(\frac{U}{L}\right)T \quad (2.21)$$

The main advantage of this type of representation is that the non-dimensional gain and time constant will typically be in the range:  $0.5 < K' < 2$  and  $0.5 < T' < 2$  for most ships [Van Amerongen, 1984, Fossen, 1994]. Crane et al. [1989] demonstrate that a ship with high responsiveness to rudder and yaw turning ability will have a small value of  $T'$  and a large value of  $K'$ . In other words, a large ratio  $\frac{K'}{T'}$  is indicative of good manoeuvrability.

Norrbin [1965] and Nomoto [1966] propose that the turning ability of a stable or marginally stable ship can be measured on the basis of the following *turning index*:

$$P = \frac{1}{2} \frac{K'}{T'} \quad (2.22)$$

Norrbin concludes that  $P > 0.3$  guarantees a reasonable standard of course change quality for most ships while  $P > 0.2$  is sufficient for large oil tankers. The above index is based on analysis of the zig-zag test (see Section 2.5.2).

## 2.4 Nonlinear Models

The Nomoto's models (also the model of Davidson and Schiff) are based on the assumption of mean constant forward speed and work well, in calm

seas, if the yaw rate and the sway velocity are small. If this is not the case, nonlinear models must be used. A number of nonlinear models have been proposed. Two of these nonlinear models are reviewed below:

#### 2.4.1 The Model of Bech and Smith

This model is an extended version of the Nomoto's second-order model and is described as follows:

Bech and Smith [1969] noticed that the parameters  $\frac{K}{T_1 T_2}$ ,  $\frac{T_1 + T_2}{T_1 T_2}$  and  $T_3$  of (2.12) are approximately constant for a given ship at constant speed, whereas  $\frac{1}{T_1 T_2}$  changes considerably. They argued that any non-linear effects could be included in this term. Hence they amended (2.12) as follows:

$$\ddot{\Psi} + \left( \frac{1}{T_1} + \frac{1}{T_2} \right) \dot{\Psi} + \frac{K}{T_1 T_2} H_B(\dot{\Psi}) = \frac{K}{T_1 T_2} (T_3 \dot{\delta} + \delta) \quad (2.23)$$

where  $H_B(\dot{\Psi})$  is a non-linear function of  $\dot{\Psi}$ , while the other coefficients are constant.  $H_B(\dot{\Psi})$  is determined from the relationship between  $\delta$  and  $\dot{\Psi}$  in the steady-state ( $\ddot{\Psi} = \dot{\Psi} = \dot{\delta} = 0$ ). This relation is determined from the results of the reverse spiral test for course unstable ships or the spiral test in the case of course stable ships (See Section 2.5.3).

Mathematically,  $H_B(\dot{\Psi})$  can be expressed as follows:

$$H_B(\dot{\Psi}) = \beta_3 \dot{\Psi}^3 + \beta_2 \dot{\Psi}^2 + \beta_1 \dot{\Psi} + \beta_0 \quad (2.24)$$

where  $\beta_i$ , ( $i = 0, 1, 2, 3$ ) are called Bech's coefficients.

For the ships with a symmetrical hull,  $\beta_2 \approx \beta_0 \approx 0$

$$\therefore H_B(\dot{\Psi}) = \beta_3 \dot{\Psi}^3 + \beta_1 \dot{\Psi} \quad (2.25)$$

where the parameter  $\beta_3$  is always positive in value, and  $\beta_1$  can have positive and negative values. When  $\beta_1$  is negative the ship is course unstable.

#### 2.4.2 Norrbin's Model

This model is similar to the model of Bech and Smith and can be obtained by substituting  $\Psi = H_N(\dot{\Psi})$  in (2.14):

$$T\ddot{\Psi} + H_N(\dot{\Psi}) = K\delta \quad (2.26)$$

where the non-linearity  $H_N(\dot{\Psi})$  is defined as [Norrbin, 1963]:

$$H_N(\dot{\Psi}) = \alpha_1 \dot{\Psi}^3 + \alpha_2 \dot{\Psi}^2 + \alpha_1 \dot{\Psi} + \alpha_0 \quad (2.27)$$

where  $\alpha_i$  ( $i = 0, 1, 2, 3$ ) are Norrbin's coefficients. These coefficients can be related to Bech's coefficients as follows: [Fossen and Paulsen, 1993; Fossen, 1994]

$$\alpha_i = \frac{\beta_i}{|\beta_1|}, \quad i = 0, 1, 2, 3. \quad (2.28)$$

For most ships,  $\alpha_2 = \alpha_0 = 0$ . Therefore, (2.27) reduces to

$$H_N(\dot{\Psi}) = \alpha_3 \dot{\Psi}^3 + \alpha_1 \dot{\Psi} \quad (2.29)$$

Now (2.26) becomes

$$\delta = m\ddot{\Psi} + d_1\dot{\Psi} + d_3\dot{\Psi}^3 \quad (2.30)$$

where  $m = \frac{T}{K}$ ,  $d_1 = \frac{\alpha_1}{K}$  and  $d_3 = \frac{\alpha_3}{K}$

The model of Bech and Smith and the Norrbin's model have been used extensively by many authors for the design of ship autopilots. These include [Bech, 1972; Honderd and Winkelman, 1972; Van Leeuwen, 1972; Van Amerongen, 1982; 1984; Fossen and Paulsen, 1993; Layne and Passino, 1993; Desanj et al., 1997].

## 2.5 Full Scale Trials:

To determine a ship's manoeuvring characteristics it is common practice to carry out a series of standard trials. The manoeuvring characteristics include the turning and course keeping ability, stability and performance of the ship. The ship parameters may also be estimated from these trials. The International Towing Tank Conference (ITTC) has proposed a number of standard ship manoeuvres, some of these are briefly described below.

### 2.5.1 Turning Circle Manoeuvre:

This is the most commonly used and probably the oldest test for models and ships. In this type of manoeuvre the ship makes a straight course

approach at a fixed speed. The rudder is then put over to a specific angle and held there whilst the ship completes a circle. It is proposed that the ship should be turned at a maximum speed with a rudder angle of minimum  $15^{\circ}$  to obtain the turning circle. A typical turning circle is shown in Figure 2.3. The following measurements are usually taken:

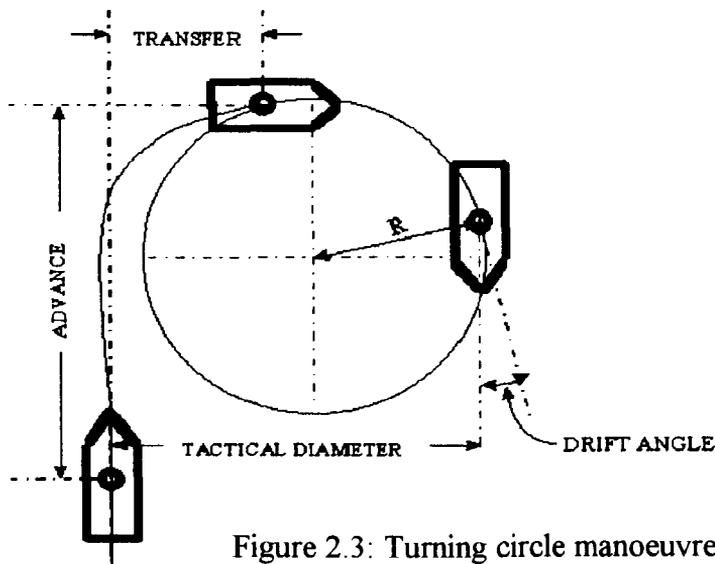


Figure 2.3: Turning circle manoeuvre

1. **Advance:** The distance the ship travels forward in the direction of the approach course from the instant the rudder is put over until the vessel has made a  $90^{\circ}$  change of heading.
2. **Transfer:** The distance normal to the original course in which the ship achieves a  $90^{\circ}$  change of course.
3. **Tactical Diameter:** It is the distance normal to the original course in which the ship achieves  $180^{\circ}$  change of course.
4. **Drift Angle:** It is the angle between the ship's head and the tangent to the path on the circle.
5. **Steady Turning Diameter:** The diameter of the steady circle path the ship takes up.

The steady turning radius  $R$  can be computed as follows:

$$R = K\delta_0 = K'\delta_0\left(\frac{U}{L}\right) \quad (2.31)$$

where  $\delta_0$  is the constant rudder angle.

The steady diameter  $D$  of the turning circle can be defined as [Crane et al., 1989]:

$$D = \frac{2U}{R} \quad (2.32)$$

substituting the value of  $R$  from (2.31) yields

$$D = \frac{2L}{K'\delta_0} \quad (2.33)$$

This equation shows that with a larger value of  $K'$  a smaller rudder angle may be used in achieving a given turning diameter.

### 2.5.2 Zig-Zag Manoeuvre:

The zig-zag manoeuvre, also called a Kempf manoeuvre after G. Kempf [1932], essentially determines the response of the ship to reversal of rudder. This test can be used to compare the manoeuvring properties and control characteristics of a ship with those of other ships. A typical zig-zag manoeuvre would be as follows (Figure 2.4):

1. Set a certain rudder angle (e.g.  $20^\circ$ ) to starboard.

2. When ship's course deviation has reached this angle to starboard, reverse the rudder to the same angle to port.
3. When the course deviation has reached the same angle to port, reverse the rudder again to the same angle to starboard and so on.

Important parameters of this manoeuvre are:

- (i) the time between successive rudder movements.
- (ii) the overshoot angle which is the amount by which the ship's heading exceeds the  $20^\circ$  (say) deviation before reducing.

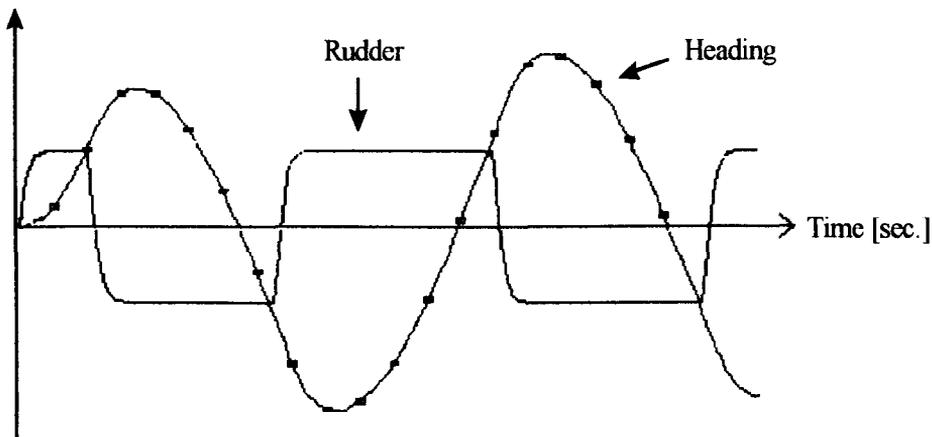


Figure 2.4: Zig-Zag Manoeuvre

Nomoto et al. [1957] determined the parameters  $K$  and  $T$  of equation (2.13) by using this manoeuvre. Their method is reviewed by many authors including [Clarke, 1987].

### 2.5.3 Spiral Manoeuvre:

This manoeuvre was first proposed by J. Dieudonné [1953]. This test produces a characteristic which indicates whether the ship is directionally stable or unstable. This manoeuvre consists of starting the vessel in a  $10^\circ$  -

$15^{\circ}$  (say) rudder angle turn and then altering the rudder in steps back through amidships to the same angle on the opposite helm. This procedure is then repeated in the opposite direction back to the original helm. As each rudder step is made the angle is held while the rate of turn steadies, and this rate of turn is measured and plotted against the corresponding rudder angle. For a stable ship, the  $r - \delta$  characteristic is a single valued function passing through or near the origin (Figure 2.5(a)). For a directionally unstable ship, the spiral manoeuvre yields a hysteresis type curve, as shown in Figure 2.5(b). To eliminate this discontinuity, Bech [1966,1968] suggested a modified manoeuvring trial whereby the rudder position is adjusted in order to maintain a constant rate of turn and hence produce a reverse spiral curve (Figure 2.5(c)). This characteristic is now a continuous function which equates the non-linearity  $H_B(\dot{\Psi})$  in (2.24).

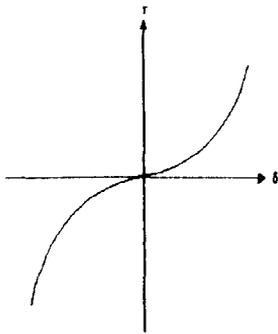


Figure 2.5 (a): Course stable ship

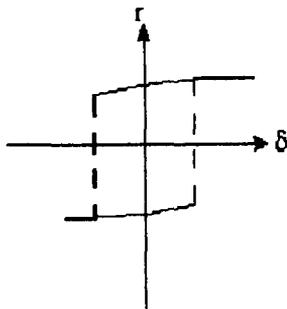


Figure 2.5 (b): course unstable ship.

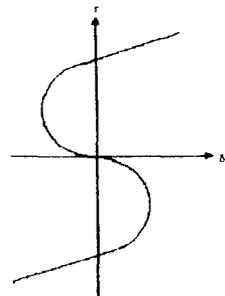


Figure 2.5 (c): Bech's reverse spiral curve

#### 2.5.4 The Pull - Out Manoeuvre:

This manoeuvre is also used to determine the directional stability of a ship [Burcher, 1972]. A rudder angle ( $\approx 20^{\circ}$ ) is put over and returned to amidships after steady turning has been attained. Both a port and starboard turn are performed and the ship's turn is measured/noted in each case. For a directionally stable ship, the rate of turn will be reduced to zero (Figure 2.6

(a)) and the ship will attain a new straight path. On the other hand, a residual rate of turn will persist if the ship is unstable, as shown in Figure 2.6 (b).

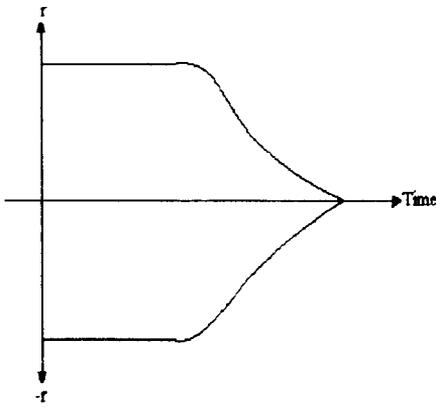


Figure 2.6 (a): Pull out manoeuvre for a stable ship

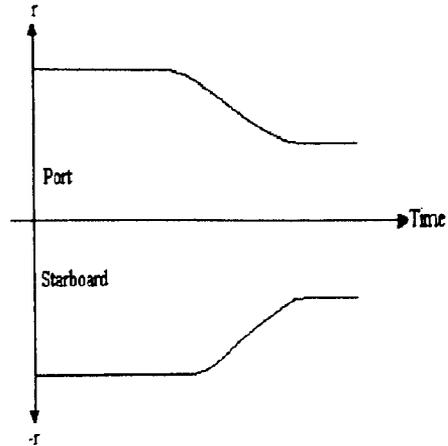


Figure 2.6 (b): Pull-out manoeuvre for an unstable ship.

The important feature of this test is that it is amenable to analysis to determine a measure of stability. Thus stable ships can be compared as to the degree of stability they possess. If the time history of rate of turn is plotted as the log of rate against time, it is found that after the initial transient the plot of a stable ship becomes a straight line, as shown in Figure 2.7. The slope of this plot can be used as a measure of stability, the steepest the slope the greater the degree of stability.

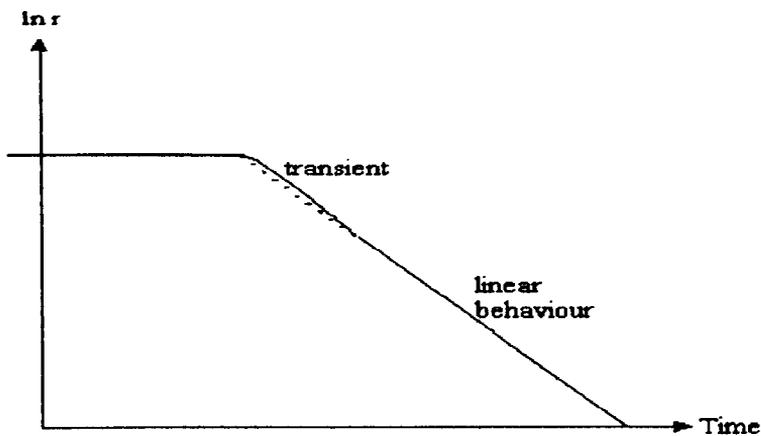


Figure 2.7: Logarithmic presentation of the pull-out manoeuvre

## 2.6 Parameter variations:

As we have already mentioned, the parameters of the ship models change with various operating conditions such as the forward speed of the ship, the depth of water and loading conditions etc. In this Section we shall discuss in some detail the influence of these operating conditions on parameters of the ship models.

### 2.6.1 Forward Speed Effects:

The ship parameters change significantly with the forward speed of the ship. One possible way to remove the influence of the forward speed on the model parameters is to use non-dimensional (normalized) quantities as discussed in Section 2.3. Particularly the relationship (2.21) is very important that clearly explains how the parameters  $K$  and  $T$  can be made dimensionless.

An alternative way to remove the influence of the forward speed  $U$  is described below [Fossen, 1994]:

Let  $K_0$  be the gain constant and  $T_0$  be the time constant of the Nomoto's first order model (equation (2.13)) at a nominal speed  $U_0$  m/sec. Using (2.21), we can obtain

$$K' = \left( \frac{L}{U_0} \right) K_0; \quad T' = \left( \frac{U_0}{L} \right) T_0 \quad (2.31)$$

Eliminating  $K'$  and  $T'$  from these expressions yields

$$\mathbf{K} = \left( \frac{U}{U_0} \right) \mathbf{K}_0; \mathbf{T} = \left( \frac{U_0}{U} \right) \mathbf{T}_0 \quad (2.32)$$

The parameters  $T_0$  and  $K_0$  can be computed from a standard manoeuvring test, e.g. from the results of the turning circle or zig-zag manoeuvres.

Similarly, the influence of the forward speed can be removed from the parameters of the model (2.11) and (2.15) as follows:

$$\begin{aligned} T_1 &= T_{10} \left( \frac{U_0}{U} \right) \\ T_2 &= T_{20} \left( \frac{U_0}{U} \right) \\ T_3 &= T_{30} \left( \frac{U_0}{U} \right) \\ a &= a_0 \left( \frac{U}{U_0} \right) \\ b &= b_0 \left( \frac{U}{U_0} \right)^2 \end{aligned} \quad (2.33)$$

where  $T_{i0}$  ( $i=1,2,3$ ),  $a_0$  and  $b_0$  are the values of the corresponding parameters at speed  $U_0$ .

### 2.6.2 Effects of the depth of water:

The ship parameters also change significantly with the depth of water. Fujino [1968] has studied the effect of the water depth on scale models of both a MARINER class ship and the tanker TOKYO MARU at various depth to draft ( $H/DT$ ) ratios. The parameters of these ships at a speed of 7

knots (1 knot = 1852 m/hour  $\approx$  0.5144 m/s) under various conditions of water depth are given in Tables 2.5 and 2.6 respectively. The open loop eigenvalues  $\lambda_1, \lambda_2$  are also given in the Tables. The third eigenvalue is always at the origin of the s-plane.

Table 2.5 shows that the MARINER vessel is stable under all conditions of H/DT. There is an initial decrease in stability at H/DT = 2.5 and then an increased stability in very shallow water (H/DT = 1.5). The two eigen values converge at approximately H/DT = 1.4 and then become a complex conjugate pair for lowest values of H/DT.

The tanker TOKYO MARU is less course stable overall than the MARINER. This can be verified from Table 2.6. The tanker becomes unstable for the intermediate range of water depth to draft ratios around H/DT = 2.50 and H/DT = 1.89.

Table 2.5: Parameters and eigenvalues of the MARINER ship at 7 knots at various depth to draft (H/DT) ratios

H/DT	1.21	1.50	1.93	2.50	$\infty$
<b>Parameters</b>					
$T_1'$	0.231+j0.153	1.072	2.678	3.244	2.749
$T_2'$	0.231-j0.153	0.362	0.356	0.382	0.367
$T_3'$	0.227	0.497	0.585	0.778	0.729
$T'$	0.234	0.957	2.449	2.848	2.386
$K'$	-0.326	-1.086	-2.377	-2.549	-2.026
$\lambda_1$	-3.009+j1.993	-0.9328	-0.3734	-0.3083	-0.3638
$\lambda_2$	-3.009-j1.993	-2.7624	-2.8090	-2.6178	-2.7248

Table 2.6: The parameters of the tanker TOKYO MARU at 7 knots at different H/DT ratios

H/DT	1.23	1.50	1.89	2.50	$\infty$
Parameters					
$T_1'$	1.279	15.630	0.405	0.415	12.680
$T_2'$	0.403	0.376	-9.986	-18.880	0.420
$T_3'$	0.373	0.576	0.664	0.804	0.893
$T'$	1.299	15.43	-10.25	-19.23	12.21
$K'$	-0.773	-8.511	5.141	9.140	-5.815
$\lambda_1$	-0.787	-0.064	+0.1	+0.053	-0.079
$\lambda_2$	-2.481	-2.660	-2.469	-2.140	-2.381

These results show that the various depths of water can change the parameters of ships and a course stable ship can become unstable at certain depths of water.

### 2.7.3 Effects of different loading conditions:

Apart from variations with speed and the depth of water, the parameters of ship models also change considerably with trim and loading. Åström [1980] has computed the parameters of the Nomoto's second order model of (2.11) for a large tanker at a constant speed of 8 m/sec under four different loading conditions. These parameters are given in Table 2.7. It can be seen that there is a significant variation in parameters at different loading conditions. It can also be noted that the ship is stable in operating conditions OC1, OC2 and OC3 but it is unstable in condition OC4. The condition OC1 corresponds to ballast and OC3 and OC4 to full load. In OC4 the tanker has a forward trim.

Table 2.7: Parameters of a tanker at different loading conditions.

Operating condition	T <sub>1</sub>	T <sub>2</sub>	T <sub>3</sub>	K	T
OC1	80	15	40	-0.013	50
OC2	160	20	30	-0.040	150
OC3	1000	25	60	-0.130	1000
OC4	-300	30	65	0.040	-400

## Chapter 3

### AUTOMATIC STEERING OF SHIPS

Automatic steering of ships has its origin near the beginning of this century, following the invention of gyrocompass. Sperry discussed the problem of automatic steering in 1922 [Sperry, 1922] and in the same year Minorsky presented the basic theory for directional stability of automatically steered ships. Minorsky [Minorsky, 1922] begins his historical and pioneering paper of 1922 with the following words:

*The problem of directional stability of automatically steered ships is gradually becoming of increasingly greater importance for various reasons. The possibility of obtaining more accurate steering by automatic means than can be accomplished by manual control with its inherent limitations due to the low sensitiveness of the human eye in detecting slow angular motions, fatigue etc., becomes of great importance with the increase in size of ships and cost of fuel. For merchant ships, an accurate and reliable automatic steering device becomes a real money saving proposition, largely justifying its use.*

Amazingly and interestingly, these 76 years old words are still true and the design of a good and reliable ship autopilot is still a hot topic of research. This Chapter throws some light on the development of ship autopilots during these 76 years. Various methods of designing ship autopilots are reviewed with illustrative examples. The Chapter is organized as follows. A brief introduction of a ship steering control system is presented in Section 3.1. Reference models are discussed in Section 3.2. A simplified model of steering machine is presented in Section 3.3. A ship autopilot is defined in Section 3.4. Section 3.5 reviews the early proportional autopilots. In Section 3.6 a number of methods of designing PD/PID controllers for the application are described.

Section 3.7 covers two non-linear autopilot design methods - the feedback linearization and the sliding mode control. Section 3.8 provides an overview of adaptive autopilots. In Section 3.9, intelligent autopilots have been reviewed briefly.

### 3.1 Ship Steering Control

Generally speaking, a ship steering control system is a single input single output (SISO) control system, as shown in Figure 3.1, where  $\Psi_r$  is the reference heading,  $\Psi_d$  is the desired heading,  $\Psi$  is the actual heading and  $\delta_c$  is the commanded rudder angle (all in degrees).

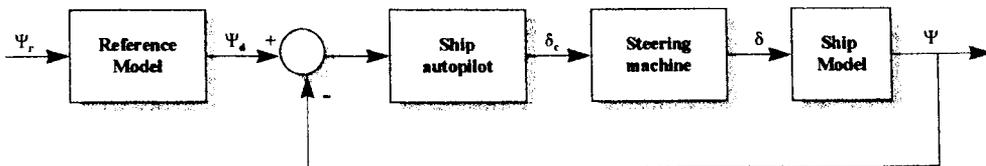


Figure 3.1: Ship Steering Control System

We have already reviewed a number of ship models in the previous Chapter, in this Chapter we shall describe the remaining blocks of the control system (Figure 3.1) with particular emphasis on the controller design.

### 3.2 Reference Model:

A reference model may be regarded as a prefilter which ensures that the numerical difficulties associated with large step input are avoided [Fossen, 1994]. The dynamics of the reference model should be matched to the dynamics of the ship regardless of the magnitude of the demanded change of

reference yaw angle. A reference model which is too sluggish can not produce an optimal performance since the ship can not reach the required heading in the minimum time. On the other hand we should not use a reference model which is too fast compared with the ship response characteristics because this may cause rudder actuator saturation and performance degradation.

Generally a second order reference model is used. Such a model can be described mathematically as follows:

$$\frac{\Psi_d}{\Psi_r} = \frac{K_m}{T_m s^2 + s + K_m} \quad (3.1)$$

where  $T_m$  and  $K_m$  are the design parameters that describe the closed loop behaviour of the system. For most practical implementations, these parameters are chosen as [Van Amerongen, 1982; Fossen, 1994]:

$$T_m \leq \frac{1}{2} \frac{LT}{U} \text{ and } K_m = \frac{1}{4\zeta^2 T_m} \quad (3.2)$$

where  $\zeta$  is the damping ratio of the closed loop system which is typically chosen in the interval  $0.8 \leq \zeta \leq 1$ .

Comparing (3.1) with the general second order system:

$$\frac{\Psi_d}{\Psi_r} = \frac{w_n^2}{s^2 + 2\zeta w_n s + w_n^2} \quad (3.3)$$

we can express  $T_m$  and  $K_m$  in terms of the damping ratio  $\zeta$  and the undamped natural frequency  $w_n$  of the closed loop system as follows:

$$T_m = \frac{1}{2\zeta w_n} \text{ and } K_m = w_n^2 T_m \quad (3.4)$$

A second order model may not be sufficient to generate smooth accelerations. In such situations, a third order model of the following form can be used:

$$\frac{\Psi_d}{\Psi_r} = \frac{c_m}{s^3 + a_m s^2 + b_m s + c_m} \quad (3.5)$$

where  $a_m$ ,  $b_m$  and  $c_m$  are constants. The method of computing these parameters can be found elsewhere [Simensen and Murray-Smith, 1995].

Some other higher order reference models are also available in the literature [Fossen, 1994].

### 3.3 Steering Machine:

The function of a steering machine is to move the rudder angle to a desired heading when demanded by the control system or by the helmsmen. A simplified model of the steering machine is shown in Figure 3.2. This model of the steering machine was proposed by Van Amerongen [1982].

Generally, the rudder limiter and rudder rate limiters in Figure 3.2 are typically in the ranges:

$$\delta_{\max} = \pm 35^\circ, \text{ and } \dot{\delta}_{\max} = \pm 2 \text{ to } \pm 7^\circ/\text{s}. \quad (3.6)$$

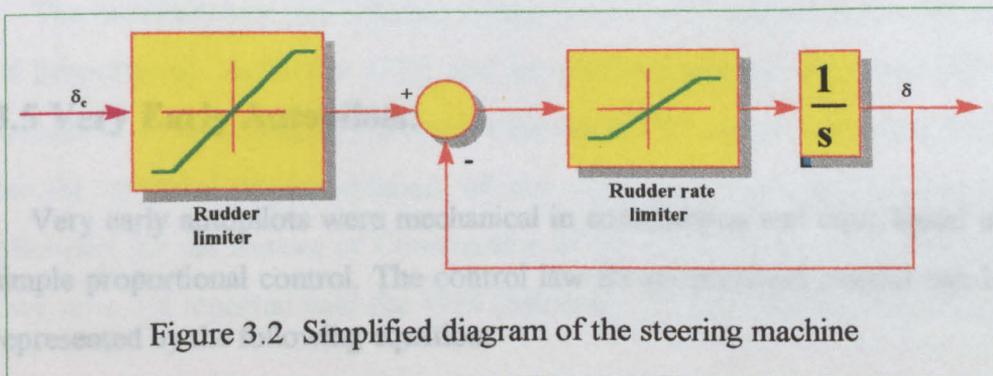


Figure 3.2 Simplified diagram of the steering machine

### **3.4 Ship Autopilot:**

An autopilot is a ship's steering controller, which automatically manipulates the rudder to decrease the error between the reference heading angle and the actual heading angle. Ship autopilots can be designed to perform two entirely different functions: course keeping and course changing. In course keeping the ship should stay on a set course, whereas in course changing, the control system should provide good manoeuvrability. An autopilot that can be used to make course changes is especially needed for large tankers, as they are difficult to handle because of their large size. Particular consideration is therefore given to the course changing problem in this thesis.

The development of ship autopilots has undergone the following stages during the past 76 years:

- Very early autopilots (proportional controllers)
- PD/PID (linear) autopilots
- Nonlinear autopilots
- Adaptive autopilots
- Intelligent autopilots

These developments are reviewed in the following sections:

### **3.5 Very Early Autopilots:**

Very early autopilots were mechanical in construction and were based on simple proportional control. The control law for proportional control can be represented by the following equation:

$$\delta = K_p (\Psi_d - \Psi) \quad (3.7)$$

where  $K_p$  is a proportional constant.

In practical terms the measured heading signal from the gyrocompass is compared to the desired heading and the heading error is input to the controller. The controller output then drives the rudder servo. Such controllers were popular until the 1950s.

Proportional control could only be satisfactory for the control of small ships but could not be adequate for the steering of large tankers. This type of control would cause the vessel to continue to oscillate either side of the required course; the steering gear would be constantly hunting to keep the ship on the correct mean course. The vessel would eventually reach its destination but expensive wear in the rudder gears and abnormal high fuel consumption restricted their use as automatic control devices. Early autopilots had other problems as well. The hydraulic telemotor unit, a device used to control the movement of the rudder, was reported to malfunction because of leakage [Tetley and Calcutt, 1991]. An electrical system now replaces this inefficient device.

### **3.6 PD/PID Autopilots:**

The unsatisfactory performance of the proportional control led to the use of proportional- derivative (PD) and proportional-integral-derivative (PID) autopilots. In fact, Minorsky proposed the use of PID autopilot in 1922. Tests on the practical implementation of the controller were carried out by Minorsky for the Bureau of Construction of the US Navy in 1923, although they were not reported until the 1930 [Minorsky, 1930]. Despite considerable

success of the control system, the operating personnel at sea were very definitely and strenuously opposed to automatic steering, the US Navy withdrew its support and Minorsky was forced to sell his patents rights to the Bendix Corporation [Bennet, 1993]. In the meantime, Sperry's proportional controller got considerable popularity and in the 1930s, over four hundred of Sperry's autopilots had been installed on merchant ships throughout the world [Burns, 1990]. After the introduction of the Ziegler-Nichols rules [Ziegler and Nichols, 1942, 1943] in the early 40s, the popularity of PID controllers grew exponentially and found enormous applications including ship steering control. Probably the first commercial use of PD control was mentioned in 1951 by Luke and West [Luke and West, 1960]. They showed that the correct combination of both proportional and derivative terms could produce reasonable control of the vessel. From the 1950s to the 1980s, almost all ship autopilots were based on PD/PID controller. Even today, most commercial ships use this type of autopilot.

A PID controller can be represented by the following equation:

$$u(t) = K_p e(t) + K_d \dot{e}(t) + K_i \int_0^t e(\tau) d\tau \quad (3.8)$$

where  $K_p$ ,  $K_d$  and  $K_i$  are the proportional, derivative and integral gains respectively.  $u(t)$  is the controller output and  $e(t)$  is the error signal. Another equivalent form of the controller is given by

$$u(t) = K_p \left[ e(t) + \frac{1}{T_i} \int_0^t e(\tau) d\tau + T_d \frac{de(t)}{dt} \right] \quad (3.9)$$

where  $T_i = \frac{K_p}{K_i}$  and  $T_d = \frac{K_d}{K_p}$  are known as the integral and derivative time

constants respectively. Many methods have been developed to compute the parameters  $K_p$ ,  $K_d$  and  $K_i$  (or  $K_p$ ,  $T_i$  and  $T_d$ ) of the controller. A quick review

of these methods can be found in [Åström et al., 1993; Åström and Hägglund, 1995; Unar, 1995; Unar et al., 1996 etc.].

In the particular case of ship steering control system, the PID controller is usually implemented in the following form [Crane et al. 1989; Fossen, 1994]:

$$\delta = K_p(\Psi_d - \Psi) - K_d\dot{\Psi} + K_i \int_0^t (\Psi_d - \Psi) d\tau \quad (3.10)$$

In the following Sections, we shall review some methods of designing PD/PID controllers for a ship steering control system.

### 3.6.1 PID parameters in terms of the damping ratio and the undamped natural frequency:

Fossen [1994] has derived simple relations for computing the PD/PID controller parameters in terms of the damping ratio  $\zeta$  and the natural frequency  $\omega_n$  of a closed loop ship control system. His method is described below:

For the sake of simplicity, we shall first design a PD controller and then we shall include the integral gain.

A PD controller can be expressed as follows:

$$\delta = K_p(\Psi_d - \Psi) - K_d\dot{\Psi} \quad (3.11)$$

If we represent the ship dynamics by Nomoto's first order model of (2.14) then the corresponding characteristic function of the closed loop system can be obtained as

$$T\ddot{\Psi} + (1 + KK_d)\dot{\Psi} + KK_p\Psi = KK_p\Psi_d \quad (3.12)$$

The damping ratio and the natural frequency of the system can therefore be obtained as

$$\zeta = \frac{1 + KK_d}{2\sqrt{TKK_p}}; \quad \text{and} \quad \omega_n = \sqrt{\frac{KK_p}{T}}$$

which suggest that the parameters  $K_p$  and  $K_d$  can be computed as:

$$K_p = \frac{T\omega_n^2}{K}; \quad K_d = \frac{2T\zeta\omega_n - 1}{K} \quad (3.13)$$

Fossen [1994] suggests the following rule of thumb for the integral action:

$$\frac{K_i}{K_p} \approx \frac{\omega_n}{10} \quad (3.14)$$

Therefore,

$$K_i = \frac{\omega_n K_p}{10} = \frac{\omega_n^3 T}{10K} \quad (3.15)$$

### Example 3.1

Consider a fully loaded unstable oil tanker of length 350m. The parameters of the tanker at a speed of 8.1 m/sec are [Dyne and Trägårdh, 1975] :  $K = -0.019$  per second and  $T = -153.7$  seconds. The rudder limiter and rudder rate limiter are chosen as  $\pm 30^\circ$  and  $\pm 2.33^\circ/\text{s}$  respectively. A second order reference model of equation (3.3) is used to generate the desired states. If the desired damping ratio is 1 and the natural frequency is 0.03 radians per second, then the PID controller parameters ( $K_p$ ,  $K_d$  and  $K_i$ ) can be computed from (3.13) and (3.15) as:

$K_p = 7.2805$ ,  $K_d = 538$ ,  $K_i = 0.0218$ .

The performance of the controlled system for a reference heading change of  $20^\circ$  is shown in Figure 3.3.

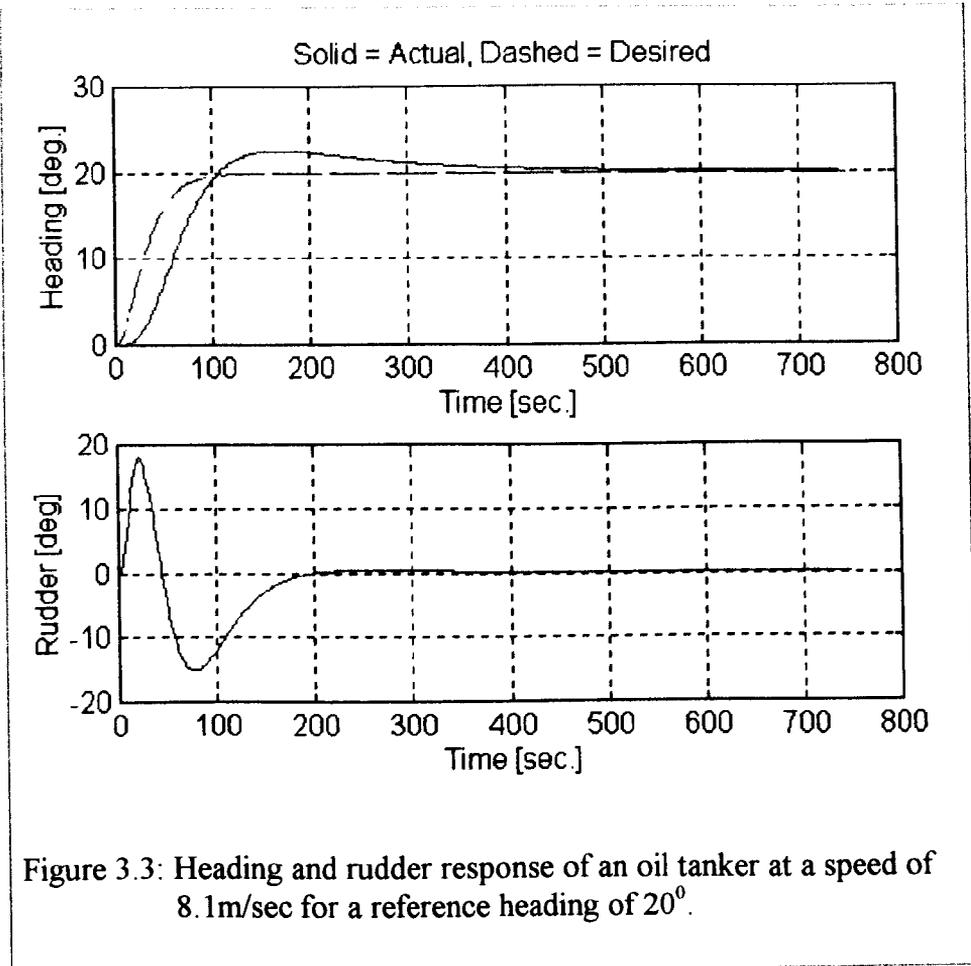


Figure 3.3: Heading and rudder response of an oil tanker at a speed of 8.1m/sec for a reference heading of  $20^\circ$ .

The Figure shows that the ship does not follow the desired response during turning. The performance of the controller for a reference heading change of  $50^\circ$  at the same speed is illustrated in Figure 3.4. An overshoot of about  $18^\circ$  can clearly be observed. This can be dangerous, if other ships are in the surroundings or if the ship is sailing in a narrow channel. Moreover, the rudder reaches its saturation limits whilst the ship is turning. The performance of the controller is therefore not robust, even at a fixed forward speed of the ship.

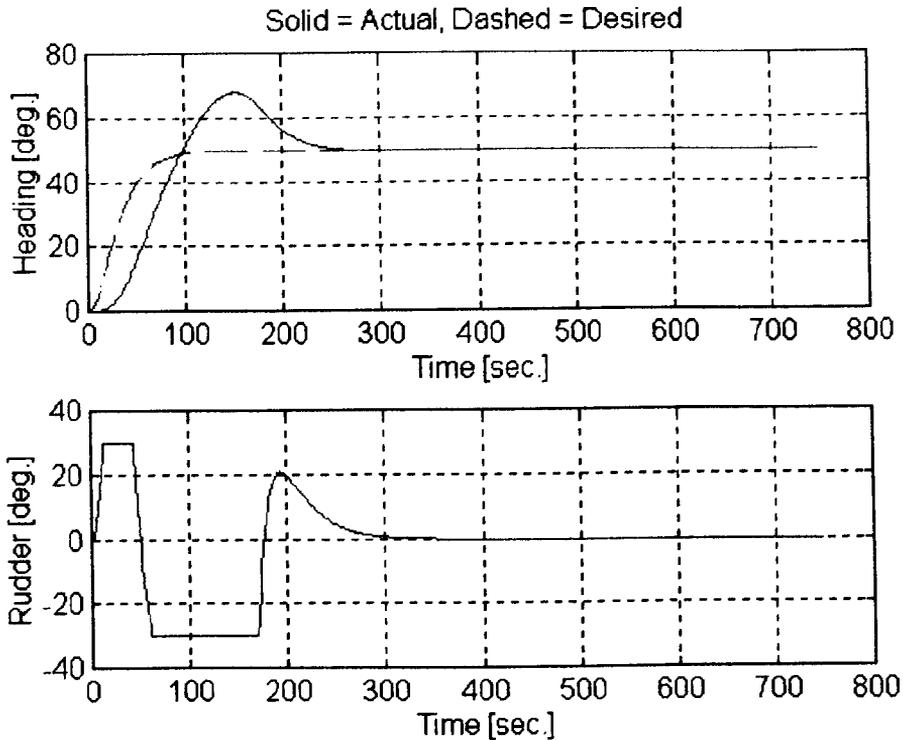


Figure 3.4: Heading and rudder response of oil tanker at a speed of 8.1m/sec for a reference heading of  $50^{\circ}$ .

### 3.6.2 Internal Model Control Laws:

Internal model control (IMC) is one of the powerful methods that has received considerable attention during the past 15 years. A brief introduction to this approach is given in appendix A. This approach has found many successful applications particularly in the field of chemical process control. However, this technique has been considered for a ship steering control system only by this author [Unar and Murray-Smith, 1997a]. This approach can be used to derive PD/PID controller settings for a wide variety of process models [Rivera et al., 1986; Morari and Zafiriou, 1989]. For example, for the

Nomoto's first order model of equation (2.13), the PD parameters can be calculated as

$$\begin{aligned} K_p &= \frac{1}{K\Omega} \\ K_d &= T \end{aligned} \quad (3.16)$$

The integral action may be included by using the rule of thumb of equation (3.15).

From the above equation, we see that the controller parameter  $K_p$  is inversely proportional to the gain constant and the parameter  $K_d$  is equal to the time constant of the ship. When the speed of the ship changes, both  $T$  and  $K$  change and hence the controller parameters change. The main advantage of the method is that there is only one parameter  $\Omega$  to be specified. This parameter is known as the IMC tuning constant.

### Example 3.2

In this example, we consider the model of ROV Zeefakkel. It is a ferry of length 45 m. The parameters of the ferry at a speed of 5 m/sec. are  $K = 0.5$  per second and  $T = 31$  sec [Van Amerongen, 1982]. Using equation (3.16) the PD parameters can be computed as:  $K_p = 57.1429$  and  $K_d = 31$  for  $\Omega = 0.035$ . The desired heading response is represented by the third order model of equation (3.5) with  $a_m = 0.9341$ ,  $b_m = 0.2040$ , and  $c_m = 0.0182$  [Unar and Murray-Smith, 1997a]. The maximum rudder limit is chosen as  $\pm 35^\circ$  and the maximum rudder rate is  $\pm 7^\circ/\text{sec}$  [Fossen and Paulsen, 1992, 1993]. The performance of the controller at a speed of 7m/sec for a reference heading change of  $10^\circ$  and  $50^\circ$  is illustrated in Figure 3.5. As can be seen, the performance is quite satisfactory. The controller will also perform well at other speeds, as the controller parameters are directly related to the ship parameters  $T$  and  $K$  which vary with the forward speed of the ship. However,

it should be noted that the internal model controllers usually perform well only for open loop stable systems. The performance can not be guaranteed for open loop unstable ships.

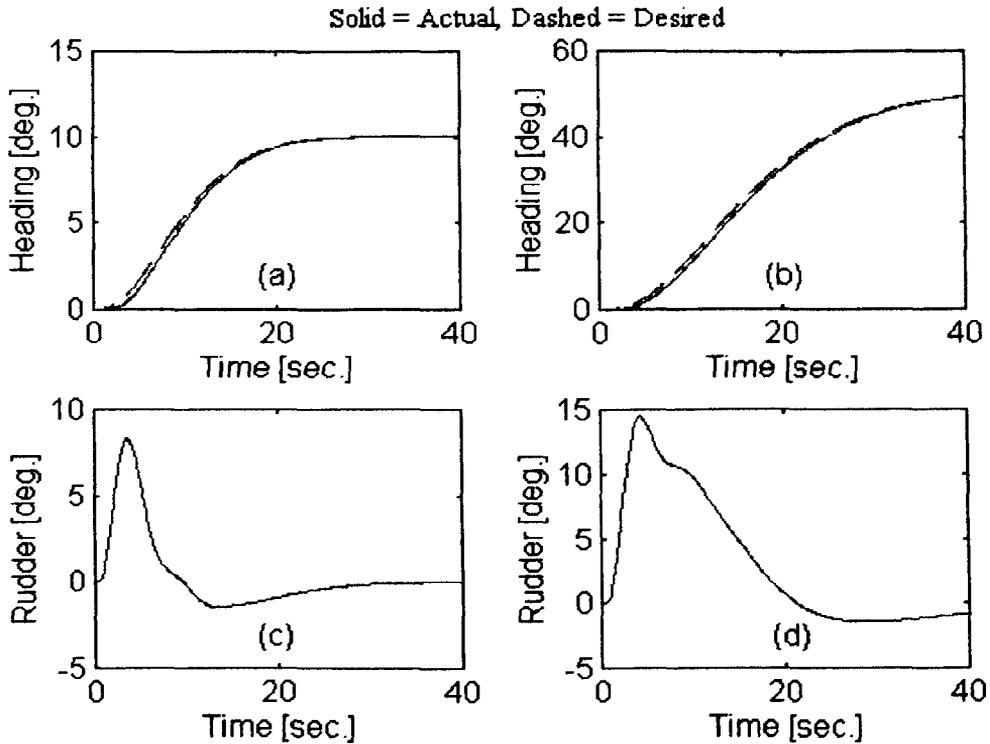


Figure 3.5: Performance of the PD controller tuned using the IMC laws  
 (a) Heading at a reference input of  $10^0$  (c) the corresponding control signal  
 (b) Heading at a reference input of  $50^0$  (d) the corresponding control signal

### 3.6.3 Controller Design using Optimal Control Theory

The optimising of steering requires first the consideration of a suitable criterion for good steering. The definition of a suitable criterion is not straightforward as there are many requirements on an autopilot for ship steering. It should guarantee stability and manoeuvrability over a wide operating range and it should attempt to minimize the propulsion loss induced by steering. A simple loss function can be formulated from the following argument. When a ship is underway, the action of the external influences -

such as wind and waves - gives rise to a continuous yawing motion to either side of the desired course. This motion increases the drag of the ship, which in turn reduces its speed and also causes the ship to follow a longer sinusoidal path, which further reduces its down track speed. The action of the autopilot in trying to correct the heading deviations gives rise to rudder movements, which also cause an increase in drag.

Nomoto and Motoyama [1966] were the first to point out that the magnitude of these drag forces may be sufficient to cause significant speed reductions in certain cases. Koyama [1967] later showed that, although these effects could not be totally eliminated, they could be minimized by the correct selection of the proportional and derivative gain controls on the autopilot in a particular sea way. Matora [1967] and Matora and Koyama [1968] considered the use of a performance index for the measurement of propulsion losses. This expression consisted of two terms. One term was attributed to the excess distance covered by the yawing of the vessel; the second term was linked to the rudder resistance. The performance index is normally expressed as:

$$J = \frac{1}{T} \int_0^T (\tilde{\Psi}^2 + \chi \delta^2) dt \quad (3.17)$$

where  $\tilde{\Psi}$  is the heading error and  $\chi$  is a weighting factor.

If we describe the ship steering dynamics by (2.13), then it is a simple exercise in optimal control theory to find the feedback which minimizes (3.17). See for example [Åström, 1970; Anderson and Moore, 1971; Kwakernaak and Sivan, 1972]. The solution is given by

$$\delta = -K_p \Psi_e - K_d \dot{\Psi} \quad (3.18)$$

where

$$K_p = \begin{cases} \frac{1}{\sqrt{\chi}} & b > 0 \\ -\frac{1}{\sqrt{\chi}} & b \leq 0 \end{cases}$$

$$K_d = \begin{cases} \sqrt{\left(\frac{a}{b}\right)^2 + \frac{2}{b\sqrt{\chi}}} - \frac{a}{b} & b > 0 \\ -\sqrt{\left(\frac{a}{b}\right)^2 - \frac{2}{b\sqrt{\chi}}} + \frac{a}{b} & b \leq 0 \end{cases} \quad (3.19)$$

where  $a = \frac{1}{T}$  and  $b = \frac{K}{T}$

Note that the gain  $K_p$  depends only on  $\chi$ , while  $K_d$  depends also on the dynamics of the ship.

### Example 3.3

Consider a tanker with  $a = 0.01$ ,  $b = -0.0005$  and  $\chi = 0.08$ . The controller parameters  $K_p$  and  $K_d$  can be found from (3.19) as follows:  $K_p = -3.54$  and  $K_d = -100.59$ . The performance of the autopilot is shown in Figure 3.6. The desired response of the Figure is a critically damped second order system with a natural frequency of 0.01 radians per second.

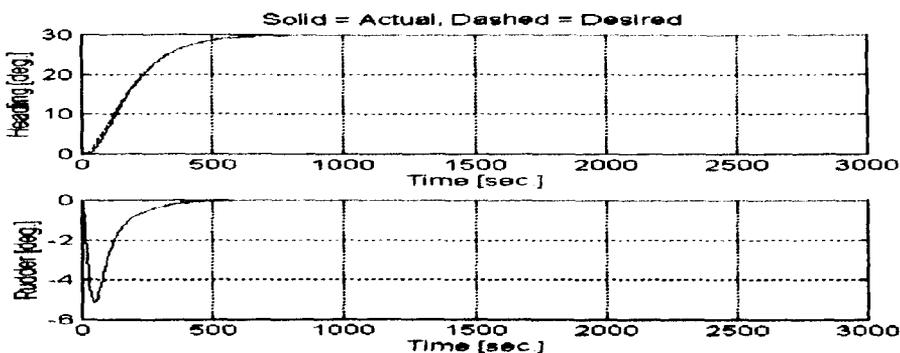


Figure 3.6: Performance of the optimal controller of equation (3.19)

### 3.6.4 Pseudo Derivative Feedback Algorithm:

A pseudo derivative feedback (PDF) controller is essentially a PID controller in which the proportional and derivative terms of the controller are in a feedback loop with the integral term in the forward part of the loop. Vahedipour et al.[1990] have demonstrated that this arrangement leads to smoother rudder activity and gives greater ability to the overall controller to resist load disturbances and responds faster to heading demands compared with standard PID controllers.

The PDF control algorithm was first used for autopilot design by Flower and Sparrius [1986]. Block diagram for an autopilot using a PDF controller is shown in Figure 3.7.

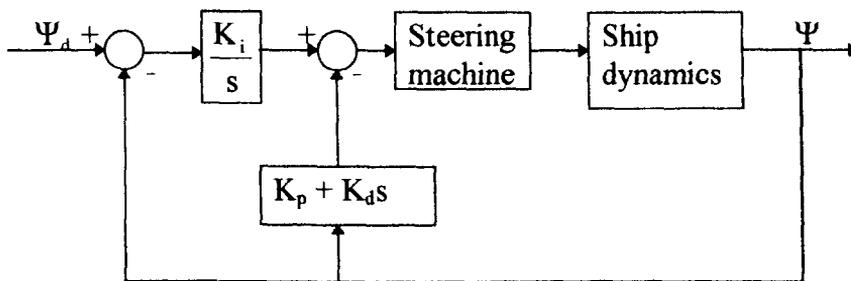


Figure 3.7: Basic block diagram showing structure of PDF controller

## 3.7 Non-Linear Autopilots

In the previous Section, we reviewed some linear autopilots. A number of non-linear autopilots are also available in the literature. In this Section we shall review two of these.

### 3.7.1 Autopilot based on feedback linearization Control laws

The idea of feedback linearization (FB) is to cancel the non-linearities in a system so that the closed loop dynamics will be linear. Fossen and Paulsen [1993] have applied this method to a ship steering control system. The method is reviewed as follows:

Using the Norrbin's nonlinear model given in (2.30), the control law can be expressed as

$$\delta = \hat{m}a_{\psi} + \hat{d}_1 \dot{\Psi} + \hat{d}_3 \dot{\Psi}^3 \quad (3.20)$$

where '^' indicates the estimate of the real parameters and  $a_{\psi}$  the commanded acceleration. Putting (3.20) into (2.30) results in

$$m(\ddot{\Psi} - a_{\psi}) = \tilde{m}a_{\psi} + \tilde{d}_1 \dot{\Psi} + \tilde{d}_3 \dot{\Psi}^3 \quad (3.21)$$

In this equation, the symbol '~' indicates the parameter error, i.e.  $\tilde{m} = \hat{m} - m$ ,  $\tilde{d}_1 = \hat{d}_1 - d_1$  and  $\tilde{d}_3 = \hat{d}_3 - d_3$ .

If there are no parametric uncertainties, equation (3.21) reduces to

$$\ddot{\Psi} = a_{\psi} \quad (3.22)$$

which suggests that the commanded acceleration should be chosen as:

$$a_{\psi} = \ddot{\Psi}_d - K_d \dot{\tilde{\Psi}} - K_p \tilde{\Psi} \quad (3.23)$$

where  $\tilde{\Psi}$  is the heading error. This in turn yields the error dynamics:

$$\ddot{\tilde{\Psi}} + K_d \dot{\tilde{\Psi}} + K_p \tilde{\Psi} = 0 \quad (3.24)$$

With this commanded acceleration the closed loop system has been transformed to a linear second order system. By changing the values of the constants  $K_d$  and  $K_p$ , we change the position of the poles.

Using a pole placement algorithm like

$$K_p = \lambda^2; \quad K_d = 2\lambda; \quad \lambda > 0 \quad (3.25)$$

yields the critically damped error dynamics

$$\ddot{\tilde{\Psi}} + 2\lambda\dot{\tilde{\Psi}} + \lambda^2\tilde{\Psi} = 0 \quad (3.26)$$

To ensure convergence of the tracking error in the presence of constant disturbances, it is desirable to include integral action in the control law. In order to achieve this, Fossen and Paulsen [1993] suggest that the commanded acceleration should be modified to

$$a_\psi = \ddot{\Psi}_d - 3\lambda\dot{\tilde{\Psi}} - 3\lambda^2\tilde{\Psi} - \lambda^3 \int_0^t \tilde{\Psi}(\tau) d\tau \quad (3.27)$$

which changes the error dynamics to

$$\ddot{\tilde{\Psi}} + 3\lambda\dot{\tilde{\Psi}} + 3\lambda^2\tilde{\Psi} + \lambda^3\tilde{\Psi} = 0 \quad (3.28)$$

This suggests that

$$K_p = 3\lambda^2, \quad K_d = 3\lambda, \quad \text{and} \quad K_i = \lambda^3 \quad (3.29)$$

### Example 3.4

Consider a ship with the following set of parameters [Van Amerongen, 1982]:  $U = 5\text{m/sec.}$ ,  $K = 0.5$  per sec.,  $T = 31$  sec.,  $\alpha_1 = 1.0$ , and  $\alpha_3 = 0.4 \text{ s}^2$ . (equation (2.30)). The controller parameters can be computed by using equation (3.29).

The performance of the control system at a speed of 5 m/sec. is shown in Figure 3.8 when  $\lambda = 0.2$ .

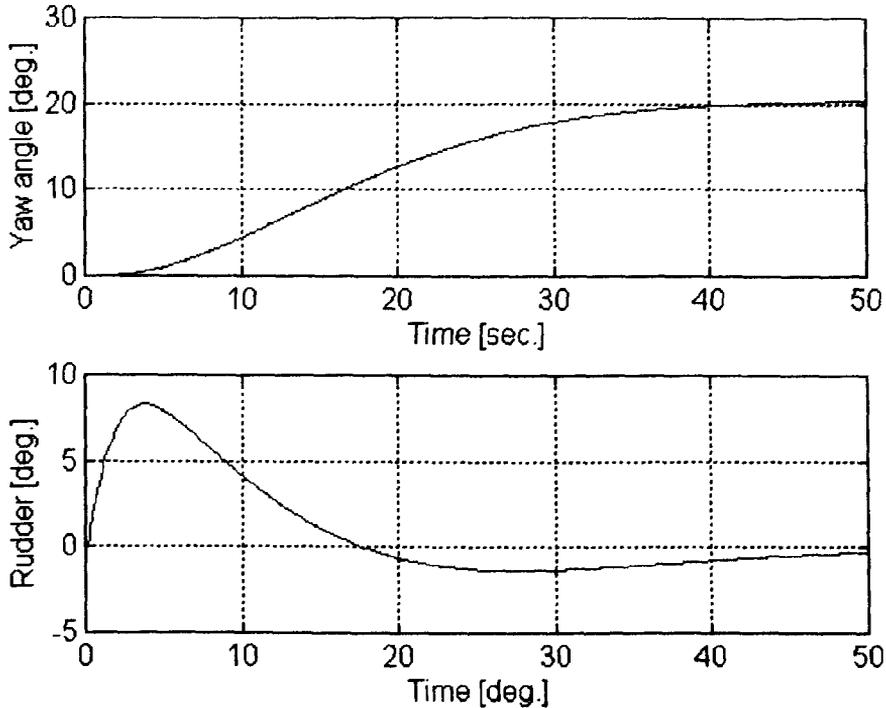


Figure 3.8: Performance of feedback linearization controller at 5 m/s.

### 3.7.2 Sliding Mode Control

Sliding mode control (SMC) is a well established non-linear control technique. This technique is based on the assumption that the system states are directly measurable. The state space is divided into regions by *switching surfaces* and these regions determine the value of the system control signal. The basic theory of this technique can be found in [Utkin, 1992; Edwards and Spurgeon, 1998]. The theory has been applied successfully in the control of under water vehicles by many researchers [e.g. Yoerger and Slotine, 1984, 1985, 1991, 1995; Cristi et al., 1990; Dougherty and Woolweaver, 1990;

Healey and Lienard 1993]. Recently McGookin [1997] has applied decoupled SMC theory to the control of a submarine and a super tanker.

A decoupled SM controller can be derived from the linear state space representation of the decoupled dynamics that are being controlled. A decoupled system is a subsystem whose states are selected in such a way that the dominant dynamics of the manoeuvre (that is being controlled) are decoupled from the dynamics that have very little influence on the manoeuvre. The resulting controller has the following form [McGookin, 1997]:

$$\mathbf{u}_s = -\mathbf{k}_s^T \mathbf{x}_s + \left( \mathbf{h}_s^T \mathbf{x}_s \right)^{-1} \left( \mathbf{h}_s^T \dot{\mathbf{x}}_{sd} - \eta_s \tanh \left( \frac{\sigma_s(\hat{\mathbf{x}}_s)}{\phi_s} \right) \right) \quad (3.30)$$

The derivation of the controller is given in appendix B. In the above equation,  $\mathbf{k}_s$  is the feedback gain vector for the subsystem,  $\mathbf{x}_s$  is subsystem state vector,  $\hat{\mathbf{x}}_s$  is subsystem state error,  $\mathbf{h}_s$  is the right eigen vector of the desired closed loop system,  $\mathbf{b}_s$  is subsystem input vector and  $\mathbf{x}_{sd}$  is the desired state vector. The tanh term provides the switching action which characterises SM controllers. The magnitude of this switching action is determined by  $\eta$ , the switching gain and its activity governed by the sliding surface  $\sigma$ .  $\phi$  is called the boundary layer thickness which smoothes the switching action in order to eradicate high frequency chattering [Healey and Marco, 1992; McGookin, 1993, Fossen, 1994].

### 3.8 Adaptive autopilots:

A fixed structure controller can provide optimal performance only at the operating point it is designed for. As mentioned in the previous Chapter, the ship dynamics vary with different operating conditions which include speed,

trim, loading and water depth. The dynamics also vary with external disturbances such as wind, waves and ocean currents. When the internal and external disturbances become significant, it is tedious and difficult to properly determine the fixed parameters of the controller that result in good performance. Fixed controller settings often result in energy losses due to excessive rudder motion or inadequate manoeuvring performance. An adaptive controller can avoid these problems of a fixed structure autopilot.

Development towards adaptive autopilots began in the 1970s and has remained a major area of research until recently. Adaptive control of ships provides several benefits, for example, improved fuel economy, increase speed of vessel, and reduced steering. Adaptive control also makes the ship operation more convenient in all weather conditions.

There has been much research effort into the design of adaptive autopilots. Oldenburg [1975], Sugimoto and Kojima [1978], and Kanamaru and Sato [1979] suggest monitoring the environment prior to adjusting the autopilot. Self tuning regulators have been proposed by numerous authors [e.g. Brink et al., 1978; Jia and Song, 1987, Källström et al., 1979; Lim and Forsythe, 1983; Mort and Linkins, 1981; Tiano et al., 1980], and model reference adaptive control (MRAC) has been used by Honderd and Winkelman, 1972; Van Amerongen and Udink Ten Cate, 1975; Van Amerongen, 1982, 1984; and Arie et al., 1986. Van Amerongen has worked extensively on this problem and his PhD thesis [Van Amerongen, 1982] contains many practical insights. Hill climbing methods for optimizing a performance index have been reported in [Oldenburg, 1975; Schilling, 1976, Broome and Lambert, 1978; Reid and Williams, 1978; Arie et al. 1986] and stochastic methods for identification and control have been attempted in [Millers, 1973; Merlo and Tiano, 1975; Ohtsu et al., 1979; Herther et al., 1980]. Other important

contributions in adaptive autopilots include [Katebi and Byrne, 1988; Fossen and Paulsen, 1993; Rubio and López, 1993; Desanj et al. 1997 etc.].

Here we review only the MRAC approach of Van Amerongen [1982].

### 3.8.1 Model Reference Adaptive Control

In a model reference adaptive control (MRAC) system the desired behaviour is specified by a reference model, and the parameters are adjusted based on the error, which is the difference between the output of the closed loop system and the reference model. The parameters can be adjusted by using a gradient method or by applying Lyapunov stability theory. The following review is based on the later approach.

Assume that the ship dynamics are represented by the Nomoto's first order model:

$$T\ddot{\Psi} + \dot{\Psi} = K\delta + K_w \quad (3.31)$$

where  $K_w$  is a slowly varying term due to external disturbances. Suppose that the control law is:

$$\delta = \hat{K}_p(\Psi_r - \Psi) - \hat{K}_d\dot{\Psi} - \hat{K}_i \quad (3.32)$$

where  $\hat{K}_p$ ,  $\hat{K}_d$  and  $\hat{K}_i$  are the adaptive estimates of the PID controller parameters respectively, to be determined later. The closed loop dynamics resulting from (3.31) and (3.32) can be obtained as

$$T\ddot{\Psi} + (1 + K\hat{K}_d)\dot{\Psi} + K\hat{K}_p\Psi = K\hat{K}_p\Psi_r + (K_w - K\hat{K}_i) \quad (3.33)$$

Equating this dynamics with the dynamics (3.1) of the reference model, we get

$$K_p = \frac{K_m T}{T_m K} \quad (3.34)$$

$$K_d = \frac{1}{K} \left( \frac{T}{T_m} - 1 \right) \quad (3.35)$$

$$K_i = \frac{K_w}{K} \quad (3.36)$$

These values of  $K_p$ ,  $K_d$  and  $K_i$  are called the perfect matching conditions.

The estimates  $\hat{K}_p$ ,  $\hat{K}_d$  and  $\hat{K}_i$  can be computed as follows:

The reference model given in (3.1) can be rewritten as

$$\ddot{\tilde{\Psi}} + \frac{1}{T_m} \dot{\tilde{\Psi}} + \frac{K_m}{T_m} \tilde{\Psi} = \frac{K_m}{T_m} \Psi_r - \ddot{\Psi} - \frac{1}{T_m} \dot{\Psi} - \frac{K_m}{T_m} \Psi \quad (3.37)$$

multiplying both sides of the above equation by  $\frac{T}{K}$  and applying (3.34), (3.35)

and (3.36) yields the following error system:

$$\dot{x} = Ax + B\phi^T \tilde{\Theta} \quad (3.38)$$

where  $x = \begin{bmatrix} \Psi_d - \Psi \\ \dot{\Psi}_d - \dot{\Psi} \end{bmatrix}$ ,  $A = \begin{bmatrix} 0 & 1 \\ -\frac{K_m}{T_m} & -\frac{1}{T_m} \end{bmatrix}$ ,  $B = \begin{bmatrix} 0 \\ b \end{bmatrix}$ ,  $\phi = [\Psi - \Psi_r \quad \dot{\Psi} \quad 1]$ ,

$$\tilde{\Theta} = \begin{bmatrix} \tilde{K}_p \\ \tilde{K}_d \\ \tilde{K}_i \end{bmatrix}.$$

Here  $b = \frac{K}{T}$ ,  $\tilde{K}_p = \hat{K}_p - K_p$ ,  $\tilde{K}_d = \hat{K}_d - K_d$  and  $\tilde{K}_i = \hat{K}_i - K_i$ . Clearly the

control objective is that both the heading angle error and heading rate error

should converge to zero. This can be guaranteed by applying the following theorem [Åström and Wittenmark, 1995]:

$$\dot{\hat{\Theta}} = -\Gamma \frac{1}{|b|} \phi \mathbf{B}^T \mathbf{P} \mathbf{x}; \quad \Gamma = \Gamma^T > 0 \quad (3.39)$$

where  $\mathbf{P} = \mathbf{P}^T > 0$  satisfies the Lyapunov equation:

$$\mathbf{A}^T \mathbf{P} + \mathbf{P} \mathbf{A} = -\mathbf{Q}; \quad \mathbf{Q} = \mathbf{Q}^T > 0 \quad (3.40)$$

The adaptation law (3.39) can be implemented to rewrite the term:

$$\frac{\mathbf{B}^T}{|b|} = \begin{bmatrix} 0 & \mathbf{B} \end{bmatrix} = [0 \quad \text{sgn}(b)] \quad (3.41)$$

The main advantage of (3.41) is that the magnitude of the ratio  $b = \frac{K}{T}$  is not used, only the sign of the ratio is sufficient.

Now ((3.39) can be written as

$$\hat{K}_p = -\gamma_1 \text{sgn}(b)(\Psi - \Psi_r) \quad (3.42)$$

$$\hat{K}_d = -\gamma_2 \text{sgn}(b)\dot{\Psi}_e \quad (3.43)$$

$$\hat{K}_i = -\gamma_3 \text{sgn}(b)e \quad (3.44)$$

where  $e = \mathbf{P}_{21}(\Psi_d - \Psi) + \mathbf{P}_{22}(\dot{\Psi}_d - \dot{\Psi}_d)$ ,  $\mathbf{P}_{21} = \mathbf{P}_{12}$ ,  $\mathbf{P} = \begin{bmatrix} \mathbf{P}_{11} & \mathbf{P}_{12} \\ \mathbf{P}_{21} & \mathbf{P}_{22} \end{bmatrix}$  and

$$\Gamma = \text{diag}(\gamma_1, \gamma_2, \gamma_3)$$

Further details can be found in [Åström and Wittenmark, 1995].

### 3.9 Intelligent Autopilots

Despite their potential benefits, adaptive control systems have some disadvantages. These include the following:

- The design and analysis of nonlinear adaptive systems is difficult and in comparison with neural networks leading to relatively expensive solutions in computational terms [Pao, 1989].
- Some forms of adaptive control systems do not have long term memory and therefore do not retain the optimal controller parameters corresponding to different configurations of the plant [Narendra and Mukhopadhyay, 1992].
- They need enough *a priori* information for successful applications [Tulunay, 1991]
- There is some concern about potential instabilities associated with adaptive system behaviour [Simensen and Murray-Smith, 1995].

These and other disadvantages of adaptive control systems provide motivation for intelligent control. The development of intelligent autopilots for ships is in an early stage, and only a few papers have appeared in the literature on the applicability of fuzzy logic [Garcia and Castelo, 1993; Layne and Passino, 1993; Parsons, et al., 1995; Tomera and Morawski, 1996; Arbil, et al., 1997] and neural networks [Endo et al. 1989; Burns, 1995; Zhang et al., 1995; Unar and Murray-Smith, 1997a, 1997b] for ship steering control system. We have already discussed the feasibility of ANNs for ship steering control systems in Chapter 1. The development of fuzzy logic autopilots is

beyond the scope of this thesis. However, the development of ANN autopilots will be carried out in detail in later Chapters.

## Chapter 4

### ARTIFICIAL NEURAL NETWORKS

The field of artificial neural networks (ANNs) has become enormously fashionable area of research in recent years and ANNs have found numerous successful applications in almost every field of science and engineering. ANNs can easily handle complicated problems and can identify and learn correlated patterns between sets of input data and corresponding target values. After training, these networks can be used to predict the outcome from new input data. Neural networks mimic the human learning process and can handle problems involving highly non-linear and complex data even if the data are imprecise and noisy. They are ideally suited for pattern recognition and do not require a prior fundamental understanding of the process and phenomenon being modelled. They are also highly suitable for applications involving parameter varying and/or time varying systems.

A universal definition of artificial neural networks is not available, however, the following definition summarizes the basic features of an ANN:

*Artificial Neural Networks, also called Neurocomputing, or parallel distributed processes (PDP) or connectionist networks or simply neural networks are interconnected assemblies of simple processing elements, called neurons, units or nodes, whose functionality is loosely based on the biological neuron. The processing ability of the network is stored in the inter-unit connection strength, or weights, obtained by a process of adaptation to, or learning from, a set of training patterns [Hecht-Nielsen, 1990].*

#### 4.1 Historical Motivation:

The history of ANNs begins with the pioneering work of McCulloch and Pitts who designed a very simple artificial neuron in 1943. The neurons were binary, summing their unweighted inputs and performing a threshold

operation. ANNs sprang into existence around the same time as the first computers, and it is widely known that John Von Neumann, instrumental in the construction of the modern serial computer was heavily influenced by the work of McCulloch and Pitts.

In 1949 Donald Hebb's famous book *The Organization of Behaviour* was published. In this book, Hebb postulated a plausible qualitative mechanism for learning at cellular level in brains. An extension of his proposals is widely known today as the *Hebbian Learning Rule*. In a biological context this rule states that if two neurons are fired simultaneously, the connection strength between them is increased, otherwise it is weakened.

In 1951, Minsky constructed the first neurocomputer, *the Snark* [Minsky, 1954]. The neurocomputer did operate successfully from a technical standpoint (it adjusted its weights automatically), but never actually carried out any particular interesting information processing function. Nonetheless, it provided design ideas that were used later by other investigators.

In 1958 Rosenblatt developed his neurocomputer, *the Perceptron*. He proposed a learning rule, *the perceptron convergence theorem*, and proved that, given linearly separable classes, a perceptron would, in a finite number of trials, develop a weight vector that would separate the classes [Rosenblatt, 1960a, 1960b]. Rosenblatt also wrote a book on neurocomputing, *Principles of Neurodynamics* [Rosenblatt, 1962], that is still worth reading.

In 1960, Widrow and Hoff [Widrow and Hoff, 1960; Widrow and Lehr, 1990] introduced the least mean squares (LMS) algorithm and used it to formulate the ADALINE (adaptive linear element). The difference between the perceptron and the ADALINE lies in the training procedure. The LMS algorithm is still in widespread use, particularly in the field of adaptive signal processing.

Although people like Widrow approached this field from an analytic point of view, most of the research on this field was done from an experimental point of view. In the 1960s many sensational promises were made which were not fulfilled. This discredited much early research on ANNs. About the same time Minsky and Papert began promoting the field of artificial intelligence (AI) at the expenses of ANN research. Their book *Perceptrons* [Minsky and Papert, 1969] almost served as death sentence for ANN research. In this book they mathematically proved that perceptrons were not able to compute certain essential computer predictions like the EXCLUSIVE OR Boolean function.

From the late 1960s to the early 1980s, research on ANN was almost non-existent. Notable exceptions from this period are the works of Amari [1967, 1972, 1977], Anderson [1968, 1972, 1983, 1985], Fukushima [1975, 1980] and Grossberg [1972, 1976, 1980, 1982]. In 1982, Hopfield led the resurgence of interest in ANNs. His work on the *Hopfield net*, a type of associative memory brought the field increased respect and exposure. His network could recall previously stored patterns when similar noisy patterns are presented.

The introduction of self organizing maps by Kohonen in 1982, simulated annealing by Kirkpatrick et al. in 1983, and Boltzmann learning by Ackley et al. in 1985 further popularized the field of ANNs.

The real breakthrough in ANN research came with the discovery of the back-propagation algorithm. Although it was discovered in 1974 [Werbos, 1974], it was not until the mid 1980s that the back propagation technique became widely publicized [Rumelhart and McClelland, 1986a, 1986b]. This algorithm still dominates the neural network literature and thousands of academic, industrial and government researchers report the results of back propagation simulations and applications at technical conferences and in journals every year.

In 1987, the first conference on neural networks, *The IEEE International Conference on Neural Networks* (1700 participants) was held in San Diego USA and the *International Neural Network Society* (INNS) was formed. In 1988 the INNS journal *Neural Networks* was formed, followed by *Neural Computation* in 1989, the *IEEE Transactions on Neural Networks* in 1990 and subsequently many others.

In 1988 Broomhead and Lowe introduced radial basis function (RBF) networks to the neural network community. The theory of these networks was further enriched by Poggio and Girosi in 1990. A generalization of RBF networks, known as the *Local Model Networks* (LMNs) was introduced by Johansen and Foss in 1992-93 [Johansen and Foss, 1992a, 1992b, 1992c, 1993] and was further popularized by Murray-Smith [1994].

Indeed the above developments have made a very important contribution to the success of ANNs. However, there are also other reasons for the recent interest in ANNs.

One is the desire to build a new breed of powerful computers, that can solve problems that are proving to be extremely difficult for current digital computers and yet are easily done by humans in everyday life. Cognitive tasks like understanding spoken and written language, image processing, retrieving contextually appropriate information from memory are examples of such tasks.

Another is the benefit that neuroscience can obtain from ANN research. New network architectures are constantly being developed, and new concepts and theories being proposed to explain the operation of these architectures. Many of these developments can be used by neuroscientists as new paradigms for building functional concepts and models of elements of the brain.

Many ANN architectures have been proposed. These can be roughly divided into three large categories: feedforward (multilayer) neural networks, feedback neural networks and cellular neural networks. This thesis is solely concerned with the feedforward neural networks. Therefore, other types of ANNs will not be described in this Chapter. The Chapter is organized as follows. In Section 4.2 a simplified model of biological neuron is described and in Section 4.3 a mathematical model of a single neuron inspired from the biological neuron, is developed. Section 4.4 describes very briefly ANN architectures in general. An overview of feedforward neural networks is presented in Section 4.5. Multilayer perceptron networks are described in Section 4.6. The Back-propagation algorithm is derived for the training of MLPs and their function approximation capabilities are discussed. Section 4.7 covers basic features of radial basis function networks. The orthogonal least squares algorithm for the training of these networks is derived and function approximation capabilities are discussed.

## 4.2 Biological Neuron:

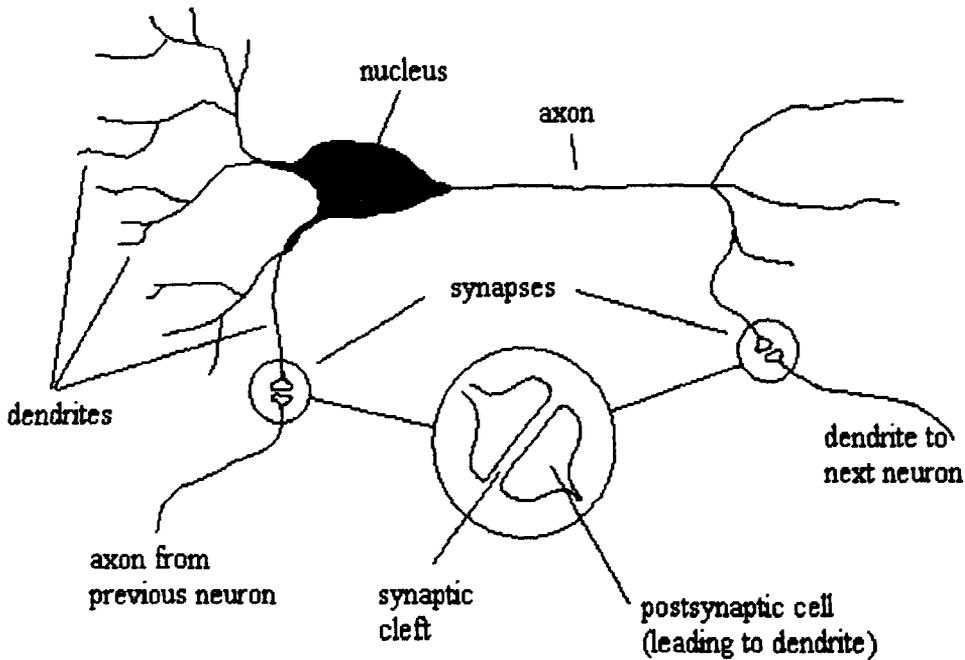


Figure 4.1: A simplified view of a biological neuron

A neuron or nerve cell is the basic building block of all biological brains. Each neuron acts as a simplified numerical processing unit. Figure 4.1 shows a very simplified view of a biological (real) neuron. In this schematic we can recognize the following fundamental parts:

- (i) **Soma or Cell body:** The central part of a neuron is called the soma or cell body which contains the nucleus and the protein synthesis machinery. The size of soma of a typical neuron is about 10 - 80  $\mu\text{m}$  [Müller, et al., 1995]. Almost all the logical functions are realized in this part of a neuron.
- (ii) **The Axon:** It is a long thin tubular fibre which divides itself into a number of branches towards its end. Its length can be from 100  $\mu\text{m}$  to 1 m [Anderson, 1995]. The function of an axon is to transmit the generated neural activity to other neurons or to muscle fibres. In other words, it is the output channel of the neuron. The point where the axon is connected to its cell body is called the Hillock zone.
- (iii) **The dendrites:** The dendrites represent a highly branching tree of fibres and are attached to the soma. The word *dendrite* has been taken from the Greek word *dendro* which means *tree*. Dendrites connect the neuron to a set of other neurons. Dendrites either receive inputs from other neurons or connect other dendrites to the synaptic outputs.
- (iv) **Synapses:** The junction at which a signal is passed from one neuron to the next is called a synapse (from the Greek verb *to join*). It has a button like shape with diameter around 1  $\mu\text{m}$ . Usually a synapse is not a physical connection (the axon and the dendrite do not touch) but there is a gap called the *synaptic gap* or *synaptic cleft* that is normally between 200 Å to 500 Å. (1 Å =  $10^{-10}$  m). The strength of synaptic connection between neurons can be chemically altered by the brain in response to favourable

and unfavourable stimuli in such a way as to adapt the organism to function optimally within its environment.

In a biological neuron, signals enter the neuron from other neurons via the dendrites. Since a neuron has many dendrites, therefore it is capable of receiving signals from many other neurons at a time. The neuron combines these inputs and pass them onto other neurons via the axon. The output signal from a neuron is stimulated by the inputs to it. Research has shown (see for example, Thompson [1993] and references therein) that an output will only occur provided there are enough inputs of sufficient strength to overcome some threshold value, and the output is some non-linear function of the sum of the input stimuli.

The human brain is a bundle of many billions of neurons all heavily interconnected and operating in parallel. It is estimated that the human cerebral cortex contains 100 billion neurons. Each neuron may have as many as 1000 dendrites and, hence within the cerebral cortex there are approximately 100,000 billion synapses [Kartalopoulos, 1996]. Moreover, the behaviour of the real nervous system is very complex and is not yet fully known. It is therefore almost impossible and also inefficient to simulate a full behaviour of human brain. Luckily, a large body of research indicates that simple models, which account for only the most basic neuron processes can provide excellent solutions to practical problems. Therefore, the artificial neuron models are not exactly constrained by real neurons and are based only loosely on biology.

### **4.3 Model of a single artificial neuron:**

In its most simplest form, an artificial neuron can be modelled as a device (usually non-linear) having one or more inputs, as shown in Figure 4.2. An input to an artificial neuron is either an input of the network of which the

neuron is a part, the output of another neuron, or its own output. As can be seen from the Figure, an artificial neuron first multiplies each input by a factor called weight.

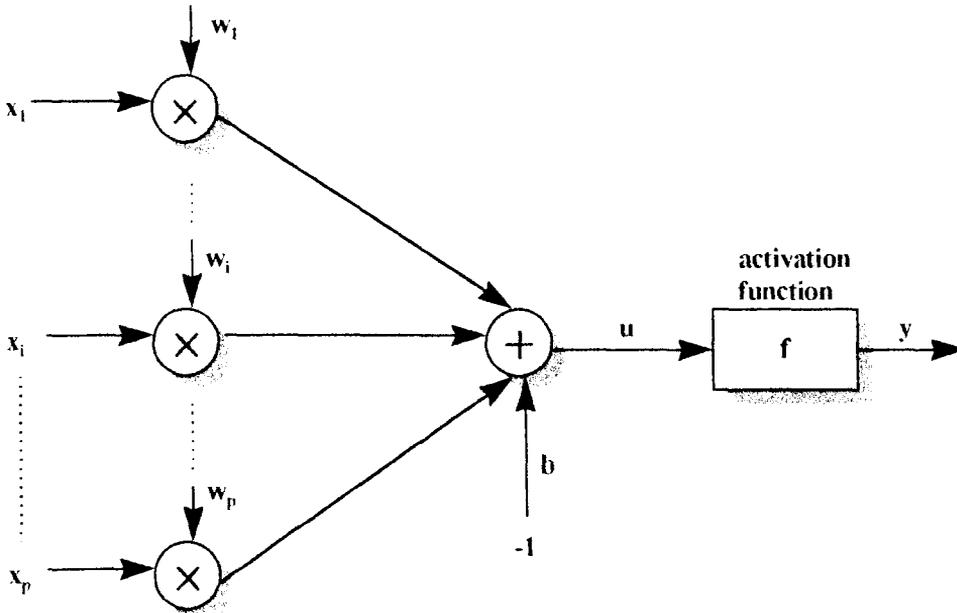


Figure 4.2: Model of a single neuron

Each input  $x_i$  ( $i = 1, 2, \dots, p$ ) has its own weight  $w_i$  which can be varied in strength, in analogy with neurobiological synapses. The neuron then calculates the sum of all the weighted inputs. In other words, the neuron calculates the linear combination of the values presented to its inputs. In most cases a bias  $b$  is also added to the weighted sum. Finally, the neuron applies an *activation function*  $f$  to the weighted sum. An activation function is also known as a *transfer function* or a *squashing function* which can be a linear or non-linear function. Some choices of activation functions are depicted in Figure 4.3.

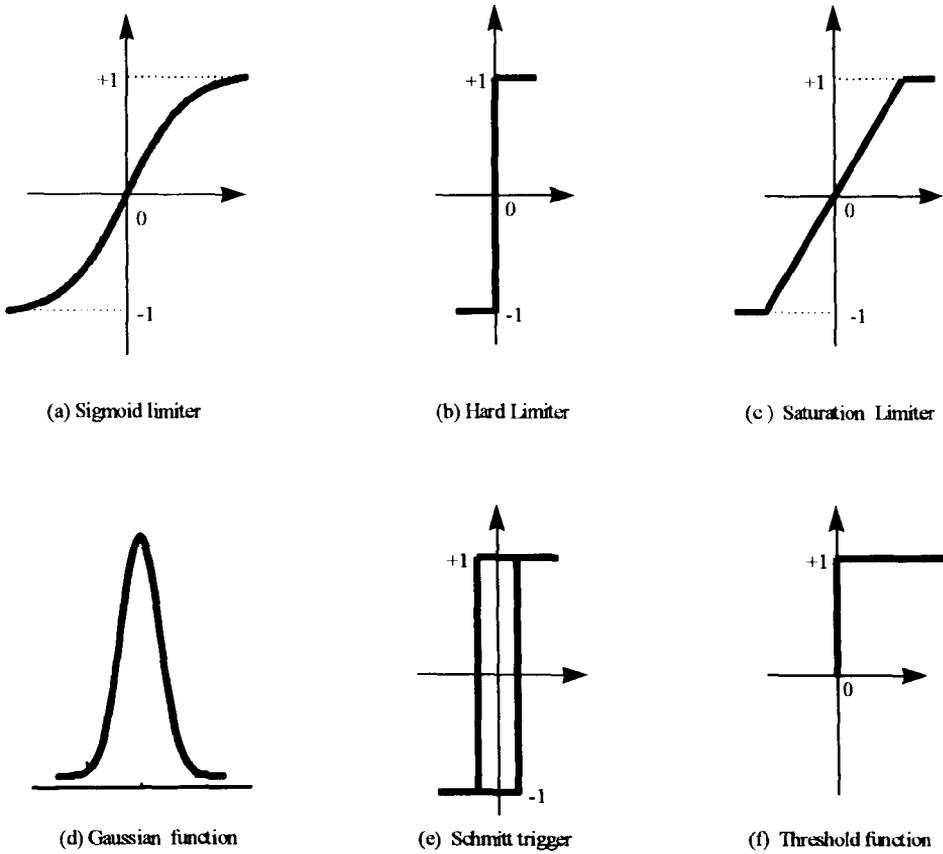


Figure 4.3: Some choices of activation functions

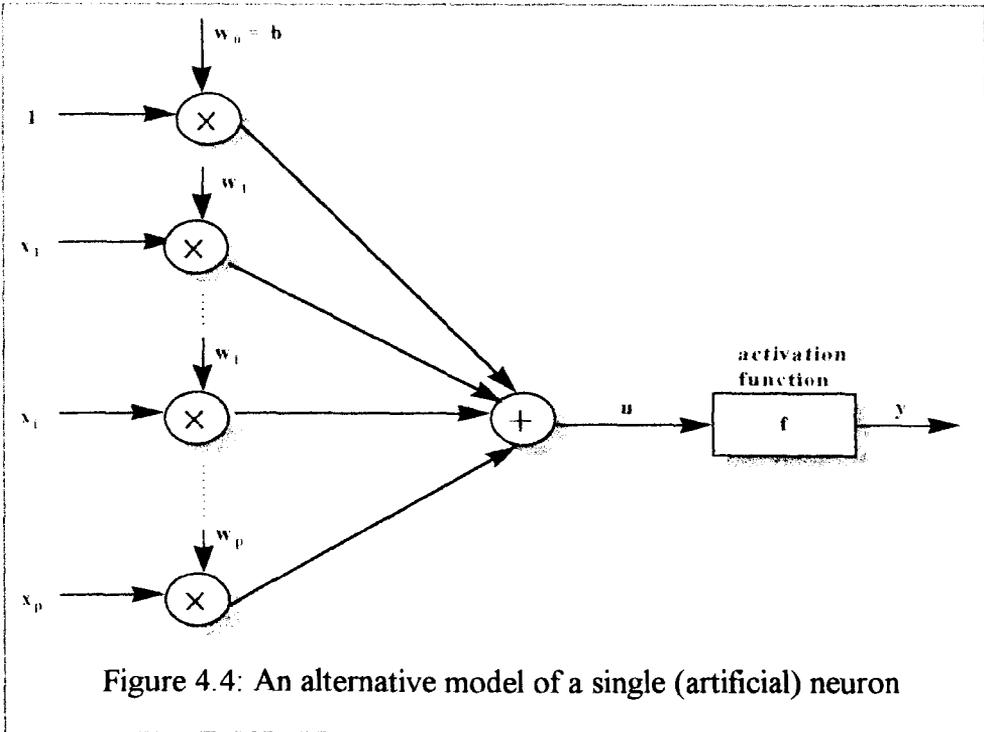
The artificial neuron of Figure 4.2 can be expressed mathematically as follows:

$$y = f\left(\sum_{i=1}^p w_i x_i - b\right) \quad (4.1)$$

where  $p$  is the total number of inputs.

In most cases, the threshold contribution  $b$  is treated as an extra input  $x_0$  to the neuron, as shown in Figure 4.4. By choosing  $x_0 = -1$  and  $w_0 = b$ , equation (4.1) simplifies to:

$$y = f\left(\sum_{i=0}^p w_i x_i\right) \quad (4.2)$$



#### 4.4 ANN Architectures:

Connecting several artificial neurons in some specific manner yields an artificial neural network. The architecture of an ANN defines the network structure, that is the number of artificial neurons in the network and their inter connectivity. In a typical ANN architecture, the artificial neurons are connected in layers and they operate in parallel. The weights or the strength of connection between the neurons are adapted during use to yield good performance. Each ANN architecture has its own learning rule. Several architectures have been proposed. Here we shall only describe the multilayer feedforward architecture. Multilayer feedforward neural networks have proved their worth particularly in engineering applications. They are easy to implement, have no stability problems [Cichocki, and Unbehauen, 1993] and

possess universal approximation property as described in the following sections.

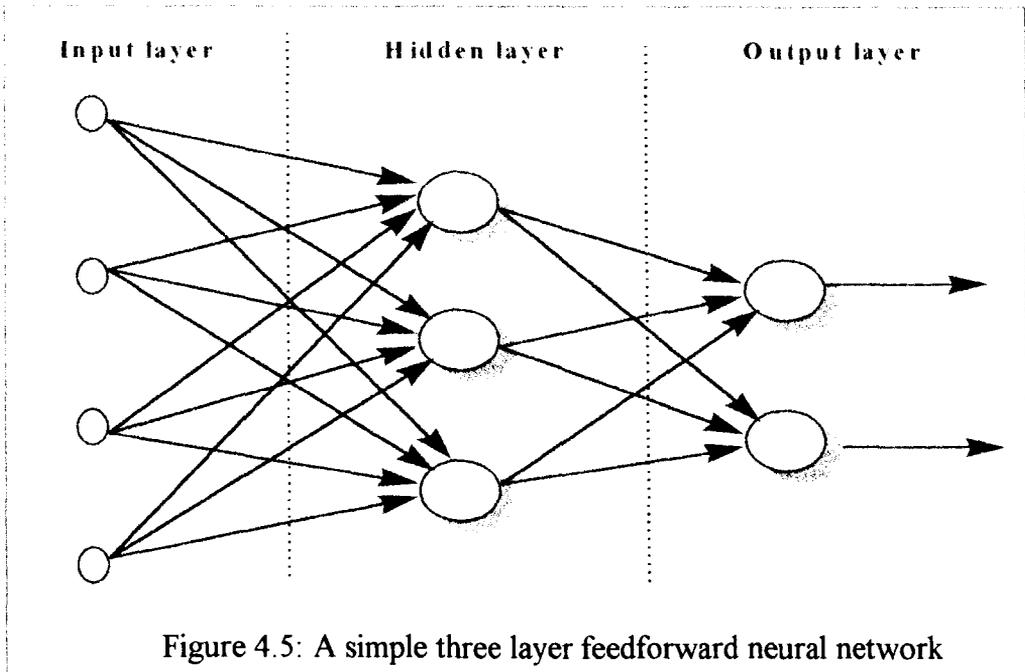
#### **4.5 Multilayer Feedforward Neural Networks:**

The feedforward type of neural network is probably the most popular ANN architecture and has found numerous successful applications, particularly in the field of systems and control. The reason why this architecture is so popular is partly because it is very easy to implement and partly because it allows *supervised learning*. In supervised learning the network is required to give some desired output for a certain class of input patterns.

The feedforward architecture is simple in the sense that it is layered. A layer consists of an arbitrary number of (artificial) neurons or nodes. In most cases, all the neurons in a particular layer contain the same activation function. However, the neurons in different layers of a network may have different activation functions. When a signal passes through the network, it always propagates in the forward direction from one layer to the next, not to the other neurons in the same or the previous layer. This is illustrated in Figure 4.5.

In a feedforward neural network, the first layer is called the *input layer*, and is the layer where the input patterns based on the data sample are fed in. This layer does not perform any processing and consists of fan out units only. Then follows one or more *hidden layers*, and as the name indicates these layers can not be accessed from outside the network. The hidden layers enable the network to learn complex tasks by extracting progressively more meaningful features from the input patterns [Haykin, 1994]. The final layer is

the *output layer*, which may contain one or more output neurons. This is where the network decision, for a given input pattern, can be read out.



Because the input to the network is known, and the activation functions are chosen by the user, the total output can be calculated as a function of the weights. Thus by changing the internal weights of the network one can adjust the total output to any desired value. This is the basic concept behind the supervised learning.

Since the pioneering work of Rumelhart and co-workers [Rumelhart and McClelland, 1986a, 1986b], a large number of papers on the application of feedforward neural networks have been published. These applications can be divided into two major groups: *function approximation* and *classification*. This division is based on the type of the desired output(s) required to accomplish the task. If the output values are continuous, the feedforward network is performing function approximation, whereas if the outputs are restricted to a finite set of values, it is doing classification.

A multilayer feedforward neural network can be divided into two major categories: the multilayer perceptron (MLP) networks and radial basis function (RBF) networks. These networks are described in the following sections.

#### 4.6 Multilayer Perceptron Networks:

The multilayer perceptron network is perhaps the best known type of feedforward neural network. It has found successful applications in almost every field of science and engineering. These applications include the following:

- Speech recognition [Cohen et al. 1993; Tadeusiewicz et al. 1998].
- Character recognition [Guyon, 1991; Mozayyani and Vaucher, 1997].
- System identification [Narendra and Parthasarathy, 1990; Chen et al., 1990; Bulsari and Saxon, 1991; Chen and Billings, 1994];
- Modelling and Control [Billings et al. 1992; Kruger and Naunin, 1996; Lightbody et al. 1997; Tulunay et al. 1998].
- Robotics [Kim and Lee, 1996].
- Biomedicine [Baxt, 1992; Robinson, 1992; Hazarika et al. 1998; Oguri and Iwata, 1998].

The structure of an MLP network is similar to that shown in Figure 4.5. It consists of an input layer, one or more hidden layers and an output layer. The number of hidden layers and the number of neurons in each layer is not fixed. Each layer may have a different number of neurons, depending on the application. The developer will have to determine how many layers and how many neurons per layer should be selected for a particular application. Generally, an MLP network has a different number of neurons and different

synaptic weights for different layers. All neurons in hidden layer have a sigmoidal nonlinearity such as a *logistic function*:

$$y_i = \frac{1}{1 + \exp(-u_i)} \quad (4.3)$$

or a *hyperbolic tangent* function:

$$y_i = a \tanh b(u_i) \quad (4.4)$$

where  $u_i$  is the net internal activity level of neuron  $i$ ,  $y_i$  is the output of the same neuron, and  $a$ ,  $b$  are constants. Generally, an MLP network learns faster with hyperbolic tangent function than the logistic function [Haykin, 1994]. The important point to emphasize here is that the nonlinearity is smooth (i.e. differentiable everywhere). The output layer neurons may have the same activation function as the hidden neurons. However, many applications use a linear function as the activation function of the output layer neurons. In other words, the output of each of these neurons is equal to its net input. An MLP network is usually trained by the back-propagation rule which is derived in the next Section.

#### 4.6.1 Error Back-Propagation Algorithm:

The development of the error back-propagation (or simply back-propagation) algorithm represents a landmark in ANNs in that it provides a computationally efficient method for the training of MLP networks.

The back-propagation algorithm was originally introduced by Paul Werbos in 1974 [Werbos, 1974]. This algorithm was rediscovered independently by David Parker in 1985 [Parker, 1985] and Rumelhart et al. in 1986 [Rumelhart

and McClelland, 1986a, 1986b]. A mathematically similar recursive control algorithm was presented by Arthur Bryson and Yu Chi Ho in 1969 [Bryson and Ho, 1975]. Rosenblatt also came very close to discovering the key to training perceptrons when he proposed a heuristic algorithm to adapt weights of his perceptron network. On page 292 of his book *Principles of Neurodynamics* [Rosenblatt, 1962], Rosenblatt states,

*“The procedure to be described here is called the ‘back propagating error correcting procedure’ since it takes its cue from the error of the R-units (the output units), propagating corrections back towards the sensory end of the network (the input units) if it fails to make a satisfactory correction quickly at the response end (output units). The actual correction procedure for the connections to a given unit, whether it is an A-unit (hidden unit) or an R-unit (output unit), is perfectly identical to the correction procedure employed for an elementary perceptron, based on the error indication assigned to the terminal unit”.*

Notwithstanding its chequered history, there is no question that credit for developing back-propagation into a usable technique, as well as promulgation of the MLP architecture to a large audience, rests entirely with Rumelhart and other members of the PDP group [Rumelhart et al. 1986a, 1986b]. Before their work, back-propagation was unappreciated and obscure. Today, it dominates the ANN literature.

The back-propagation algorithm adjusts the weights and biases of an MLP network so as to minimize the sum of squared errors of the network. This is done by continually adjusting the values of the network weights and biases in the direction of steepest descent with respect to error. This procedure is called a *steepest descent* Procedure.

#### **4.6.1.1 Derivation:**

Figure 4.6 shows a neuron  $j$  located in the output layer of an MLP network.

The input to the neuron at iteration  $n$  is

$$u_j(n) = \sum_{i=0}^p w_{ji}(n)y_i(n) \quad (4.5)$$

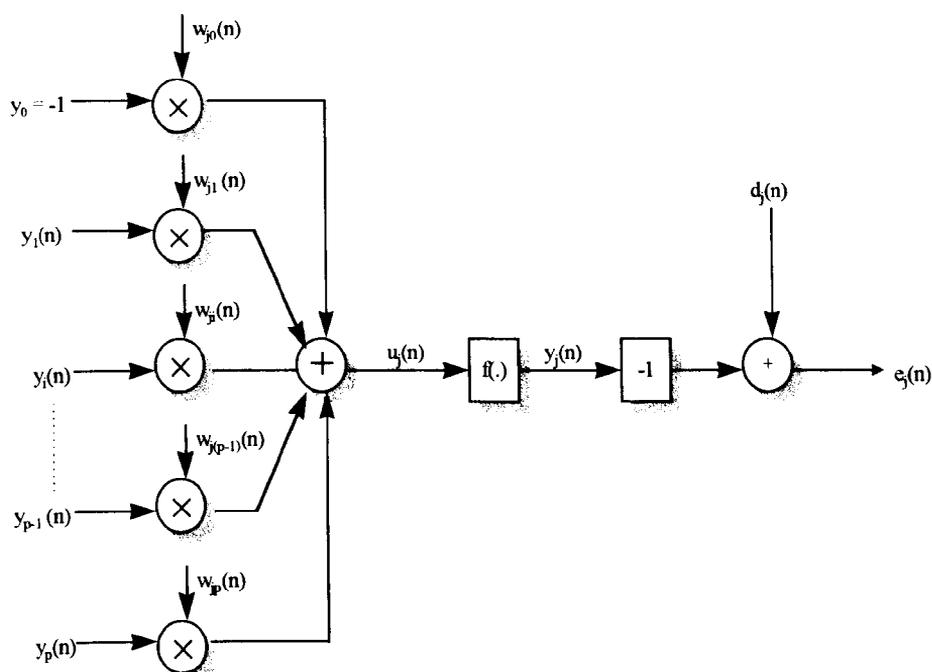


Figure 4.6: Dynamics of output neuron  $j$  of an MLP network

where  $p$  is the total number of inputs and  $p_0$  is the threshold.

The output of the neuron  $j$  will be

$$y_j(n) = f(u_j(n)) \quad (4.6)$$

where  $f(\cdot)$  is the activation function of the neuron. Assume that  $d_j(n)$  be the desired output at iteration  $n$ . The error signal will therefore be given by

$$e_j(n) = d_j(n) - y_j(n) \quad (4.7)$$

The instantaneous value of the squared error corresponding to neuron  $j$  will be

$$EE(n) = \frac{1}{2} e_j^2(n) \quad (4.8)$$

and hence the instantaneous sum of squared errors will be

$$E(n) = \frac{1}{2} \sum_{j \in C} e_j^2(n) \quad (4.9)$$

where  $C$  is a set containing all neurons of the output layer.

If the total number of patterns contained in the training set is  $N$ , then the average squared error of the network will be

$$E_{av} = \frac{1}{N} \sum_{n=1}^N E(n) \quad (4.10)$$

This is the *cost function* of the network which is to be minimized.

Differentiating  $E(n)$  with respect to  $w_{ji}(n)$  and making use of the chain rule, we get

$$\frac{\partial E(n)}{\partial w_{ji}(n)} = \frac{\partial E(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial u_j(n)} \frac{\partial u_j(n)}{\partial w_{ji}(n)} \quad (4.11)$$

The first term  $\frac{\partial E(n)}{\partial e_j(n)}$  on the right hand side (RHS) of the above equation can

be found by differentiating both sides of (4.9) with respect to  $e_j(n)$ :

$$\frac{\partial E(n)}{\partial e_j(n)} = e_j(n) \quad (4.12)$$

The next term, i.e.  $\frac{\partial e_j(n)}{\partial y_j(n)}$  on the RHS of (4.11) can be obtained by differentiating (4.7) with respect to  $y_j(n)$ :

$$\frac{\partial e_j(n)}{\partial y_j(n)} = -1 \quad (4.13)$$

To find the term  $\frac{\partial y_j(n)}{\partial u_j(n)}$ , we have to differentiate (4.6) with respect to  $u_j(n)$ .

That is,

$$\frac{\partial y_j(n)}{\partial u_j(n)} = f'_j(u_j(n)) \quad (4.14)$$

Finally the last term (i.e.  $\frac{\partial u_j(n)}{\partial w_{ji}(n)}$ ) on the RHS of (4.11) can be computed by differentiating (4.5) with respect to  $w_{ji}(n)$  and is given by

$$\frac{\partial u_j(n)}{\partial w_{ji}(n)} = y_i(n) \quad (4.15)$$

Now (4.11) becomes

$$\frac{\partial E(n)}{\partial w_{ji}(n)} = -e_j(n) f'_j(u_j(n)) y_i(n) \quad (4.16)$$

The correction  $\Delta w_{ji}(n)$  applied to  $w_{ji}(n)$  can now be defined as

$$\Delta w_{ji}(n) = -\eta \frac{\partial E(n)}{\partial w_{ji}(n)} \quad (4.17)$$

where  $\eta$  is the learning rate which is a factor deciding how fast the weights are allowed to change for each time step. The minus sign indicates that the weights are to be changed in such a way that the error decreases.

Substituting (4.16) into (4.17) yields

$$\Delta w_{ji}(n) = \eta \delta_j(n) y_i(n) \quad (4.18)$$

where the *local gradient*  $\delta_j(n)$  is defined by

$$\begin{aligned} \delta_j(n) &= -\frac{\partial E(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial u_j(n)} \\ &= e_j(n) f'_j(u_j(n)) \end{aligned} \quad (4.19)$$

which shows that the local gradient  $\delta_j(n)$  is the product of the corresponding error signal  $e_j(n)$  and the derivative  $f'_j(u_j(n))$  of the associated activation function.

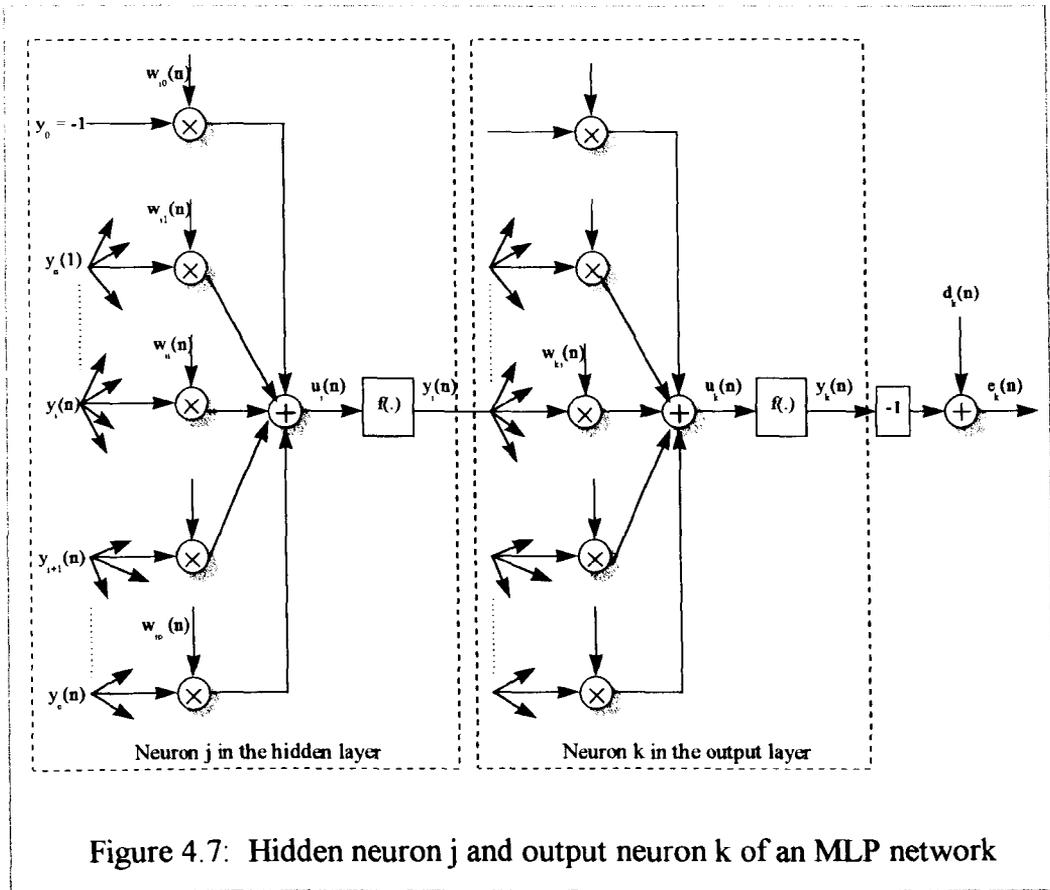
The above derivation is based on the assumption that the neuron  $j$  is located in the output layer of the network. Of course, this is the simplest case. Since neuron  $j$  is in the output layer where desired signal is always available so it is quite straightforward to compute the error signal  $e_j(n)$  and the local gradient  $\delta_j(n)$  by using (4.7) and (4.19) respectively.

Now we shall consider the case in which the neuron  $j$  is not in the output layer of the network but is located in the hidden layer immediately left to the output layer, as shown in Figure 4.7.

Note that now the index  $j$  will refer to hidden layer and the index  $k$  will refer to the output layer. Also note that the desired response  $d_k(n)$  is not directly available to hidden layer neurons.

In this new situation, the local gradient will take the following form:

$$\delta_j(n) = -\frac{\partial E(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial u_j(n)} = -\frac{\partial E(n)}{\partial y_j(n)} f'_j(u_j(n)) \quad (4.20)$$



As neuron  $k$  is located in the output layer

$$\therefore E(n) = \frac{1}{2} \sum_{k \in C} e_k^2(n) \quad (4.21)$$

which is simply (4.9) in which the index  $j$  has been replaced by the index  $k$ . Differentiating this equation with respect to  $y_j(n)$  and using the chain rule, we obtain

$$\frac{\partial E(n)}{\partial y_j(n)} = \sum_k e_k(n) \frac{\partial e_k(n)}{\partial u_k(n)} \frac{\partial u_k(n)}{\partial y_j(n)} \quad (4.22)$$

$$\text{since } e_k(n) = d_k(n) - y_k(n) = d_k(n) - f_k(u_k(n)) \quad (4.23)$$

Therefore,

$$\frac{\partial e_k(n)}{\partial u_k(n)} = -f'_k(u_k(n)) \quad (4.24)$$

The net input of neuron  $k$  is given by

$$u_k(n) = \sum_{j=0}^q w_{kj}(n) y_j(n) \quad (4.25)$$

where  $q$  is the total number of inputs applied to neuron  $k$ . Differentiating both sides of the above equation yields

$$\frac{\partial u_k(n)}{\partial y_j(n)} = w_{kj}(n) \quad (4.26)$$

substituting (4.24) and (4.26) in (4.22) yields

$$\frac{\partial E(n)}{\partial y_j(n)} = -\sum_k e_k(n) f'_k(u_k(n)) w_{kj}(n) = -\sum_k e_k \delta_k(n) w_{kj}(n) \quad (4.27)$$

The local gradient  $\delta_j(n)$  for the hidden neuron  $j$  can now be obtained by using (4.27) in (4.20):

$$\delta_j(n) = f'_j(u_j(n)) \sum_k \delta_k(n) w_{kj}(n) \quad (4.28)$$

This shows that the computation of the local gradient  $\delta_j(n)$  associated with hidden neuron  $j$  requires knowledge of the error signal from the output layer. This means that the error signal propagates back from the output layer towards the input layer via the hidden layer(s). Thus the name *error back-propagation*.

#### 4.6.2 Improved back-propagation:

The back-propagation algorithm derived above has some drawbacks. First of all, the learning parameter  $\eta$  should be chosen to be small to provide minimization of the total error signal. However, for a small  $\eta$  the learning process becomes very slow. On the other hand, large values of  $\eta$  correspond to rapid learning, but lead to parasitic oscillations which prevent the algorithm from converging to the desired solution. Moreover, if the error function contains many local minima, the network might get trapped in some local minimum, or get stuck on a very flat plateau. One simple way to improve the standard back-propagation algorithm is to use adaptive learning rate and momentum as described below:

#### **Momentum:**

Here the idea is to give weights and biases some momentum so that they will not get stuck in local minima, but have enough energy to pass these. Mathematically, adding momentum is expressed as [Rumelhart and McClelland, 1986a, 1986b]:

$$\Delta w_{ji}(n) = \xi \Delta w_{ji}(n-1) + \eta \delta_j(n) y_i(n) \quad (4.29)$$

where  $\xi$  is momentum constant and must have a value between 0 and 1 [Hertz et al., 1991; Haykin, 1994]. If  $\xi$  is 0, the algorithm is same as the basic back-propagation learning rule, that means no momentum (compare (4.18) with (4.29)).  $\xi$  equal to 1 means that the weights change exactly as they did in the preceding time step. A typical value of  $\xi$  is 0.9 - 0.95.

### **Adaptive learning rate:**

As discussed above, it is difficult to choose an appropriate value of the learning rate  $\eta$  for a particular application. The optimal value can change during training. Thus, this parameter should be updated as the training phase progresses. That is, the learning rate should be adaptive. One way of doing this is to change the learning rate according to the way in which the error function responded to the last change in weights. If a weight update decreased the error function, the weights probably were changed in the right direction, and  $\eta$  is increased. On the other hand, if the error function was increased, we reduce the value of  $\eta$

### **4.6.3 Back-propagation with Levenberg-Marquardt Algorithm:**

It has been found that the back-propagation algorithm is very slow in many applications even with adaptive learning rate and momentum. Several attempts have therefore been made to improve the training speed of the standard back-propagation algorithm in addition to adaptive learning rate and momentum [Kollias and Anastassiou, 1989; Singhal and Wu, 1989; Tollenaere, 1990; Barnard, 1992; Battitti, 1992; charalambous, 1992].

Recently Hagan and Minhaj [1994] have showed that the training time can significantly be improved if we incorporate the Levenberg-Marquardt (L-M)

algorithm [Levenberg, 1944; Marquardt, 1963] into the back-propagation algorithm. According to Zhou and Si [1998] the L-M incorporation into the back-propagation algorithm not only improves the training time but also provides superior performance in terms of training accuracy and convergence properties. However, a disadvantage of the algorithm is that it is computationally expensive and hence can be unsuitable for large networks. This disadvantage can be overcome by using a reasonably small data set for training.

This algorithm updates the network parameters as follows [see appendix C]:

$$\Delta \mathbf{w} = (\mathbf{J}^T \mathbf{J} + \mu \mathbf{I})^{-1} \mathbf{J}^T \mathbf{e} \quad (4.30)$$

where  $\mathbf{J}$  is the Jacobian matrix of derivatives to each weight,  $\mu$  is a scalar and  $\mathbf{e}$  is an error vector.

The variable  $\mu$  determines whether learning progresses according to the Gauss-Newton method or gradient descent. If  $\mu$  is large, the  $\mathbf{J}^T \mathbf{J}$  term becomes negligible and the learning progresses according to  $\mu^{-1} \mathbf{J}^T \mathbf{e}$  which approximates to gradient descent (similar to equation (4.17)). While if  $\mu$  is small, (4.30) becomes the Gauss-Newton method. When a step is taken and the error increases,  $\mu$  is increased until a step can be taken without increasing error. However, if  $\mu$  becomes too large no learning takes place (i.e.  $\mu^{-1} \mathbf{J}^T \mathbf{e} \rightarrow 0$ ). This occurs when an error minimum has been found, and is why learning stops when  $\mu$  reaches its maximum value. Further details regarding the incorporation of the L-M algorithm into the back-propagation algorithm can be found in [Hagan and Minhaj, 1994].

#### 4.6.4 Approximation Capabilities of MLP Networks:

A number of researchers [Cybenko 1989; Hornik et al., 1989; Funahashi, 1989] have proved mathematically that a single hidden layer feedforward neural network is capable to approximate any continuous multivariable function to any desired degree of accuracy, provided that sufficiently many hidden layer neurons are available. Cybenko's proof is particularly interesting as it is mathematically concise and elegant. This theorem can be stated as,

*“Let  $f(\cdot)$  be a nonconstant, bounded, and monotone-increasing continuous function. Let  $H_p$  denote the  $p$ -dimensional unit hypercube  $[0, 1]^p$ . The space of continuous functions on  $H_p$  is denoted by  $S(H_p)$ . Then, given any function  $g \in S(H_p)$  and  $\varepsilon > 0$ , there exists an integer  $I$  and set of real constants  $\alpha_i$ ,  $\theta_i$ , and  $w_{ij}$ , where  $i = 1, \dots, I$  and  $j = 1, \dots, p$  such that we may define*

$$G(x_1, \dots, x_p) = \sum_{i=1}^I \alpha_i f\left(\sum_{j=1}^p w_{ij} x_j - \theta_i\right) \quad (4.31)$$

*as an approximation realization of the function  $g(\cdot)$ ; that is,*

$$\left|G(x_1, \dots, x_p) - g(x_1, \dots, x_p)\right| < \varepsilon$$

*for all  $\{x_1, \dots, x_p\} \in H_p$ ”*

The MLP networks use sigmoidal non-linearity as their activation function which has same properties as has the function  $f(\cdot)$  described in the theorem. This means the theorem is directly applicable to MLP networks. The theorems of Hornik et al. [1989] and Funahashi [1989] are also applicable to the MLP networks.

Hornik et al. [1990] also proved another important result relating to the approximation capability of MLP networks employing sigmoidal hidden unit activations. They showed that these networks can not only approximate an unknown function but also its derivative. In fact, Hornik et al. [1990] also showed that these networks can approximate functions that are not differentiable in the classic sense but possess a generalized derivative, as in the case of piecewise differentiable functions.

Light [1992] extended Cybenko's results to continuous function on  $\mathbb{R}^n$  and showed that integer weights and biases are sufficient for accurate approximation. In other version of the theorem Light shows that the sigmoid can be replaced by any continuous function. The universality of single hidden layer feedforward networks having nonsigmoidal activation functions was formally proved by Stinchcombe and White [1989].

The above results are very promising and provide great comfort to researchers in reinforcing their beliefs about the capabilities of MLP networks. These, however, guarantee only the existence of an approximating network and do not give any clues about how to construct one. The issue of choosing an appropriate number of neurons in a hidden layer of an MLP network is almost unresolved. With few hidden neurons, the network may not produce outputs reasonably close to the targets. This effect is called *underfitting*. On the other hand, an excessive number of hidden layer neurons will increase the training time and may cause problem called *overfitting*. The network will have so much information processing capability that it will learn insignificant aspects of the training set, aspects that are irrelevant to the general population. If the performance of the network is evaluated with the training set, it will be excellent. However, when the network is called upon to work with the general population, it will do poorly. This is because it will consider trivial features unique to training set members, as well as important

general features, and become confused. Thus it is very important to choose an appropriate number of hidden layer neurons for satisfactory performance of the network.

A number of rough guidelines have been proposed to choose a suitable number of hidden layer neurons in a three layer MLP network. For example Lippmann [1987] has provided geometrical arguments and reasoning to justify why the number of neurons in the hidden layer of a three layer MLP network should be  $Q(P+1)$ , where  $Q$  is the number of output units and  $P$  is the number of input units. Another rough guideline for choosing the number of hidden neurons is the *geometric pyramid rule*. This rule states that, for many practical networks, the number of neurons follows a pyramid shape (see Figure 4.5), with the number decreasing from the input towards the output. The number of neurons in each layer follows a geometric progression. Thus, in a three layer network with  $P$  inputs and  $Q$  outputs, the number of neurons in the hidden layer should be  $\sqrt{PQ}$ .

The above formulas are only rough approximations to the ideal hidden layer size and may be far from optimal in certain applications. For example, if the problem is complex but there are only few inputs and outputs, we may need many more hidden neurons than suggested by the above formulae. On the other hand, if problem is simple with many inputs and outputs, fewer neurons will often suffice.

A common approach is to start with a small number of hidden neurons (e.g. with just two hidden neurons). Then slightly increase the number of hidden neurons, again train and test the network. Continue this procedure until satisfactory performance is achieved. This procedure is time consuming, but usually results in success.

## 4.7 Radial Basis Function Networks:

The Radial basis function network (RBF) is a powerful alternative to the MLP network. The basic idea of RBFs was originally proposed by Bashkirov et al. [1964] and the theoretical properties were developed by Aizerman et al. [1964a, 1964b]. The RBF networks were originally applied to the multivariable interpolation problem [Powell, 1985] and were first formulated as neural networks by Broomhead and Lowe [1988]. Experiments of Moody and Darkin [1989] who applied RBF networks to predict chaotic time series, further popularized these networks. Poggio and Girosi [1990] have shown how regularization theory can be applied to this class of networks for improving generalization.

During the past eight years these networks have proved to be an area of active research within the neural network community and thus have found many applications in areas as different as image processing [Saha et al. 1991], Speech recognition [Ng and Lippmann, 1991], Bioengineering [Donaldson et al., 1995] and modelling and control of dynamical systems [Barnes et al., 1991; Hartman and Keeler, 1991; Chen and Billings, 1992; Hotland et al., 1992; Hunt et al. 1992; Röscheisen et al., 1992; Sbarbaro, 1992; Pantaleón-Prieto et al., 1993; Pottman and Jorgl, 1993; Elanayar and Shin, 1994; Erives et al. 1996; Gorinevsky et al., 1996; Fathala and Farsi, 1997]. The increasing popularity of RBF networks is because of their distinctive properties of best approximation, simple network structure and efficient learning procedure. The only disadvantage is that they require more neurons than MLP networks for comparable performance levels.

The topology of an RBF network is similar to that of an MLP network. An RBF network is essentially a three layer feedforward neural network. The first layer, consists of a number of units clamped to the input vector. The hidden

layer is composed of units, each having an overall response function (activation function), usually a Gaussian as defined below:

$$g_k(\vec{x}) = \exp\left(-\frac{\|\vec{x} - \vec{c}_k\|^2}{\sigma_k^2}\right) \quad (4.32)$$

where  $\vec{x}$  is the input vector,  $\vec{c}_k$  is the centre of the  $k$ th RBF and  $\sigma_k^2$  is its variance. The centres can be either fixed before training of the network or learned through the training of the network. The third layer computes the output function for each class as follows:

$$f(\vec{x}) = \sum_{k=1}^M w_k \cdot g_k(\vec{x}) \quad (4.33)$$

where  $M$  is the number of RBFs and  $w_k$  is the weight associated with  $k$ th RBF.

Despite topological similarity with MLP networks, the RBF networks, differ from MLP networks in several important respects. These differences are outlined below [Haykin, 1994]:

- An RBF network (in most applications) is a single hidden layer neural network, whereas, an MLP network may consist of one or more hidden layers.
- All individual neurons in a hidden layer and in an output layer of an MLP network share a common neuron model. On the other hand, the neurons in the hidden layer of an RBF network are quite different and serve a different purpose from those in the output layer of the network.
- The activation function of each neuron in a hidden layer of an MLP network computes the inner product of the input vector and the synaptic

weight vector of that unit. On the other hand, the argument of the activation function of each hidden unit in an RBF network computes the Euclidean norm (distance) between the input vector and the centre of that unit.

- The hidden unit of an RBF network is nonlinear and the output unit is always linear. The hidden unit of an MLP network is also nonlinear, however, the output layer can be linear or nonlinear.
- MLPs construct global approximations to nonlinear input-output mapping and are therefore capable of generalization in regions of the input space where little or no training data are available. On the other hand, RBF networks construct local approximations to nonlinear input-output mappings and are therefore capable of fast learning and reduced sensitivity to the order of presentation of training data.

#### **4.7.1 Training:**

A number of approaches to training RBF networks are available in the literature. Most of these can be divided into two stages. The first stage involves the determination of an appropriate set of RBF centres and widths and the second stage deals with the determination of the connection weights from the hidden layer to the output layer. Indeed, the selection of the RBF centres is the most crucial problem in designing the RBF network. These should be located according to the demands of the system to be modelled. A number of different methods have been proposed for the selection of appropriate RBF centres. Here, we shall describe the orthogonal least squares (OLS) method developed by Chen et al. [1991].

#### 4.7.1.1 Orthogonal Least Squares Method:

An RBF network can be considered as a special case of the linear regression model:

$$d(t) = \sum_{i=1}^M p_i(t)\theta_i + e(t) \quad (4.34)$$

where  $d(t)$  is the desired output,  $p_i(t)$  are the regressors,  $\theta_i$  are the parameters to be estimated and  $e(t)$  is the error. In matrix notation, the above equation can be written as

$$D = P\Theta + E \quad (4.35)$$

where

$$D = [d(1) \ d(2) \ \dots \ d(N)]^T$$

$$\Theta = [\theta_1 \ \theta_2 \ \dots \ \theta_M]^T$$

$$P = [p_1 \ p_2 \ \dots \ p_M], \quad p_i = [p_i(1) \ p_i(2) \ \dots \ p_i(N)]^T, \quad 1 \leq i \leq M$$

$$E = [e(1) \ e(2) \ \dots \ e(N)]^T$$

The regressor vector  $p_i$  forms a set of basis vectors, and the least squares (LM) solution of (4.35) satisfies the condition that the matrix product  $P\Theta$  be the projection of  $D$  on to the space spanned by the basis vectors. This means that the square of the projection  $P\Theta$  is part of the desired output energy that can be counted by the regressors. However, it is not clear how an individual regressor contributes to this output energy, as different regressors are generally correlated.

The OLS method involves the transformation of the set of  $p_i$  into a set of orthogonal basis vectors, and thus makes it possible to calculate the individual contribution to the desired output energy from each basis vector. The regression matrix  $P$  can be decomposed into

$$P = WB \quad (3.36)$$

where  $B$  is an  $M \times M$  upper triangular matrix:

$$B = \begin{bmatrix} 1 & b_{12} & b_{13} & \cdots & b_{1M} \\ 0 & 1 & b_{23} & \cdots & b_{2M} \\ 0 & 0 & \ddots & \ddots & \vdots \\ \vdots & & & & \\ 0 & \cdots & 0 & 1 & b_{M-1M} \\ & & & 0 & 1 \end{bmatrix} \quad (3.37)$$

and  $W$  is an  $N \times M$  matrix with orthogonal columns  $w_i$  such that

$$W^T W = H \quad (3.38)$$

Here  $H$  is a diagonal matrix

$$H = \begin{bmatrix} h_1 & \cdots & 0 \\ \vdots & \ddots & \\ 0 & \cdots & h_M \end{bmatrix} \quad (3.39)$$

$$\text{with } h_i = w_i^T w_i = \sum_{t=1}^N w_i(t) w_i(t), \quad 1 \leq i \leq M. \quad (4.40)$$

Equation (4.35) can be rewritten as

$$D = (PB^{-1})(B\Theta) + E = Wv + E \quad (4.41)$$

where

$$B\Theta = v \quad (4.42)$$

Because  $e(t)$  is uncorrelated with the  $p_i(t)$ , it is straightforward to show that

$$v = H^{-1}W^T D \quad (4.43)$$

or

$$v_i = \frac{w_i^T D}{w_i^T w_i} \quad (4.44)$$

The classical Gram-Schmidt method [Björck, 1967] can be employed to derive  $B$  and  $v$  and thus to solve for  $\Theta$  from (4.42). This method computes one column of  $B$  at a time and orthogonalizes  $P$  as follows: at the  $k$ th step, make the  $k$ th column orthogonal to each of the  $k-1$  previously orthogonalized columns and repeat the operation for  $k = 2, 3, \dots, M$ . The computational procedure can be represented as

$$\begin{aligned} w_1 &= p_1 \\ b_{ik} &= \frac{w_i^T p_k}{w_i^T w_i}, 1 \leq i \leq k \\ w_k &= p_k - \sum_{i=1}^{k-1} b_{ik} w_i \end{aligned} \quad (4.45)$$

In most applications, the number of data points is very large in RBF networks, hence the centres must be chosen as a subset of the data set. In general, the number of all the candidate regressors  $M$ , can also be very large but an adequate model may only require  $M_s$  ( $\ll M$ ) significant regressors. These significant regressors can be selected as follows:

From equation (4.41), the sum of squares of  $D$  is

$$D^T D = \sum_{i=1}^M v_i^2 w_i^T w_i + E^T E \quad (4.46)$$

It is seen that each  $w_i$  explains a proportion of the dependent variable variance. Therefore the *error reduction ratio* due to  $w_i$  is defined as,

$$[\text{err}]_i = \frac{v_i^2 w_i^T w_i}{D^T D}, \quad 1 \leq i \leq M \quad (4.47)$$

This ratio offers a simple and effective means of selecting a subset of significant regressors from a large number of candidates in a forward regression manner. At the  $j$ th step, a regressor is selected if it produces the largest value of  $[\text{err}]_i$  from among the rest of the candidates. The selection procedure is terminated when

$$1 - \sum_{j=1}^{M_s} [\text{err}]_j < \rho \quad (4.48)$$

where  $0 < \rho < 1$  is a chosen tolerance. The parameter estimate  $\Theta_s$  from the resulting subset model is then computed from

$$A_s \Theta_s = v_s \quad (4.49)$$

where  $A_s$  is the  $M_s \times M_s$  unit upper triangular matrix. The detailed selection procedure can be found in [Chen et al., 1989; Billings and Chen, 1989]. The desired tolerance  $\rho$  can actually be learned during the forward regressor procedure so that the regressor procedure becomes an automatic procedure. This aspect is discussed by Billings and Chen [1989].

### 4.7.2 Approximation Capabilities of RBF Networks:

In theory, the RBF network, like the MLP network, is capable of approximating any continuous non-linear mapping [Girossi and Poggio 1989, 1990; Hartman et al., 1990; Poggio and Girossi, 1990; Lee and Kil, 1991; Park and Sandberg, 1991, 1993].

Poggio and Girossi [1990] emphasize that the property of approximating functions arbitrarily well is not sufficient for characterizing good approximation schemes, as many schemes have this property. These authors propose that the key property is not that of arbitrary approximation, but the property of *best approximation*. An approximation scheme is said to have this property if in the set of approximating functions there is one which has the minimum distance from the given function. The first main result of their paper [Poggio and Grossi, 1990] is that MLP networks do not have the best approximation property. Secondly, they prove that RBF networks do have the best approximation property. This result is very significant and provides theoretical support for favouring RBF networks. The result was published some eight years ago but (to the best of our knowledge) has not found any serious challenge from the advocates of MLP networks. This further enhances confidence in RBF networks over MLP networks.

### 4.8 Discussion:

The field of ANNs is a hot topic of research at present and has spread to almost every field of science and engineering. The reasons for the popularity of ANNs are many and include the following:

- ANNs learn by experience rather than by modelling or programming.

- ANN architectures are distributed, inherently parallel and potentially real time.
- They have the ability to generalize.
- They do not require a prior understanding of the process or phenomenon being studied.
- They can form arbitrary continuous nonlinear mappings.
- They are robust to noisy data

Despite these promising features of ANNs, most naval architects and other people related to marine engineering and ship control are still dubious about the application of ANNs, largely because of a lack of understanding of how they work. This Chapter is therefore, especially written for those who have little or no knowledge of ANNs and do not know the functionality and capabilities of neural networks.

The Chapter begins with a brief history of ANNs to provide a source of motivation and inspiration to the reader. A simplified model of a biological (real) neuron is described and it is explained how an artificial neuron model can be derived inspired from its biological counterpart. A single neuron is not very capable of performing complex tasks, however, when several neurons are connected in some specific manner, they can solve complex problems which would be impossible for a single neuron. Several ANN architectures have been proposed. This Chapter describes only the feedforward architecture which has already proved its worth in numerous engineering applications, particularly in the field of systems and control.

Feedforward neural networks can be divided into two major classes: the MLPs and RBF networks. Both of these classes are described in considerable detail and their learning algorithms, namely the back-propagation learning algorithm and the orthogonal least squares algorithm, are derived. Many

researchers have mathematically proved that a single hidden layer MLP network is capable of universal approximation. However, the question of how many neurons should be used in hidden layer for a given problem, is still almost unresolved. The radial basis function networks not only possess a universal approximation property but also the best approximation property. There is no problem of choosing a suitable number of hidden layer neurons for a particular application. The learning process of RBF networks is faster than that of MLP networks, however, they usually require more hidden layer neurons for a given problem. We shall investigate the potential of both of these classes of feedforward neural networks for ship steering control systems in the later Chapters.

## *Chapter 5*

### **DEVELOPMENT OF MULTILAYER PERCEPTRON NETWORKS**

In this Chapter we investigate the potential of multilayer perceptron (MLP) networks for ship steering control systems. The procedure involved in the development of these networks is described and simulation studies are undertaken to test the performance of the networks over a range of forward speeds. The Chapter also explains how the training time of these networks may be improved.

#### **5.1 Methodology:**

Multilayer perceptron networks are trained using the back-propagation learning algorithm of Section 4.6.1. This algorithm is based on the supervised learning of artificial neural networks (ANNs). In such an approach an ANN is trained to behave like a specific form of a conventional controller. Input and target data are generated from the input and output of that controller in a normal closed loop fashion in conjunction with the plant.

In the particular case of a ship steering control system, a PD/PID controller tuned at a fixed operating condition could be used as one trainer for the network in the training phase, as shown in Figure 5.1. The resulting network will learn to behave in the same manner as its trainer, i.e. it will perform well at that particular condition. If we use several conventional controllers (e.g. PD/PID controllers) tuned at different operating conditions as supervisors, the resulting neural network will be able to perform well for the range of operating conditions over which it is trained [Burns, 1995], and

probably slightly outside that range due to the generalization property of ANNs. All ANNs developed in this thesis are based on the above idea. Clearly, this procedure of training ANNs is concerned with simulation only. In principle, the same procedure for training of the ANNs could be applied to data generated from ship trials in which conventional controllers were used with parameters chosen to suit each operating condition and a series of different tests were used to generate the training set.

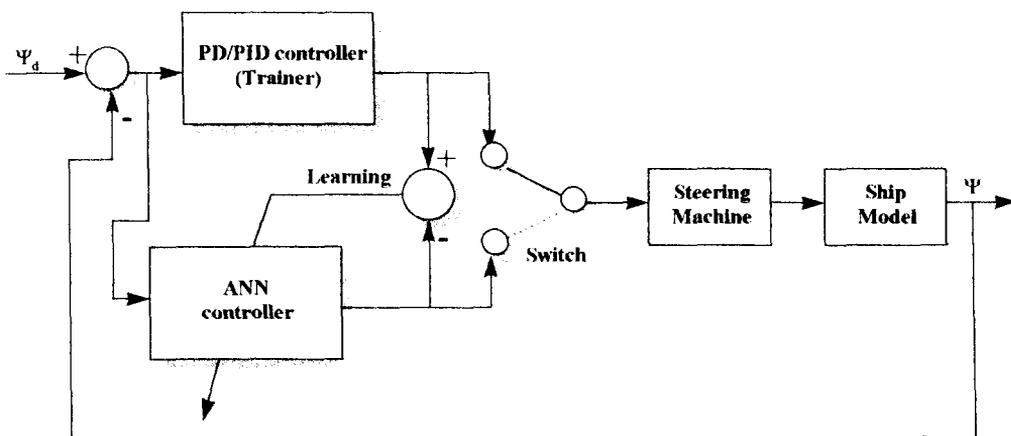


Figure 5.1: Supervised learning of an MLP network for ship steering control system. The ANN output is subtracted from the desired output. In the training phase the network minimizes the squared sum of the resulting error.

An ANN controller developed in this way will have a configuration which will remain fixed once the training phase is complete. The resulting control system will have more predictable characteristics than those which are found in many forms of self adaptive control systems. In particular, stability bounds can be investigated through simulation studies as with any other form of controller having fixed characteristics.

The selection of an appropriate network structure is crucial for a successful application. As discussed in the previous Chapter, a single hidden layer MLP network is capable of universal approximation and the network

learns faster if we use the hyperbolic tangent function of equation (4.4) as the nonlinear activation function of hidden layer neurons. Hence all networks developed in this Chapter involve only one hidden layer with a hyperbolic tangent function as the activation function of each hidden layer neuron. All networks contain only one neuron in their output layer, as the purpose of a ship autopilot is to generate an appropriate rudder signal which is the only output of the network. A linear activation function has been used as the activation function of the output neuron. The number of inputs, however, depends upon the nature and number of supervisors (conventional controllers) of the network. The number of hidden layer neurons is to be found by some trial and error procedure (see section 4.6.4 for details). The procedure for developing MLP networks is summarized below:

1. Decide the inputs to the MLP network and the number of hidden layer neurons.
2. Generate and record training data.
3. Use the back-propagation learning algorithm to train the network with the generated training data.
4. Check the success of the training. If the performance is poor, increase the size of the network and go back to step3.

## **5.2 Assumptions:**

The MLP autopilots developed in this Chapter are based on the following assumptions. The end user may regard these as limitations.

1. The system will have control over the rudder(s) only.
2. If the ship has more than one rudder, they will be coupled and move as one.
3. The system will be required to function properly only for forward motion. No reverse operation will be required.

4. The system will only be required to achieve a commanded heading, it will not be optimized for path following.
5. The networks are trained off-line by using the supervised learning methods.
6. The forward speed of the ship is the only operating condition that varies.

Most autopilots reported previously are based on the first four assumptions. See for example [Åström, 1980; Källsröm and Åström, 1981; Flower and Sparrius, 1986 etc.].

Based on the above assumptions, we have developed MLP networks for a number of ships. Some of our investigations are presented in the following Sections.

### **5.3 ROV ZEEFAKKEL:**

ROV Zeefakkel is a small ship of length 45 m. In his PhD thesis, Van Amerongen [1982] demonstrates that the motion of this ship can be described adequately by the Norrbinn's nonlinear model of equation (2.30) with the following parameter values:

$$U = 5 \text{ m/s}, T = 31 \text{ s}, K = 0.5 \text{ 1/s}, \alpha_1 = 1 \text{ and } \alpha_3 = 0.4 \text{ s}^2.$$

The maximum rudder angle for this ship is  $\pm 35^\circ$  and the maximum rudder rate is  $\pm 7^\circ/\text{s}$ .

The purpose of this investigation is to develop an MLP network that yields satisfactory performance from 5 m/s to at least 10 m/s for any reference heading from  $\pm 5^\circ$  and  $\pm 50^\circ$ . The control signal should not reach the maximum

value of the rudder angle ( $\pm 35^\circ$ ) and the maximum rudder rate (i.e.  $\pm 7^\circ/\text{s}$ ). The overall simulation set up is based on Figure 3.1.

The procedure of the development of the network is described as follows:

**Generation of data for training:** As our approach is based on the supervised learning of neural networks, hence the data for training (inputs and target of the network) must be available. For this purpose, we designed two PD controllers at a forward speed of 5m/s and 10m/s by using the IMC laws of Section 3.6.2. The PD controllers had the same structure as given in equation (3.11). Data were generated at reference headings of  $\pm 5^\circ$  and  $\pm 50^\circ$  at each speed.

**Inputs and output of the network:** Equation (3.11) suggests that the inputs to an ANN network for training should be (i)  $\Psi_d - \Psi$  (ii)  $\dot{\Psi}$ . As our aim is to develop an MLP network that yields satisfactory performance from 5m/s to 10m/s, hence a speed vector  $U$  should also be used as an additional input to the network. This vector may be regarded as a scheduling variable. The only output is the commanded rudder signal. Hence, the MLP network of this investigation consists of three inputs and one output, as shown in Figure 5.2.

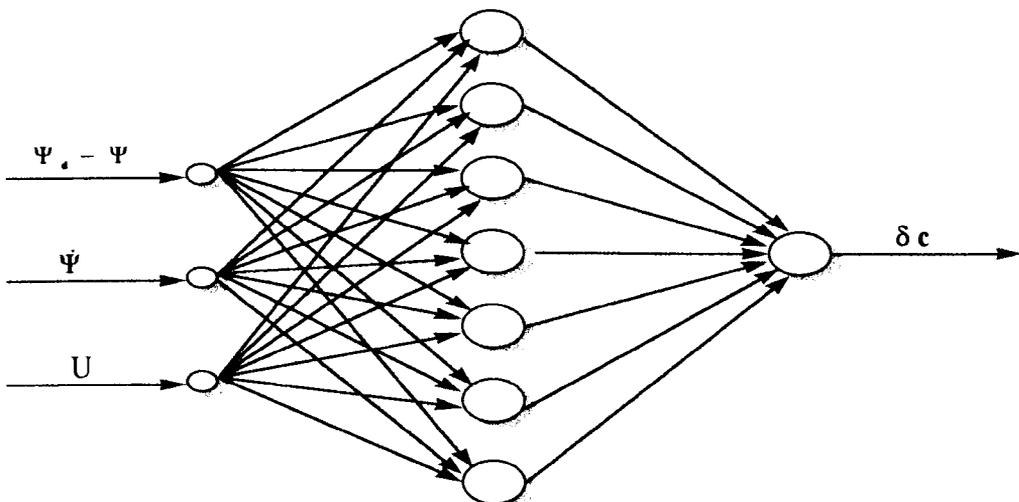


Figure 5.2: MLP network for ROV Zeefakkel

To generate the desired heading response  $\Psi_d$ , a third order reference model of equation (3.5) was used with  $a_m = 0.9341$ ,  $b_m = 0.2040$  and  $c_m = 0.0182$  [Unar and Murray-Smith, 1997a].

**Size of data sets:** 400 time samples were generated at each reference heading at each speed. This means that the total size of data set for each input for training was 3.2 kilo bytes (kb).

**Network Architecture:** As mentioned above, the network used for this investigation ( and all other investigations reported in this Chapter) is a single hidden layer MLP network with hyperbolic tangent nonlinearity in the hidden layer and a linear transfer function in the output layer. The network was trained by using the back-propagation learning algorithm with adaptive learning rate and momentum. After many simulation trials we found that 7 neurons in the hidden layer can provide satisfactory performance (see Figure 5.2).

**Training Time:** The network took 30 minutes 13 seconds in training on a 166MHz Pentium PC and the software used was MATLAB (Version 4.2c) with Neural networks and Control Systems Toolboxes.

The matching of the data with the trained MLP network is shown in Figure 5.3. This Figure clearly indicates that the network has successfully captured the dynamics of its supervisors (i.e. PD controllers) at both speeds (i.e. 5 m/s and 10 m/s).

**Performance:** The performance of the controller for a heading change of  $20^\circ$  at a speed of 5 m/s and 10 m/s is shown in Figure 5.4 and 5.5 respectively.

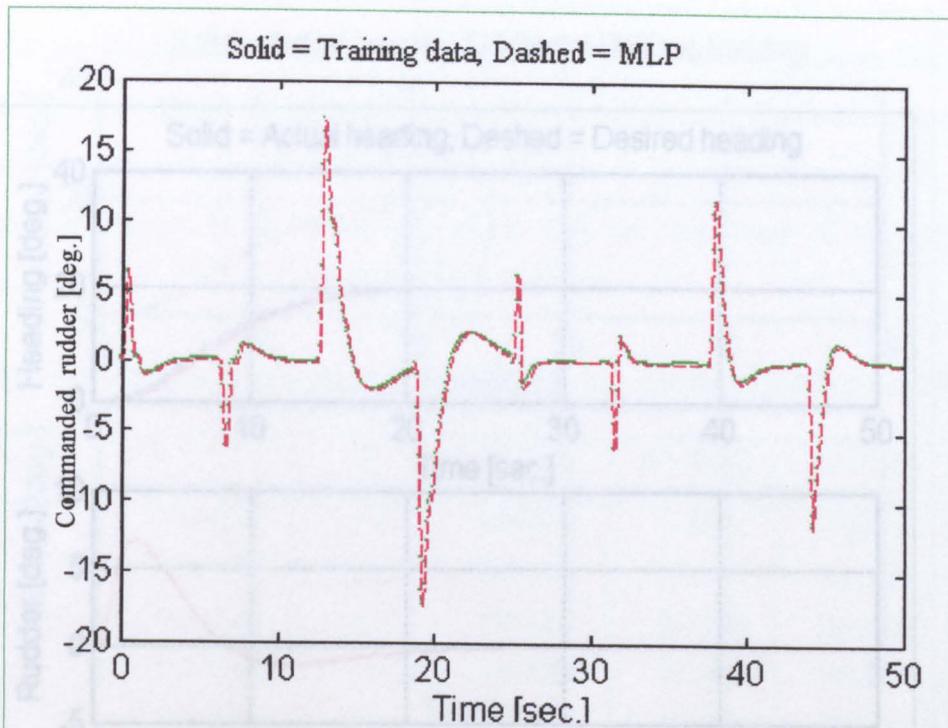


Figure 5.3: Matching with training data, i.e. comparison of the actual and desired output of the MLP network.

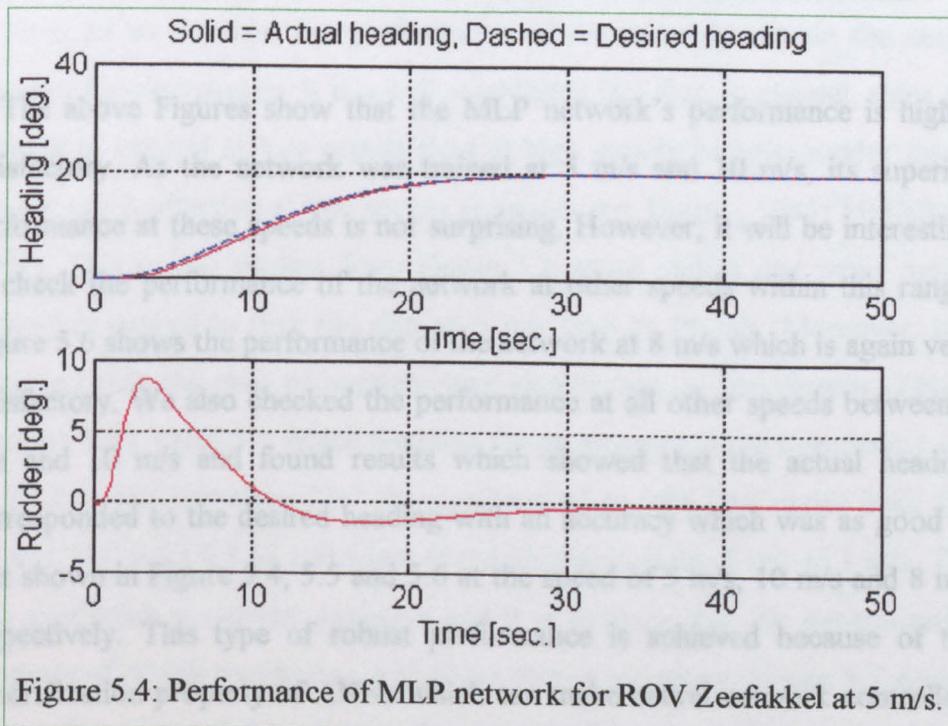
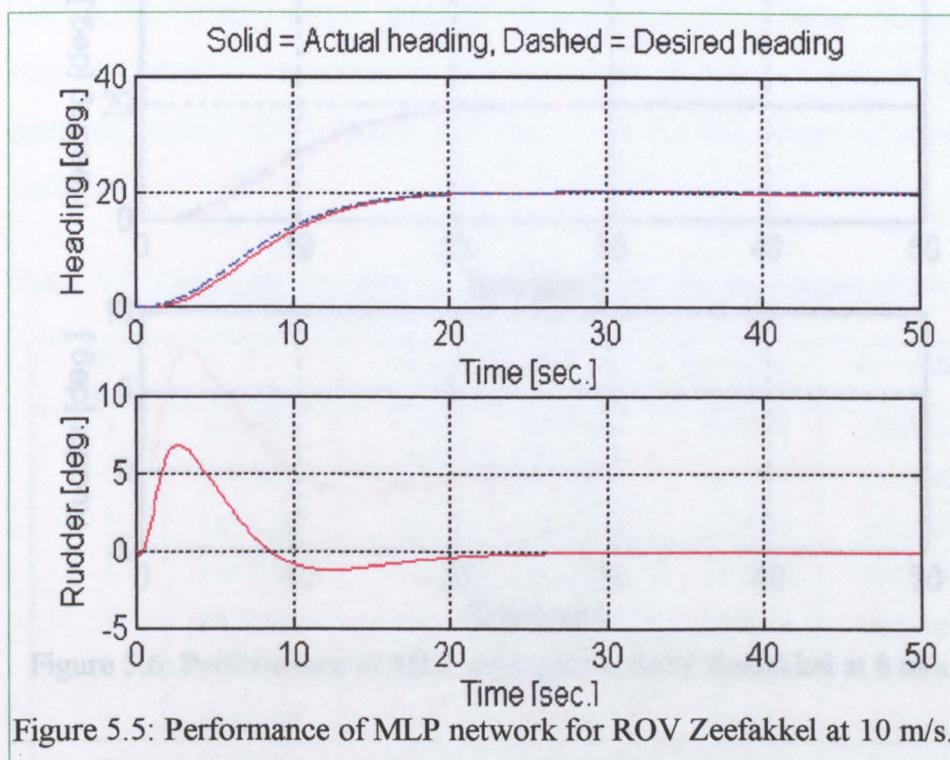


Figure 5.4: Performance of MLP network for ROV Zeefakkel at 5 m/s.



The above Figures show that the MLP network's performance is highly satisfactory. As the network was trained at 5 m/s and 10 m/s, its superior performance at these speeds is not surprising. However, it will be interesting to check the performance of the network at other speeds within this range. Figure 5.6 shows the performance of the network at 8 m/s which is again very satisfactory. We also checked the performance at all other speeds between 5 m/s and 10 m/s and found results which showed that the actual heading corresponded to the desired heading with an accuracy which was as good as that shown in Figure 5.4, 5.5 and 5.6 at the speed of 5 m/s, 10 m/s and 8 m/s respectively. This type of robust performance is achieved because of the generalization property of ANNs which can make neural network controllers superior to other forms of control systems [Bavarian, 1988; Moody, 1991; Hush and Horne, 1993].

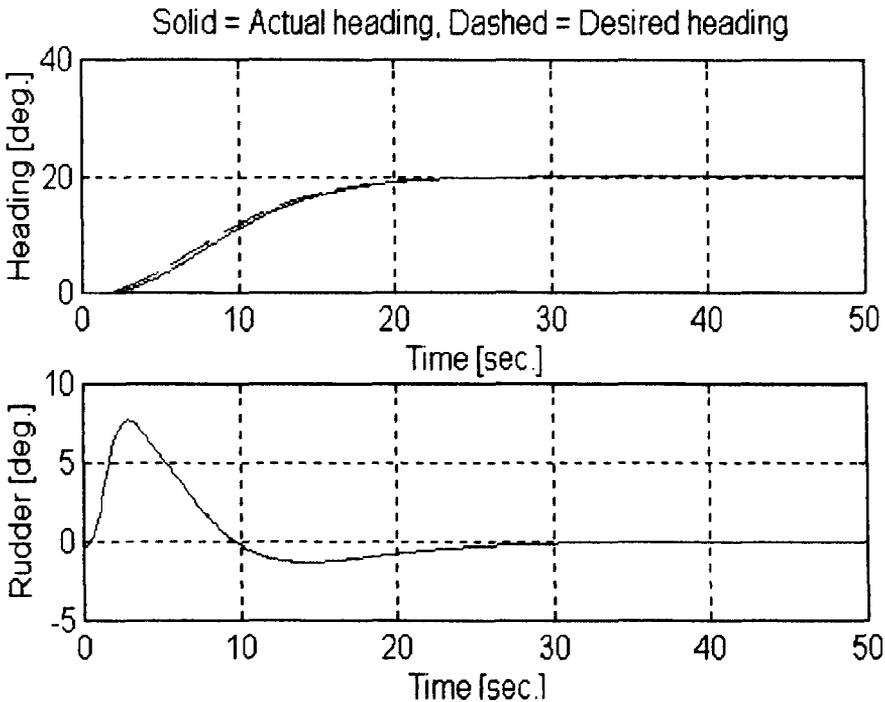


Figure 5.6: Performance of MLP network for ROV Zeefakkel at 8 m/s.

Now let us consider the performance of the network outside the above range of forward speeds. It is a well established fact that, in general, ships are difficult to control at low speeds than at high speeds because rudder forces are not much effective at low speeds [Rawson and Tupper, 1983; McCallum, 1991]. This is also evident from Figure 5.3 which shows that more rudder activity is needed to turn the ship at 5 m/s than at 10 m/s. Moreover, it is observed that the ship parameters vary significantly at low speeds. This is obvious from Table 5.1 which contains the parameters of the ship from 1 m/s to 20 m/s. The parameter “ $m$ ” changes from 62 to 550, “ $d_1$ ” varies from 2 to 10 and “ $d_3$ ” from 0.8 to 100 when speed decreases from 5 m/s to 1 m/s. This is a significant parameter variation and it is unlikely that the generalization property of ANNs could cope with these severe variations. Figure 5.7 depicts the performance of the network at a speed of 3 m/s. Although the ship is stable at this speed, the rudder response is not satisfactory. The ship goes unstable below this speed.

When the speed changes from 10 m/s to 20 m/s, “m” varies from 15.5 to 3.8750, “d<sub>1</sub>” from 1 to 0.5 and “d<sub>3</sub>” from 0.1 to 0.0125. These changes are not significant as compared to the variations at speeds below 5 m/s. A properly trained ANN should generalize well for this range of parameter variation.

Table 5.1: Parameter variations with respect to the forward speed of the ship.

U (m/s)	T	K	m=T/K	d <sub>1</sub> = α <sub>1</sub> /K	d <sub>3</sub> = α <sub>3</sub> /K
1	155.0000	0.1	1550.000	10.0000	100.0000
2	077.5000	0.2	387.5000	05.0000	012.5000
3	051.6667	0.3	172.2222	03.3330	003.7037
4	038.7500	0.4	096.8750	02.5000	001.5625
5	031.0000	0.5	062.0000	02.0000	000.8000
6	025.8333	0.6	043.0556	01.6667	000.4630
7	022.1429	0.7	031.6327	01.4286	000.2915
8	019.3750	0.8	024.2188	01.2500	000.1953
9	017.2222	0.9	019.1358	01.1111	000.1372
10	015.5000	1.0	015.5000	01.0000	000.1000
11	014.0909	1.1	012.8099	00.9091	000.0751
12	012.9167	1.2	010.7639	00.8333	000.0579
13	011.9231	1.3	009.1716	00.7692	000.0455
14	011.0714	1.4	007.9082	00.7143	000.0364
15	010.3333	1.5	006.8889	00.6667	000.0296
16	009.6875	1.6	006.0547	00.6250	000.0244
17	009.1176	1.7	005.3633	00.5882	000.0204
18	008.6111	1.8	004.7840	00.5556	000.0171
19	008.1579	1.9	004.2936	00.5263	000.0146
20	007.7500	2.0	003.8750	00.5000	000.0125

Figure 5.8 illustrates the performance at a speed of 15 m/sec which is quite satisfactory. In fact, we checked the performance of the network at all speeds from 2 m/s to 20 m/s at every reference heading from  $\pm 5^{\circ}$  to  $\pm 60^{\circ}$ . We found

that the performance of the network was robust for a range of speeds from 5 m/s to 20 m/s. However, it was not very satisfactory below 5 m/s as discussed above.

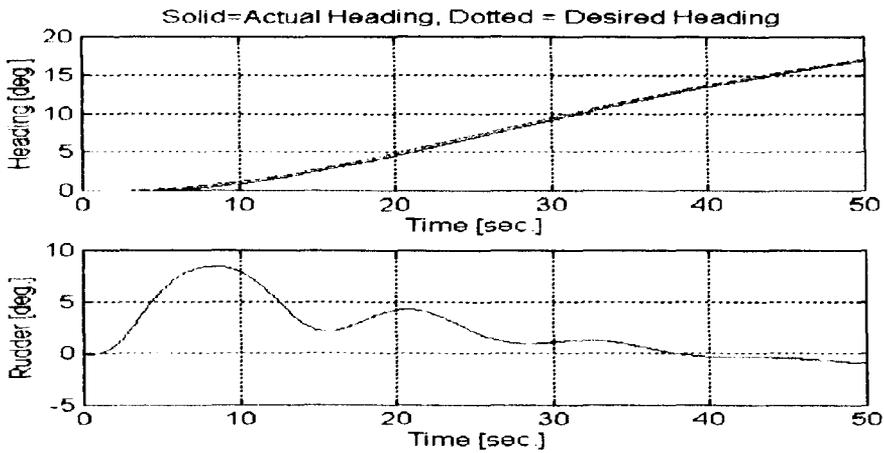


Figure 5.7: Performance of the MLP network at 3 m/s for ROV Zeefakkel.

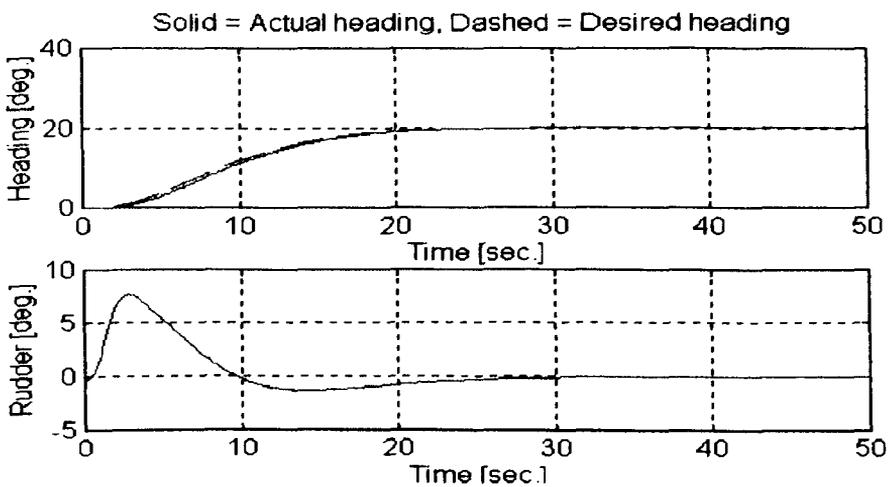


Figure 5.8: Performance of MLP network for ROV Zeefakkel at 15 m/s

## 5.4 Mariner Class Merchant ship:

In the previous Section we developed MLP network for a small ship (ROV Zeefakkell) of length 45m only and found promising results. In this Section, we present results of our investigations on a mariner class merchant ship which is much larger than ROV Zeefakkell. The main parameters of the ship at a speed of 5 m/s are [Layne, 1992; Layne and Passino, 1993]:  $K = -3.19$ ,  $T_1 = 5.71$ ,  $T_2 = 0.37$ ,  $T_3 = 0.89$ ,  $\alpha_1 = \alpha_3 = 1$  (equation (2.30)). The length of the ship is 161m.

In this case, we designed two PID controllers by using the MRAC approach of Section 3.8.1 at 5 m/s and 10 m/s. (See Layne and Passino, 1993, for details). The inputs to the network were the following: (i)  $(\Psi_d - \Psi)$  (ii)  $\dot{\Psi}$  (iii)  $\int_0^t (\Psi_d - \Psi) d\tau$  and (iv)  $U$  (see Figure 5.9).

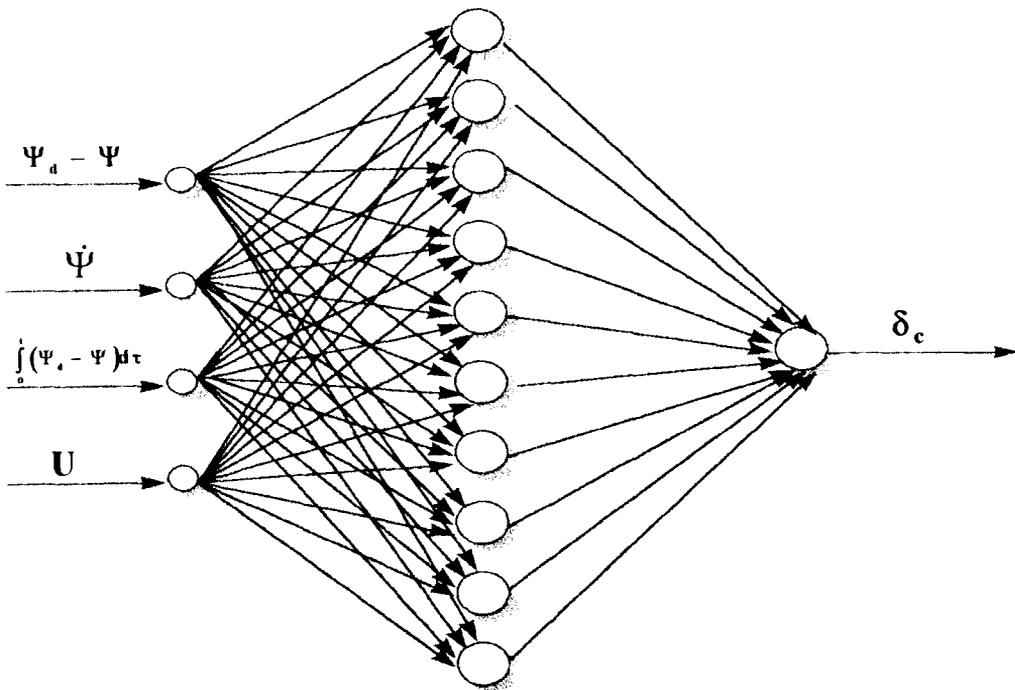


Figure 5.9: MLP network for mariner class merchant ship

The only output was the commanded rudder angle. Data were generated at reference headings of  $\pm 5^\circ$  and  $\pm 50^\circ$ . Five hundred time samples were generated at each speed (5m/s and 10m/s) for each reference heading. The total data size for each input therefore was 4kb. It was found that 10 neurons in the hidden layer are sufficient for satisfactory performance. The training time was 34 minutes 49 seconds.

On the basis of extensive simulation studies we found that the performance of the network is satisfactory for the range of forward speeds from 5 m/s to 10 m/s and even at speeds above 10 m/s. However, the performance was relatively poor at speeds below 5 m/s (i.e. outside the speed range used for training of the network). This is because the ship parameters change significantly at low speeds than at high speeds as is described in the previous Section. Some results are illustrated in Figure 5.10 and 5.11.

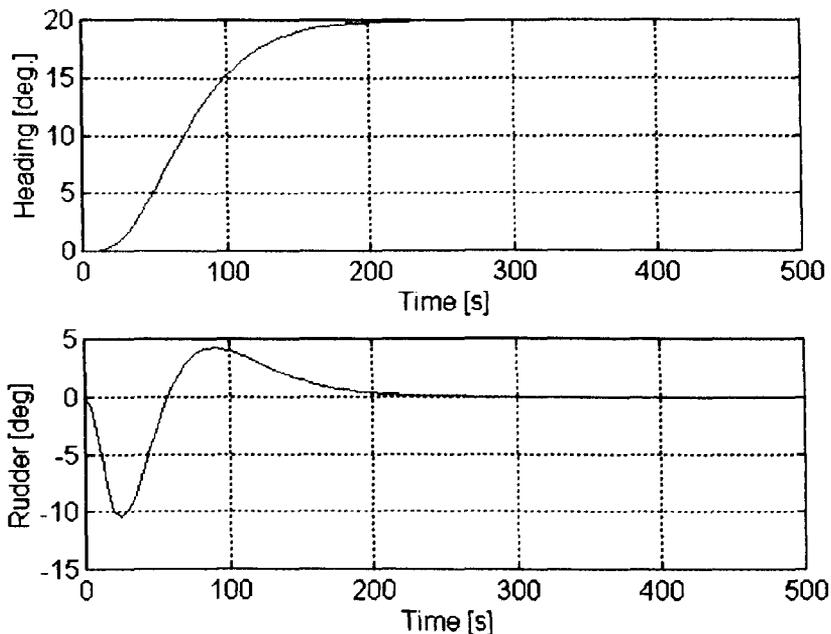


Figure 5.10: Performance of MLP controller for Marine Class Merchant ship at a speed of 5m/s.

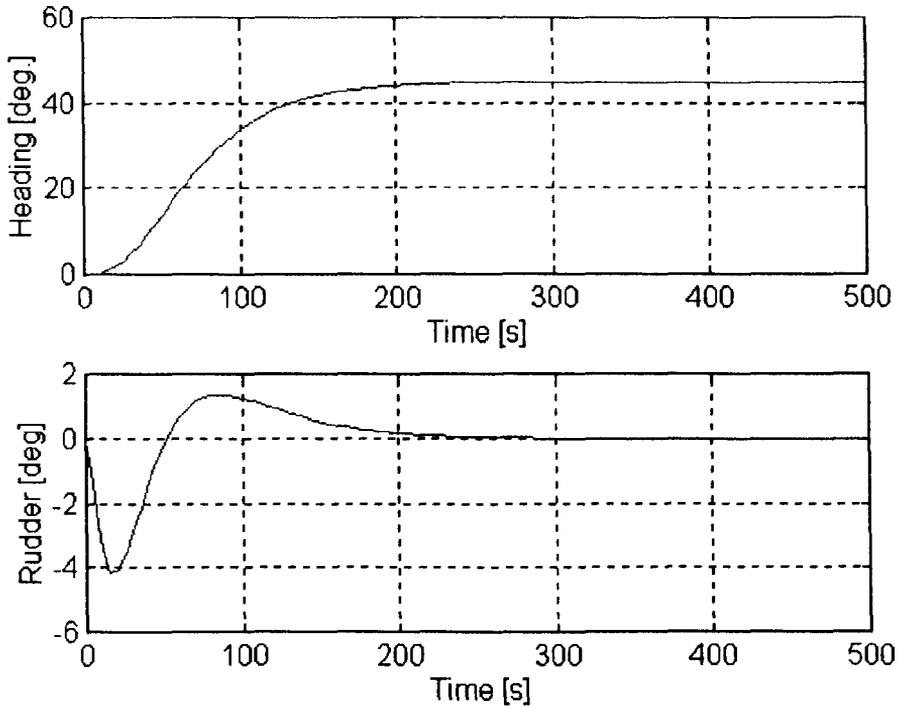


Figure 5.11: Performance of MLP controller for mariner class merchant ship at a speed of 12 m/s.

### 5.5 210,000 DWT Tanker:

This is a large tanker of length 310m. Ekdahl and Henrikson [1970] have investigated the steering characteristics of this tanker in detail and have demonstrated that the motion of the ship can be described by the following set of parameters:

$$T_1 = 1058 \text{ s}, T_2 = 37.8 \text{ s}, T_3 = 84.68 \text{ s} \text{ and } K = -0.0105 \text{ s}^{-1} \text{ at } U = 4.1 \text{ m/s.}$$

The motion of this ship is also investigated by Koyama [1972].

We developed an MLP controller for this tanker by using the similar procedure, as was used for the ROV Zeefakkell and the mariner class

merchant ship of the previous Sections. To generate the desired states, a second order reference model of equation (3.3) with  $\zeta = 1$  and  $w_n = 0.01$  rad/s was used. The performance of the MLP controller is shown in Figure 5.12 and 5.13 at a speed of 6 m/s and 10 m/s respectively. This performance was achieved when 12 neurons were used in the hidden layer of the network. The training time was 18 minutes 4 seconds.

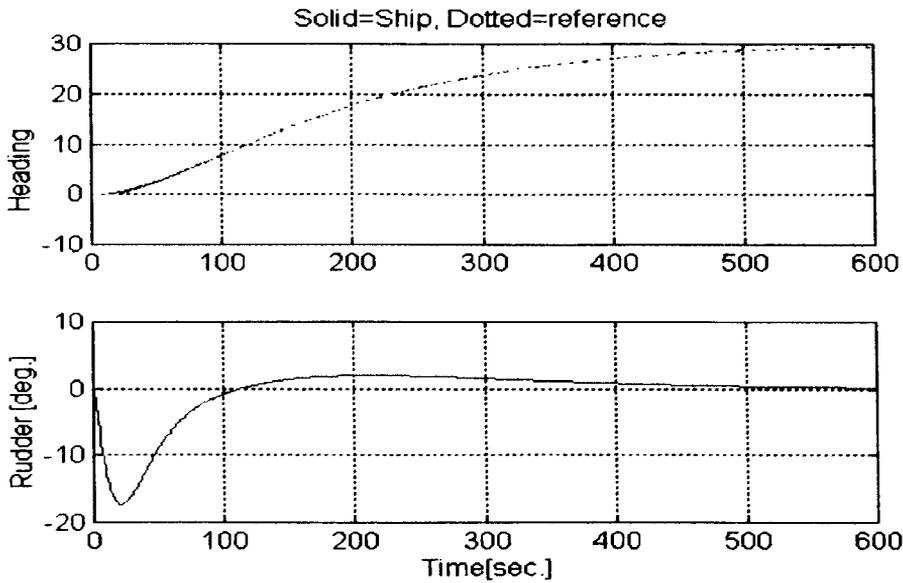


Figure 5.12: Performance of MLP network at 6 m/s for 210,000 dwt tanker

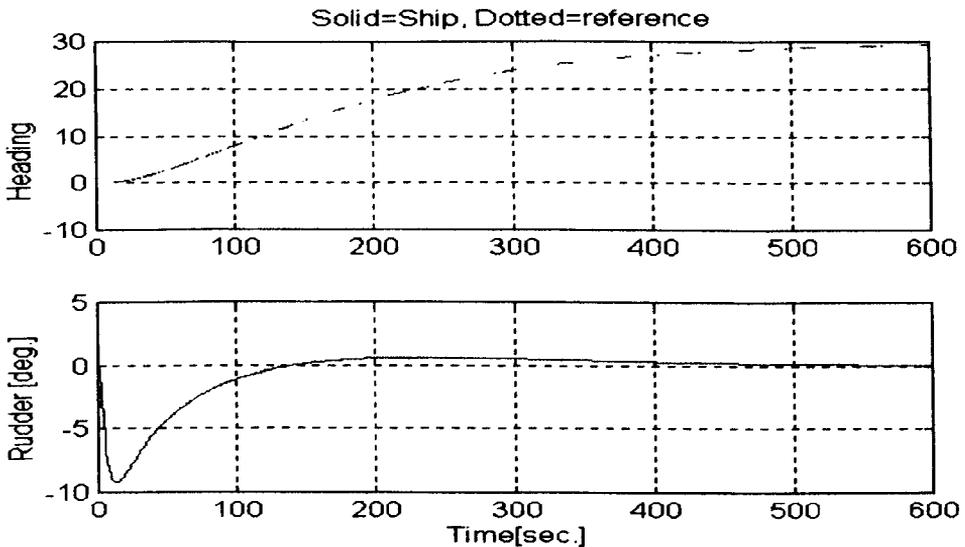


Figure 5.13: Performance of MLP network at 10 m/s for 210,000 dwt tanker

The above investigations suggest that MLP networks have a good potential for ship steering control systems. The networks are computationally cheap and provide robust performance within the range of operating conditions at which they are trained. However, the training process is slow. This disadvantage can be overcome in two possible ways:

- (a) by reducing the size of data sets for training
- (b) by using some faster version of the back-propagation algorithm.

These issues are discussed in the following Sections.

## 5.6 Reducing the size of the training data set

The choice of training data influences very strongly the success of the network since the network is taught to have the same input/output relationship as the training data. The resulting network can not be more accurate than its training data. Saying this does not mean that the training data must represent every possible future input/output relationship. The importance is that the training data must range over most of the values expected in operation. ANNs possess a generalization property which makes it possible to get good results even for inputs for which a network has not been trained. Hence, a wide range in training data is usually of bigger importance than a high density.

Many ships, particularly large tankers can take more than 300 seconds to reach their new steady state path after a turn. For example, the settling time of 210,000 dwt tanker of the previous Section is roughly 500 seconds. To show a complete manoeuvre of this ship to ANN network (in training phase), we have to generate at least 500 time samples (from 0 s to 499 s) at a given

reference heading. If we generate data at four reference headings (e.g. at  $\pm 5^\circ$  and  $\pm 50^\circ$ ), the size of training data set will be  $500 \times 4 = 2000$  time samples at a particular forward speed of the ship. If we generate data at two different speeds (e.g. at 5 m/s and 10 m/s), then the total size of data for each input will become  $2000 \times 2 = 4000$  time samples. This is of course a large data set and an ANN network trained with this data set will take much time in learning. Hence, to improve the speed of training, the size of data should be reduced.

There is also another reason for reducing the size of data sets for training. Due to memory problems (i.e. high computational cost), a number of sophisticated learning algorithms do not work when data sets are very large. For example, the back-propagation rule with L-M algorithm (Section 4.6.3) and the orthogonal least squares algorithm of Section 4.7.1.1. may not work if the data size is 4000 time samples.

Unfortunately, there is no standard method available in the literature to reduce the size of data sets for training. We reduced the data set size by using the following approach [Unar and Murray-Smith, 1997b]:

Suppose that we initially generate 500 time samples at a given reference input. As pointed out earlier, the density of data is not important, hence we may pick, for example, every fifth time sample from the above data set. This reduces the size of data from 500 to 100 time samples. Now if we generate data at four reference headings at two speeds, then the net data size for each input will be 1000 which is just  $\frac{1}{4}$  of the actual data size. The training time of network will be much faster with this reduced size of data than with the original size of data. On the basis of extensive simulation studies, we found that the reduction of data size by this approach has no serious effect on the overall performance of the network.

By using the above procedure of data reduction, we re-trained the networks for all the above ship models and found no serious effect on the performance of the networks when they were trained with a reduced size of data.

As mentioned in Section 5.3, the total size of data for each input of ROV Zeefakkel for training was 3200 time samples. There were three inputs and one output of the network and the training time of the network was 30 minutes 13 seconds. We picked every 4<sup>th</sup> time sample from the data set. That is, 800 time samples of each input were used for training. When we trained the network with this reduced data set, the training time was just 610.3 seconds. This is a significant improvement, as far as training time is concerned. The performance of this network is illustrated in Figure 5.14. If we compare this Figure, with Figure 5.4, we can reveal that the performance is exactly the same.

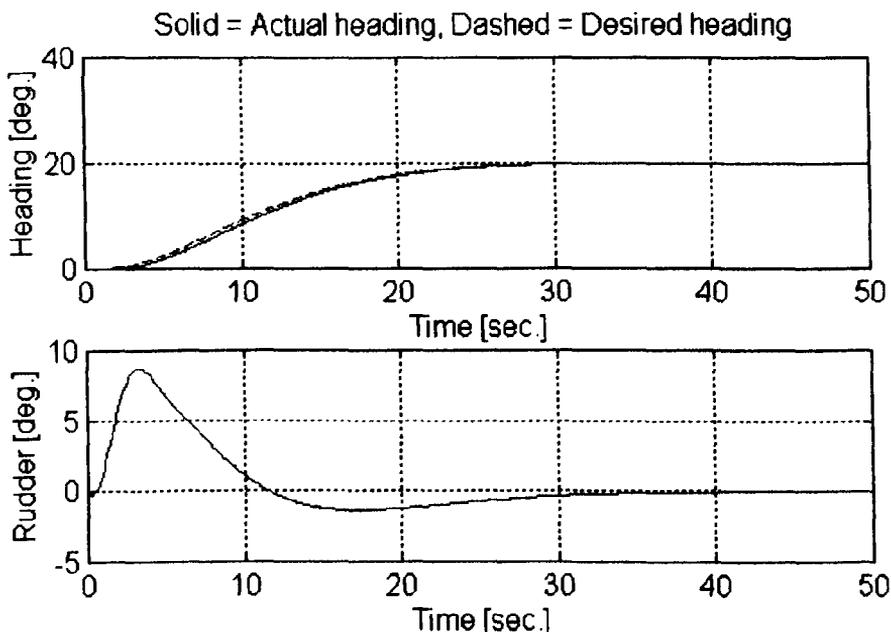


Figure 5.14: Performance of MLP network at 5 m/s for ROV Zeefakkel with reduced data for training.

Similar procedure of data reduction for training an MLP network for the other two ships (i.e. Mariner Class Merchant Ship and 120,000 dwt Tanker) was undertaken and in each case it was found that the performance of the MLP network with full data size and with the reduced data size was almost the same.

### **5.7 Back-Propagation with Levenberg-Marquardt Algorithm:**

In this Section, we demonstrate that the training time of MLP networks can be significantly improved if we incorporate the L-M algorithm into the standard back-propagation rule (See Section 4.6.3). At present, this is the fastest version of the back-propagation algorithm. However, it is computationally expensive and is not suitable if the data size for training is very large [Hagan and Minhaj, 1994].

In the previous Section, we demonstrated that the performance of an MLP network is not affected if we carefully reduce the data size. The L-M modification to the back-propagation algorithm can be useful for training MLP networks for ship steering control applications, if we reduce the data sets by the method outlined in the previous Section.

To check the performance of this version of back-propagation, we again trained an MLP network for ROV Zeefakkell. The training time in this case was only 291 seconds. Figure 5.15 compares the performance of this fastest version of the back-propagation with that achieved by back-propagation algorithm with adaptive learning rate and momentum. Similarly, Figure 5.16 compares the performance of the two versions of the back-propagation algorithm for 210,000 dwt tanker. The training time of the MLP

network trained with adaptive learning rate and momentum was 689 seconds. On the other hand, when the network was trained with back-propagation rule incorporating the L-M algorithm, the training time was reduced to 276 seconds.

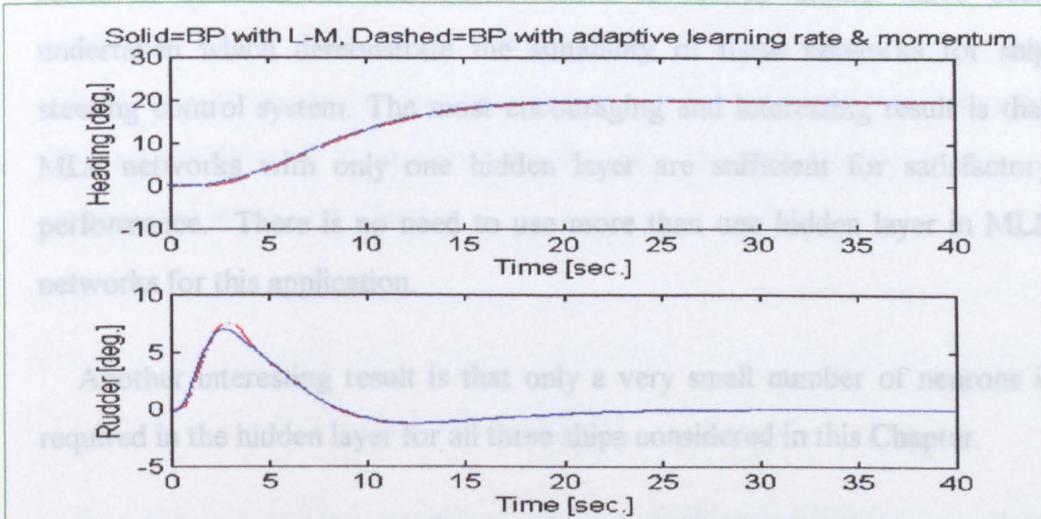


Figure 5.15: Comparison of back-propagation (BP) with adaptive learning rate and momentum and BP with L-M algorithm for ROV Zeefakkel at 10 m/s.

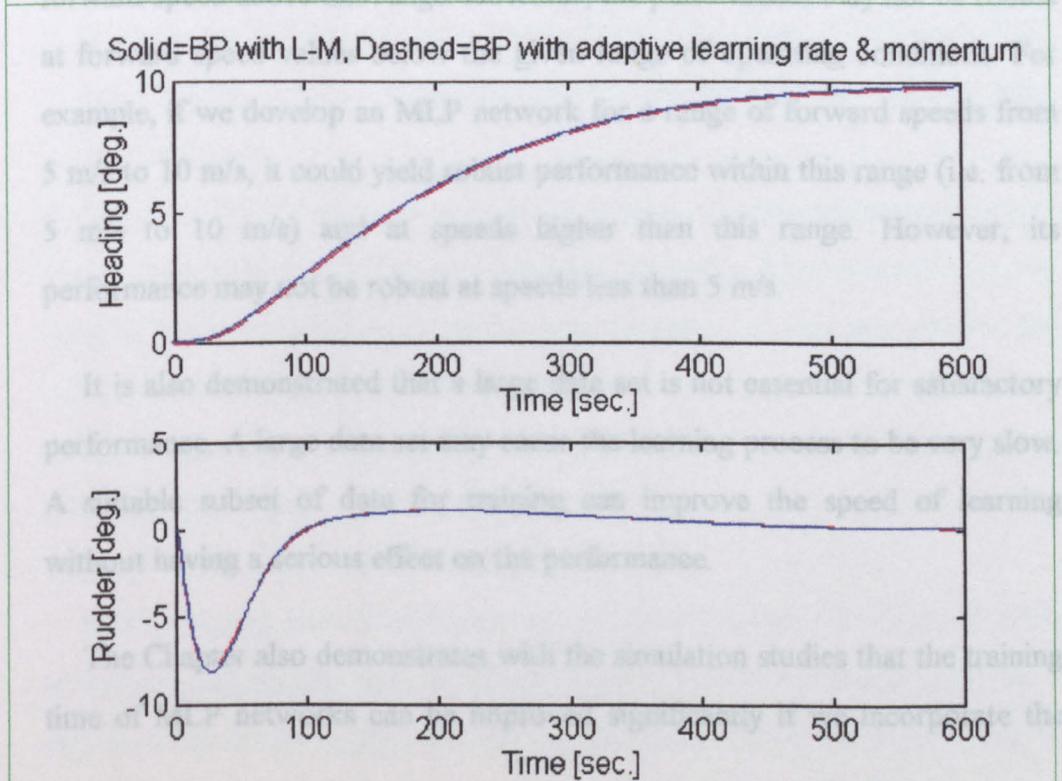


Figure 5.16: Comparison of the back-propagation algorithm with adaptive learning rate and momentum, and the back-propagation with the L-M algorithm for 210,000 dwt tanker.

## 5.8 Conclusions:

This Chapter presents results concerning the use of multilayer perceptron (MLP) networks to obtain a controller which incorporates the properties of a series of conventional controllers. Three simulation studies have been undertaken which demonstrate the suitability of these networks for ship steering control system. The most encouraging and interesting result is that MLP networks with only one hidden layer are sufficient for satisfactory performance. There is no need to use more than one hidden layer in MLP networks for this application.

Another interesting result is that only a very small number of neurons is required in the hidden layer for all three ships considered in this Chapter.

It is demonstrated that MLP networks can yield robust performance within the range of operating conditions they are trained for and even at values of forward speed above this range. However, the performance may not be robust at forward speed values below the given range of operating conditions. For example, if we develop an MLP network for a range of forward speeds from 5 m/s to 10 m/s, it could yield robust performance within this range (i.e. from 5 m/s to 10 m/s) and at speeds higher than this range. However, its performance may not be robust at speeds less than 5 m/s.

It is also demonstrated that a large data set is not essential for satisfactory performance. A large data set may cause the learning process to be very slow. A suitable subset of data for training can improve the speed of learning without having a serious effect on the performance.

The Chapter also demonstrates with the simulation studies that the training time of MLP networks can be improved significantly if we incorporate the

Levenberg-Marquardt theorem into the standard back-propagation learning rule.

A disadvantage of the MLP networks is that there is no straight forward rule to choose an optimal number of hidden layer neurons. This number has to be chosen on some trial and error basis which is time consuming and tedious. This disadvantage can be overcome by using radial basis function networks. The applicability of these networks for ship steering control systems is investigated in the next Chapter.

## *Chapter 6*

### **SHIP STEERING CONTROL USING RBF NETWORKS**

In this Chapter we investigate the applicability of radial basis function (RBF) networks for ship steering control systems. The networks are based on the same assumptions/limitations as discussed in the previous Chapter. However, we also check the performance of the trained networks when the depth of water or loading conditions change along with the forward speed of the vessel. RBF networks offer several advantages over multilayer perceptron (MLP) networks. These include the following:

- As mentioned in Chapter 4 and 5, there is no straight forward rule to find an optimal number of hidden layer neurons of an MLP network for a particular application. There is no such problem in RBF networks. For example, the orthogonal least squares (OLS) algorithm of Section 4.7.1.1 automatically chooses a suitable number of hidden layer neurons from input data sets.
- Although MLP networks possess the universal approximation property they do not have the best approximation property. RBF networks, on the other hand, are not only universal approximators but they also possess the best approximation property (See Section 4.7.2).
- RBF networks are generally faster than MLP networks for a given application.

RBF networks have found many successful applications in different areas of Science and Engineering. However, these networks are investigated for a ship steering control application only by this author [Unar and Murray-Smith,

1997b, 1999; Unar et al. 1998]. Our main investigations are presented in the following Sections:

## 6.1 ROV Zeefakkel

We have already developed MLP network for this ship in Section 5.3 of the previous Chapter. To compare the performance of MLP and RBF network for this ship, we used the same data which were generated for the MLP networks in the previous Chapter. The total size of data for each input was 1000 time samples. We trained an RBF network by using the OLS algorithm. The radial basis function used in the hidden layer was a Gaussian function of width 2. Extensive simulation studies revealed that the performance of RBF network is as good as of MLP network for this ship. Figure 6.1 compares the performance of MLP and RBF network at a speed of 7m/sec for a reference heading of  $20^{\circ}$  and Figure 6.2 compares the performance at a speed of 12 m/sec for a reference input of  $15^{\circ}$ . As can be seen from these Figures, both networks yield exactly same performance. The MLP network was trained with two versions of back-propagation algorithm: (i) Back-propagation with adaptive learning rate and momentum (MLP1) and (ii) Back-propagation with Levenberg-Marquardt algorithm (MLP2). The training times of both MLP1 and MLP2 networks are compared with that of the RBF network in Table 6.1. The number of hidden layer neurons for these networks are also given in the Table. It is obvious from the Table that RBF network is faster than both versions of MLP networks. However, the RBF network requires more neurons in the hidden layer.

Note: In the above Figures, we have compared the performance of MLP1 (part of MLP2) with RBF networks. The performance of MLP1 and MLP2 is almost the same, as was demonstrated in Chapter 5.

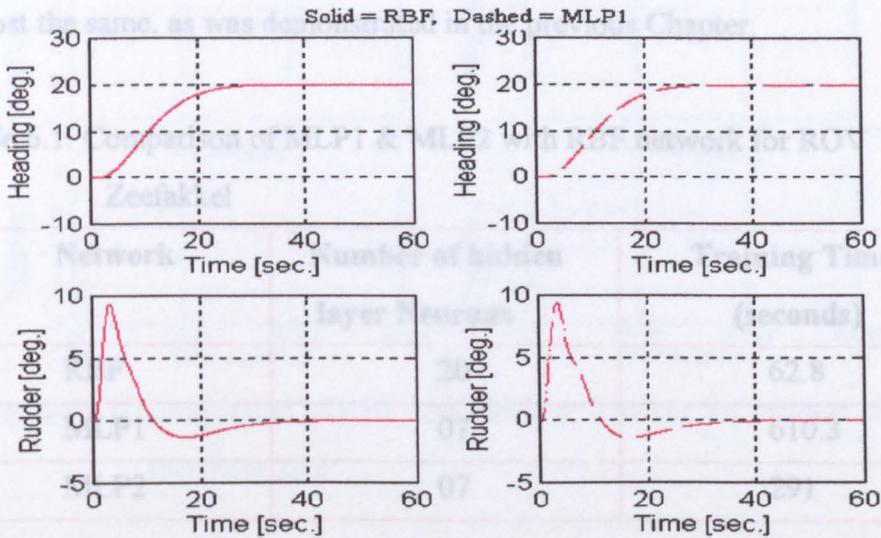


Figure 6.1: Comparison of MLP and RBF network at a speed of 7m/sec. for ROV Zeefakkel.

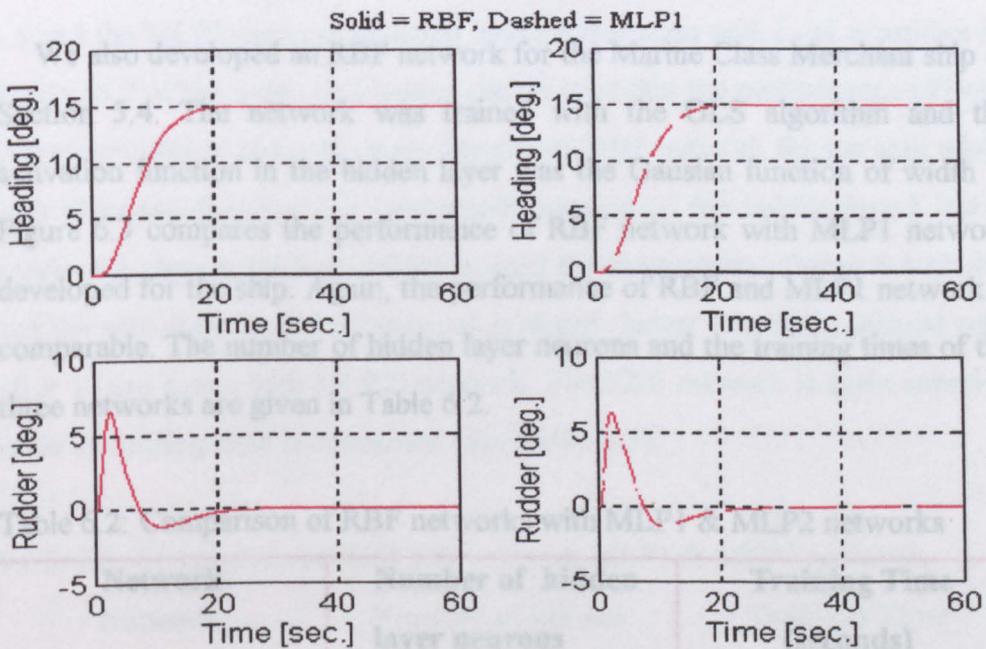


Figure 6.2: Comparison of MLP1 and RBF network at 12 m/sec for ROV Zeefakkel

Note: In the above Figures, we have compared the performance of MLP1 (not of MLP2) with RBF networks. The performance of MLP1 and MLP2 is almost the same, as was demonstrated in the previous Chapter.

Table 6.1: Comparison of MLP1 & MLP2 with RBF network for ROV  
Zeefakkel

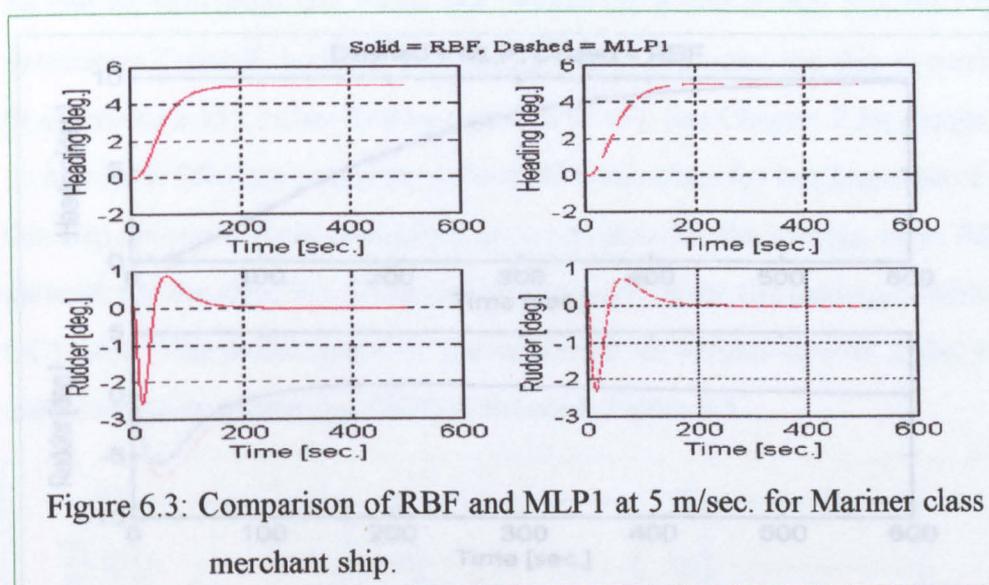
<b>Network</b>	<b>Number of hidden layer Neurons</b>	<b>Training Time (seconds)</b>
RBF	20	62.8
MLP1	07	610.3
MLP2	07	291

## 6.2 Mariner Class Merchant Ship:

We also developed an RBF network for the Marine Class Merchant ship of Section 5.4. The network was trained with the OLS algorithm and the activation function in the hidden layer was the Gaussian function of width 3. Figure 6.3 compares the performance of RBF network with MLP1 network developed for the ship. Again, the performance of RBF and MLP1 network is comparable. The number of hidden layer neurons and the training times of the three networks are given in Table 6.2.

Table 6.2: Comparison of RBF networks with MLP1 & MLP2 networks

<b>Network</b>	<b>Number of hidden layer neurons</b>	<b>Training Time (seconds)</b>
RBF	30	114.6
MLP1	10	749
MLP2	10	210.8



### 6.3 210,000 dwt Tanker

We have already developed an MLP1 network for this ship by using the back-propagation rule with adaptive learning rate and momentum in Section 5.5 and the MLP2 network by using back-propagation with L-M algorithm in Section 5.7 of the previous Chapter and showed that the performance of both of the networks is same. Here we develop an RBF network for the ship when each Gaussian function (i.e. activation function of the hidden layer) has a width of 5. Again, OLS algorithm is used for the training. Figure 6.4 shows that the performance of RBF network is slightly better than that achieved with MLP 1 (and hence with MLP2) network. The RBF network is again superior as far as training time is concerned (See Table 6.3).

Table 6.3: Comparison of RBF network with MLP1 & MLP2 networks

Network	Number of hidden layer Neurons	Training Time (seconds)
RBF	30	122.3
MLP1	12	689
MLP2	12	276

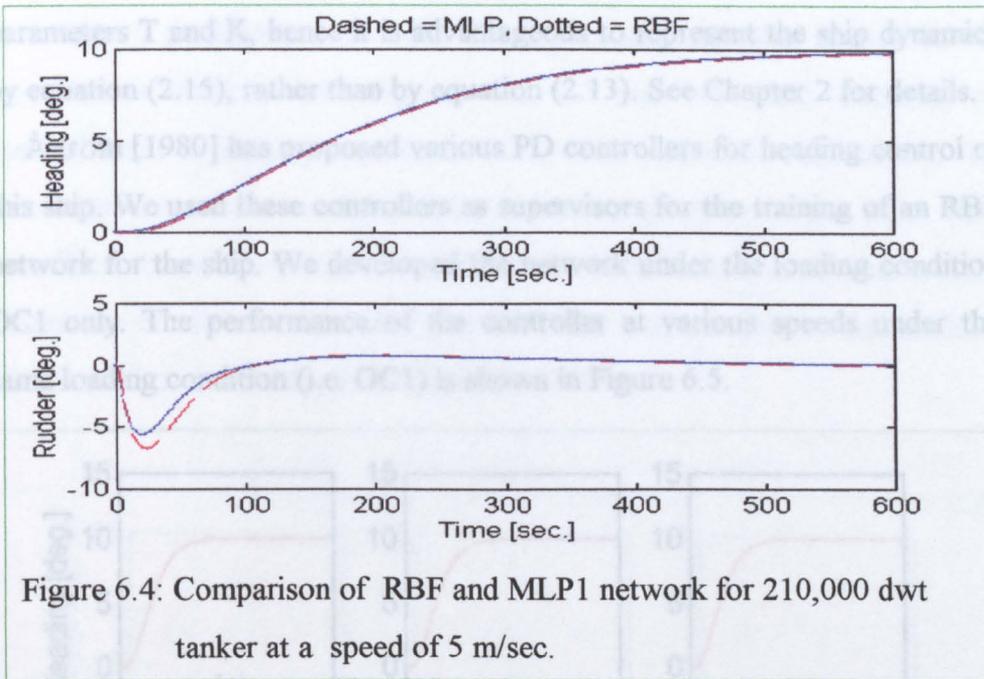


Figure 6.4: Comparison of RBF and MLP1 network for 210,000 dwt tanker at a speed of 5 m/sec.

#### 6.4 Performance of RBF controllers at different loading conditions:

In this Section, we develop an RBF network of a tanker at different forward speeds, and then check the performance of the network when its parameters change with different loading conditions. The parameters of the tanker are given in Table 2.7 of Chapter 2 and are reproduced here in Table 6.4 for the convenience.

Table 6.4: Parameters of a tanker at different loading conditions at 8 m/s

Operating condition	$T_1$	$T_2$	$T_3$	K	T	a	$b \times 10^6$
OC1	80	15	40	-0.013	50	0.020	-260
OC2	160	20	30	-0.040	150	0.007	-270
OC3	1000	25	60	-0.130	1000	0.001	-130
OC4	-300	30	65	0.040	-400	-0.003	-100

As can be seen from the Table, the parameters  $a$  and  $b$  vary less than the parameters  $T$  and  $K$ , hence it is advantageous to represent the ship dynamics by equation (2.15), rather than by equation (2.13). See Chapter 2 for details.

Åström [1980] has proposed various PD controllers for heading control of this ship. We used these controllers as supervisors for the training of an RBF network for the ship. We developed the network under the loading condition OC1 only. The performance of the controller at various speeds under the same loading condition (i.e. OC1) is shown in Figure 6.5.

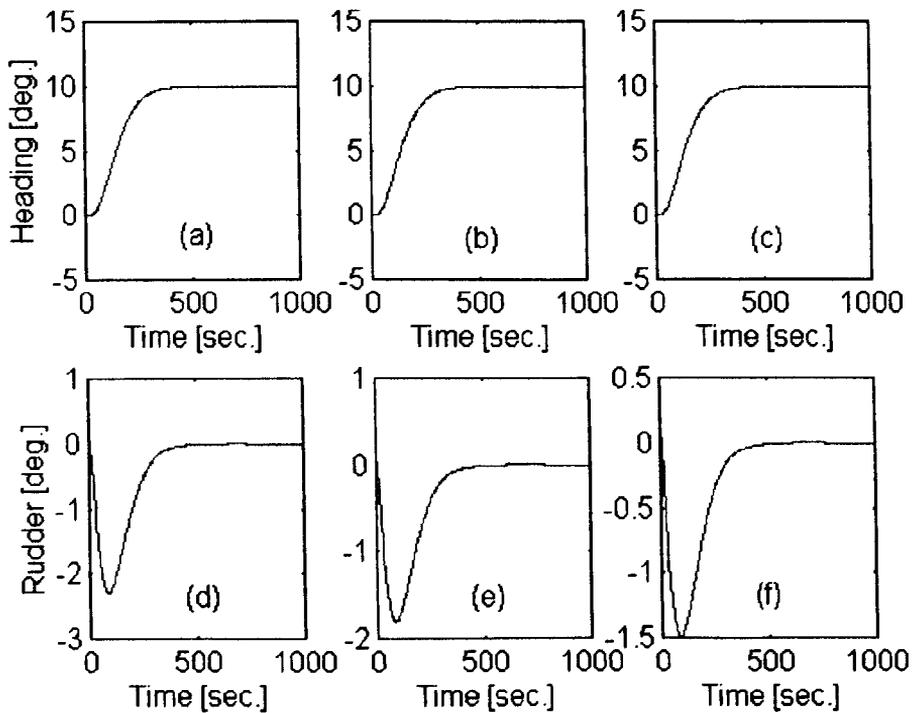


Figure 6.5: Performance of RBF controller at various speeds under loading condition OC1

- (a) Heading response at 8 m/s (d) the corresponding rudder response
- (b) Heading response at 10 m/s (e) the corresponding rudder response
- (c) Heading response at 12 m/s (f) the corresponding rudder response

The satisfactory performance of RBF network at the given loading condition (i.e. OC1) is not surprising, as the controller was developed by using the

parameters at this particular condition. Is the same controller capable to yield satisfactory performance at other loading conditions? The answer to this question is obvious from Figures 6.6, 6.7 and 6.8 which illustrate the performance at loading conditions OC2, OC3 and OC4 respectively. The performance under condition OC4 is not as good as in the other three conditions but it is expected. As can be seen from Table 6.4, the ship is stable under the condition OC1, OC2 and OC3 but it is unstable in the condition OC4 and thus more difficult to control.

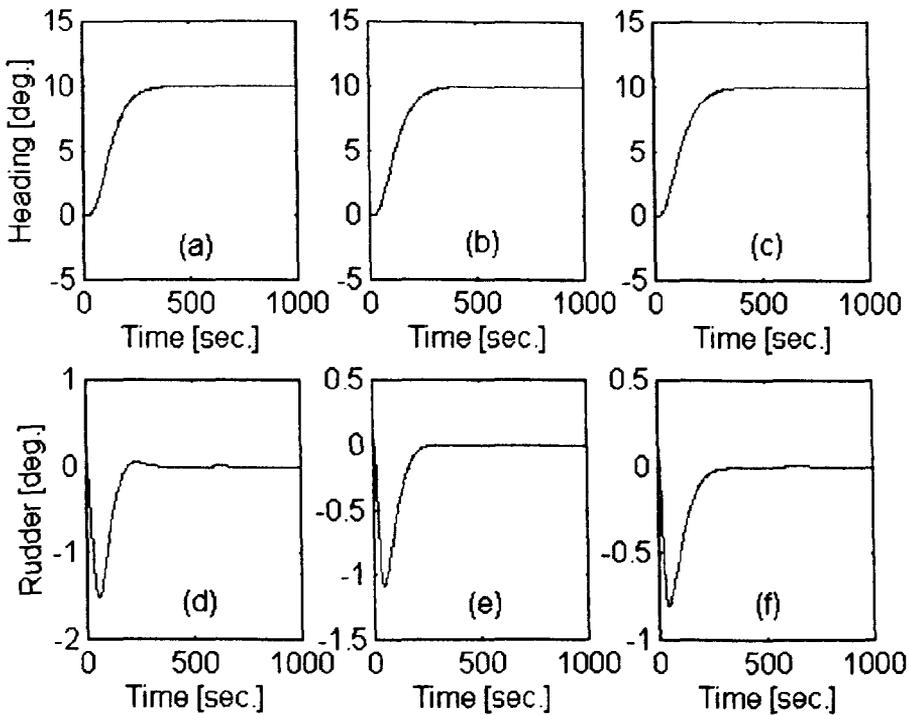


Figure 6.6: Performance of RBF controller at loading condition OC2

- (a) Heading response at 8 m/s (d) the corresponding rudder response
- (b) Heading response at 10 m/s (e) the corresponding rudder response
- (c) Heading response at 12 m/s (f) the corresponding rudder response

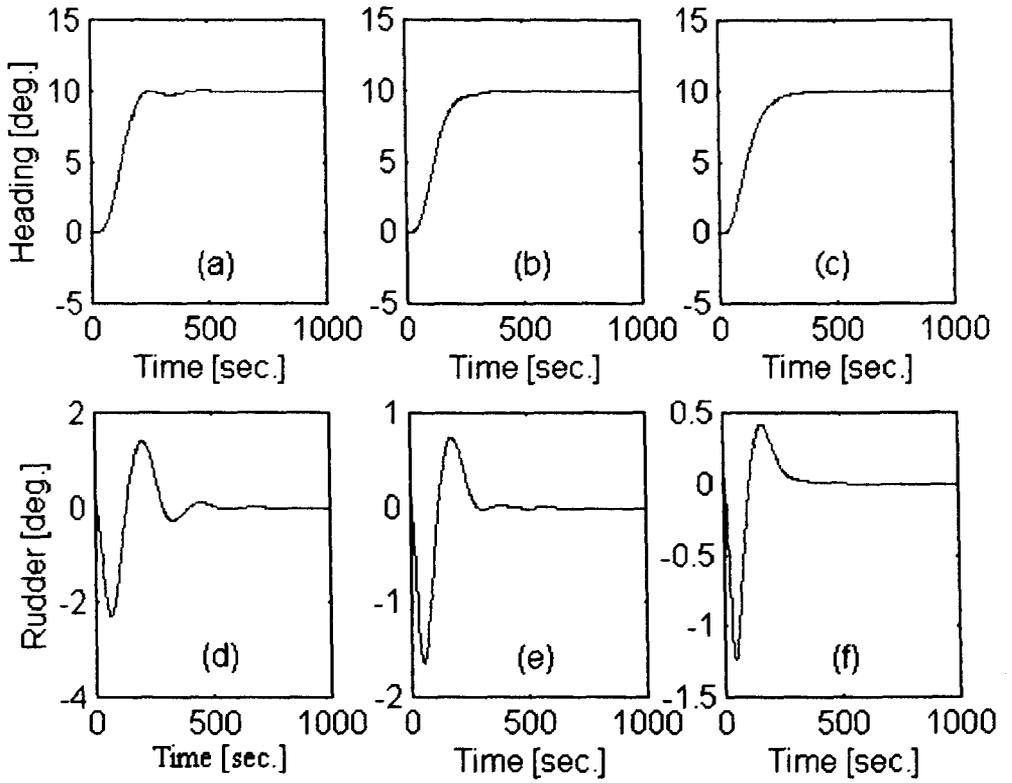


Figure 6.7: Performance of RBF controller at operating condition OC3  
 (a) Heading response at 8 m/s (d) the corresponding rudder response  
 (b) Heading response at 10 m/s (e) the corresponding rudder response  
 (c) Heading response at 12 m/s (f) the corresponding rudder response

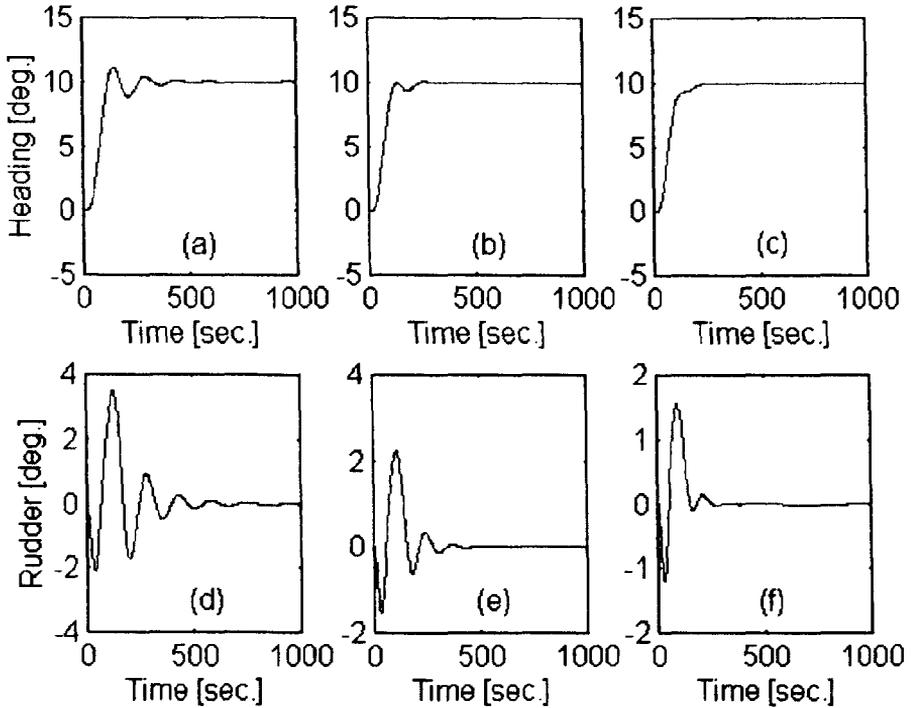


Figure 6.8: Performance of RBF network under loading condition OC4

- (a) Heading response at 8m/s (d) the corresponding rudder response
- (b) Heading response at 10m/s (e) the corresponding rudder response
- (c) Heading response at 12m/s (f) the corresponding rudder response

## 6.5. Performance of RBF controller at different depths of water:

As discussed in Chapter 2, the ship parameters not only change with forward speed of the vessel and the loading conditions but they also change with the depth of water. We have already presented parameters of a mariner type ship at different depth to draft ratios at a speed of 7 knots in Table 2.5 of Chapter 2. Fujino [1968] have also computed experimentally the parameters

of the ship at a speed of 12 knots at different depth to draft ratios. These parameters are given in Table 6.5.

Table 6.5: Parameters of mariner type ship at a speed of 12 knots

H/DT	1.5	1.93	2.5	$\infty$
$T_1$	27.23	70.47	83.45	102.8
$T_2$	09.44	09.39	09.57	08.92
$T_3$	08.68	13.77	17.71	19.51
T	26.94	66.07	75.32	92.22
K	-0.106	-0.107	-0.105	-0.112
a	0.0371	0.0151	0.0133	0.0108
b	-0.0039	-0.0016	-0.0014	-0.0012

We developed an RBF controller for this ship at the depth to draft ratio (H/DT) of 1.93 under varying conditions of forward speed of the ship. PDF controllers (see Section 3.6.4) for this ship were designed by Vahedipour et al. [1990]. These were used as supervisors of the RBF controller. The performance of the controller is illustrated in Figure 6.9. As can be seen, the performance of the controller is quite satisfactory at the given depth of water (H/DT = 1.93). To check the robustness of the controller, we used this same controller when parameters of the ship change at H/DT = 1.5, 2.5 and  $\infty$ . The performance of the controller under these conditions is shown in Figure 6.10, 6.11 and 6.12 respectively. These Figures suggest that the RBF controller performance is robust at various depths of water.

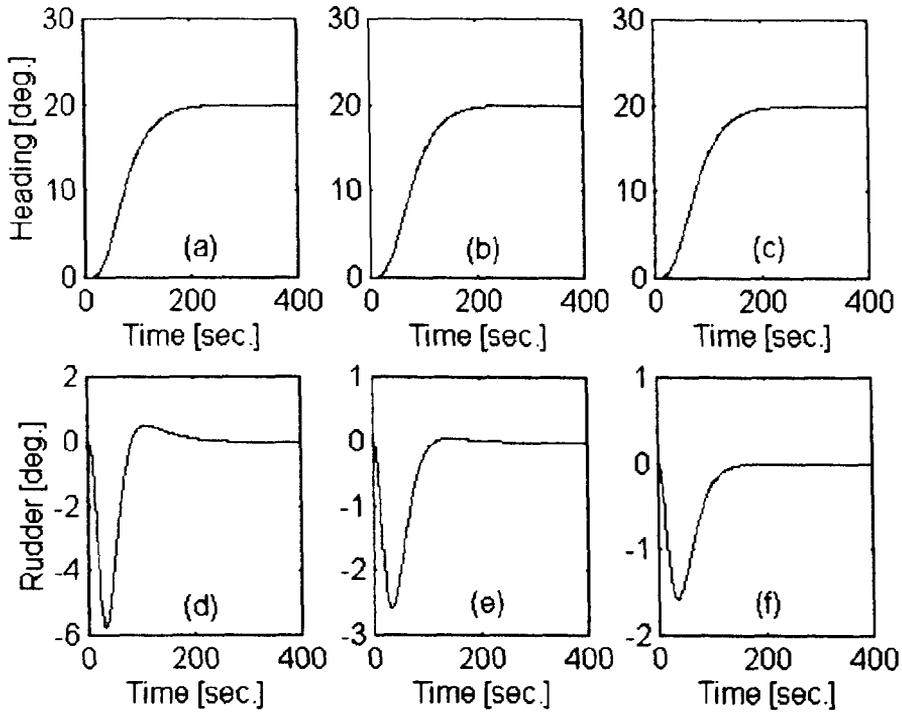


Figure 6.9: Performance of RBF controller at  $H/DT = 1.93$

- (a) Heading response at 6 m/s (d) the corresponding control signal  
 (b) Heading response at 9 m/s (e) the corresponding control signal  
 (c) Heading response at 12 m/s (f) the corresponding control signal

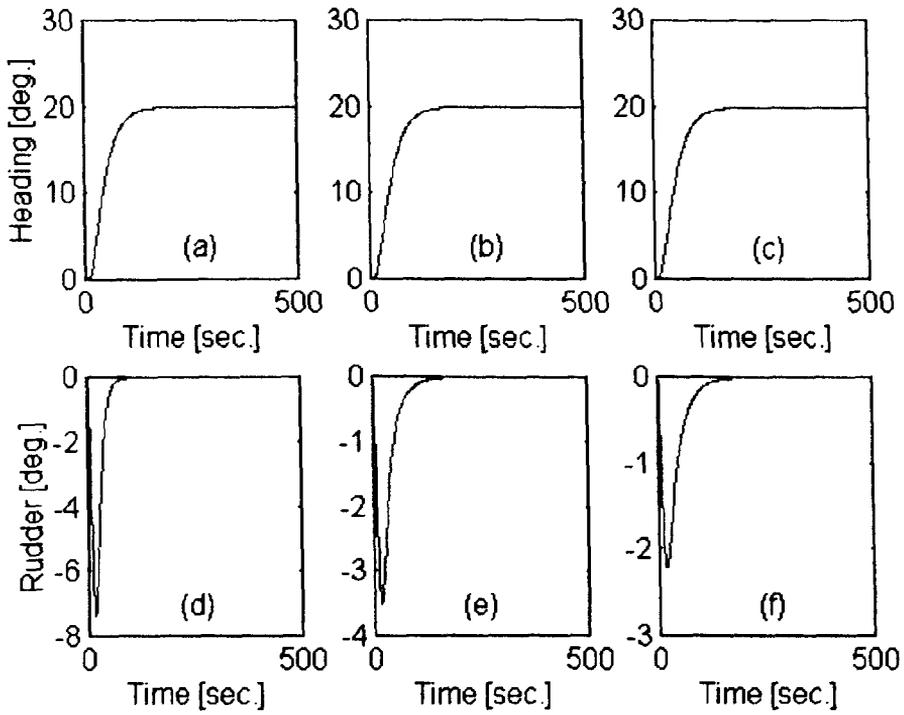


Figure 6.10: Performance of RBF controller at  $H/DT = 1.5$

- (a) Heading response at 6 m/s (d) the corresponding rudder response
- (b) Heading response at 9 m/s (e) the corresponding rudder response
- (c) Heading response at 12m/s (f) the corresponding rudder response

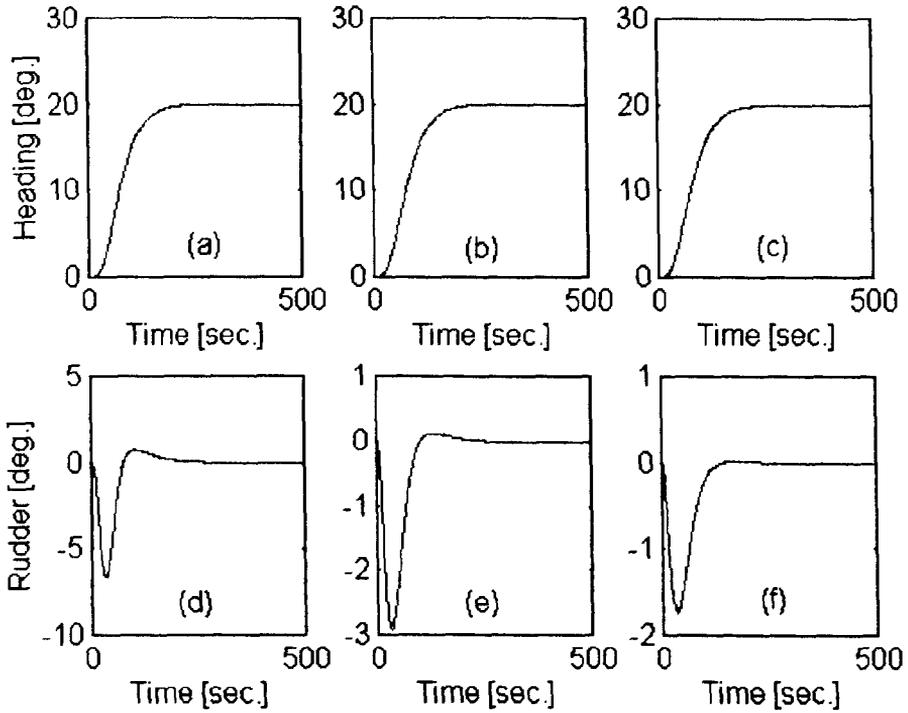


Figure 6.11: Performance of RBF controller at  $H/DT = 2.5$

- (a) Heading response at 6 m/s (d) the corresponding control signal
- (b) Heading response at 9 m/s (e) the corresponding control signal
- (c) Heading response at 12 m/s (f) the corresponding control signal

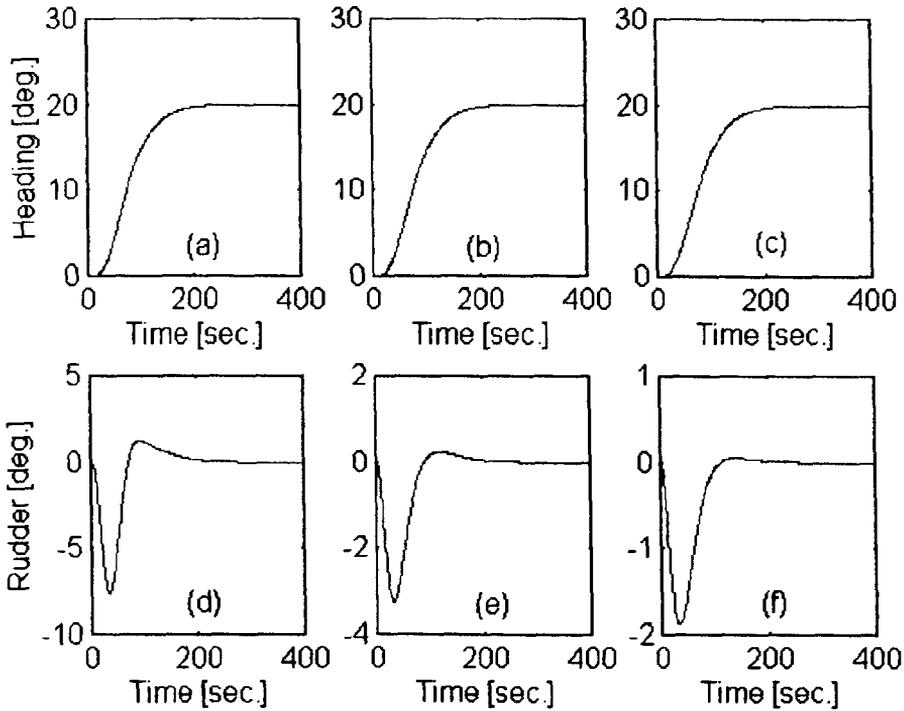


Figure 6.12: Performance of RBF controller at  $H/DT = \infty$

- (a) Heading response at 6 m/s (d) the corresponding control signal
- (b) Heading response at 9 m/s (e) the corresponding control signal
- (c) Heading response at 12 m/s (f) the corresponding control signal

## 6.6 Conclusions:

This Chapter investigates the potential of radial basis function networks for ship steering control systems. The networks are trained by using the orthogonal least squares algorithm and their performance is compared with that achieved by the multilayer perceptron networks of the previous Chapter. Following are the main conclusions of this Chapter:

- The performance of radial basis function networks is as good as achieved by multilayer perceptron networks of Chapter 5. However,

radial basis function networks have favourable characteristics in terms of the best approximation property and a compact network structure.

- Radial basis function networks have a faster training time than multilayer perceptron networks. However, like in other applications, they require more hidden layer neurons than multilayer perceptron networks in this application for comparable performance levels.

## Chapter 7

### AN APPLICATION INVOLVING REAL DATA

The multilayer perceptron (MLP) and radial basis function (RBF) network controllers of the previous Chapters were developed on the basis of simulation studies, where the control signal was the rudder angle. The investigations of this Chapter are different from those presented in the previous Chapters, in two important respects:

1. The ANN controller is developed from real data. The data were obtained from Dr. Euan McGookin, [McGookin, 1997] and were gathered while he was testing his sliding mode controller on scale model of a supply ship at the Guidance, Navigation and Control (GNC) Laboratory, Department of Engineering Cybernetics, Norwegian University of Science and Technology, Trondheim. The model is called *CyberShip1*. Its length is 1.17m which is one seventieth the size of the actual vessel. The linear as well as angular velocities of *CyberShip1* are approximately eight times larger than the actual vessel's [McGookin, 1997]. A picture of the model is shown in Figure 7.1.
2. Supply ships are used for oil platform support. To carryout this role, they should be highly manoeuvrable. Moreover, they should also maintain their position accurately while loading and unloading. Such a manoeuvre is called *Dynamic Position Keeping (DPK)* and can be difficult for a vessel to execute if steering is through the use of a rudder only. To avoid this problem, most supply ships use movable thrusters to maintain their position and heading. This means that the dynamics of this type of ships are different from those considered in the previous Chapters. Hence, this

study is significantly different from those presented earlier both in terms of the ship dynamics and the fact that experimental data are used. The *CyberShip1* has four thrusters, two at the bow and two at the stern, as shown in Figure 7.1.

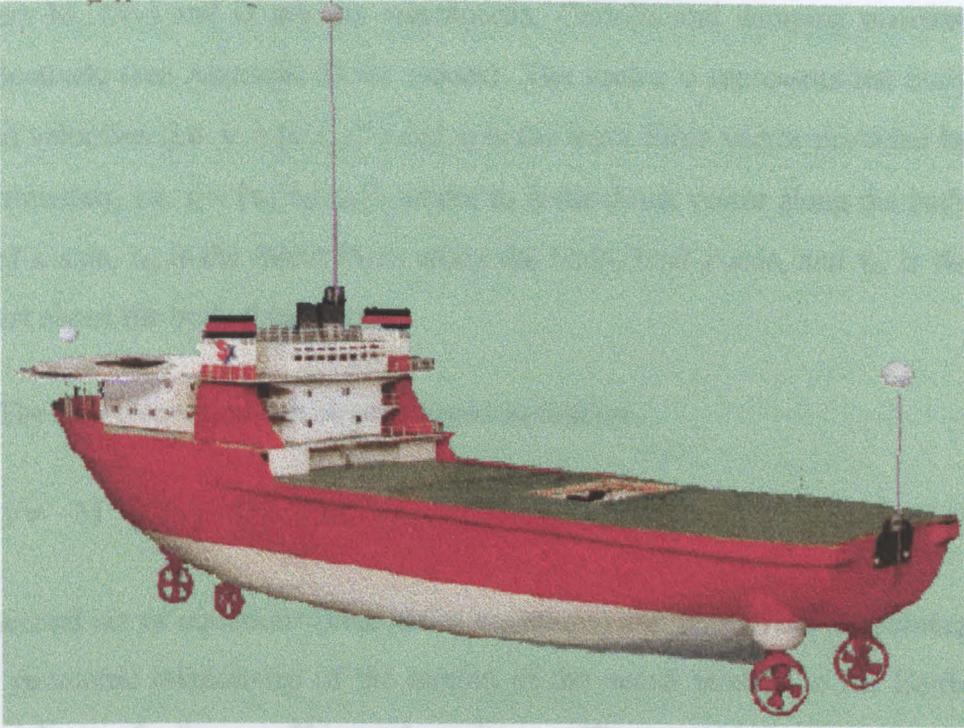


Figure 7.1: CyberShip 1.

## 7.1 Ship Model Dynamics:

As described in Chapter 2, the equations describing the motion of a ship are derived using Newton's laws of motion by taking two co-ordinate systems into account: an Earth fixed reference frame  $X_0O_0Y_0Z_0$  and a vessel fixed reference frame  $XOYZ$  (see Figure 2.1). These reference frames provide two sets of equations, the *Kinetic* and *Kinematics* equations. The kinetic equations result from the hydrodynamics of the vessel in the body fixed reference frame.

For a thrust driven ship, these equations can be represented as [Fossen, 1994; Fossen and Grøvlén, 1998]:

$$M\dot{v} + C(v)v + Dv = \tau_t \quad (7.1)$$

where  $M$ ,  $C(v)$  and  $D$  are the mass/inertia, Coriolis and damping matrices respectively (see Appendix D for values). The vector  $v$  represents the body fixed velocities (i.e.  $v = [u \ v \ r]^T$ ) and  $\tau_t$  is the input force vector provided by the thrusters, i.e.  $\tau_t = [\tau_{t1} \ \tau_{t2} \ \tau_{t3}]^T$ , where  $\tau_{t1}$  is the thrust vector along the body fixed x-axis,  $\tau_{t2}$  is the thrust force along the body fixed y-axis, and  $\tau_{t3}$  is the thrust about the body fixed z-axis.

The above equation can be rearranged as follows:

$$\dot{v} = -M^{-1}(C(v) + D)v + M^{-1}\tau_t \quad (7.2)$$

A second set of equations (kinematics equations) can be derived by defining the geometric relationship of the motion of the vessel relative to the Earth-fixed frame of reference. This can be written as [Fossen, 1994]:

$$\dot{\eta} = J(\eta)v \quad (7.3)$$

where  $J$  denotes Euler equations relating the two reference frames and  $\eta = [\Psi \ x \ y]^T$  represents the Earth-fixed states.

By combining equation (7.2) and (7.3), the following state-space representation can be obtained:

$$\dot{x} = Ax + B\tau_t \quad (7.4)$$

where  $A = \begin{bmatrix} -M^{-1}(C(v) + D) & 0 \\ J(\eta) & 0 \end{bmatrix}$ ,  $B = \begin{bmatrix} M^{-1} \\ 0 \end{bmatrix}$ ,  $\dot{x} = \begin{bmatrix} \dot{v} \\ \dot{\eta} \end{bmatrix}$ , and  $x = \begin{bmatrix} v \\ \eta \end{bmatrix}$ .

## 7.2 Thruster Dynamics:

As mentioned above, the thrusters provide the driving forces that propel the vessel. The thruster configuration for *CyberShip1* is shown in Figure 7.2. In the Figure, the position of the thrusters is given relative to the centre of gravity which is also the origin of the body fixed reference frame. Each thruster is represented by (a) the force it produces (i.e.  $f_i$ ,  $i = 1,2,3,4$ ) and (b) the azimuth angle  $\alpha_i$  defining the direction of the corresponding force, as shown in Figure 7.2.

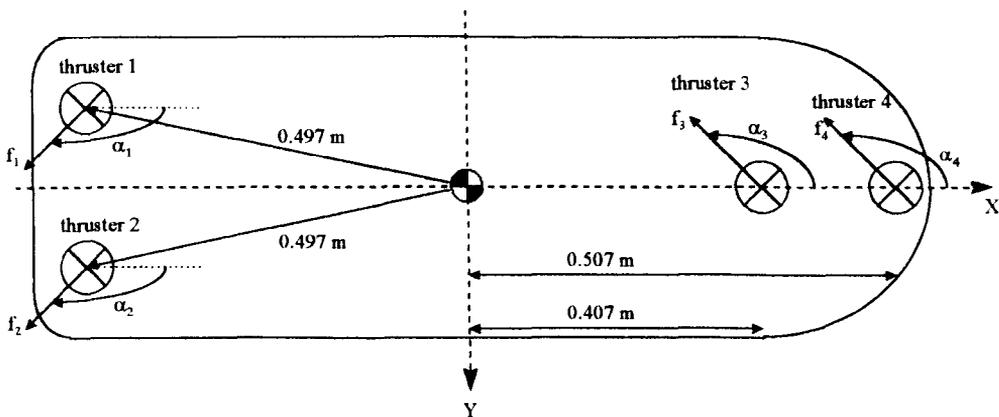


Figure 7.2: Thruster Configuration

The angles  $\alpha_1$  and  $\alpha_2$  for the model can be set independently whereas  $\alpha_3$  and  $\alpha_4$  are always equal showing that their corresponding thrusters operate in the same direction.

The thruster forces and angles can be related to the thruster inputs  $\tau_i$  of equation (7.4) as follows:

$$\tau_i = G(\alpha)f \quad (7.5)$$

where  $f = [f_1 \ f_2 \ f_3 \ f_4]^T$ ,  $\alpha = [\alpha_1 \ \alpha_2 \ \alpha_3 \ \alpha_4]^T$  and

$$G(\alpha) = \begin{bmatrix} \cos\alpha_1 & \cos\alpha_2 & \cos\alpha_3 & \cos\alpha_3 \\ \sin\alpha_1 & \sin\alpha_2 & \sin\alpha_3 & \sin\alpha_3 \\ 0.497 \sin(\alpha_1 - \theta_1) & 0.497 \sin(\alpha_2 - \theta_2) & 0.407 \sin\alpha_3 & 0.527 \sin\alpha_3 \end{bmatrix} \quad (7.6)$$

where  $\theta_1$  and  $\theta_2$  are the phase shift angles of the thrusters relative to the centre of gravity.

The maximum force that a thruster of *CyberShip1* can produce is estimated as  $\pm 1.2$  N. The corresponding maximum azimuth angle limit is estimated at  $\pm\pi$  radians. There are no rate limits given since the forces occur almost instantaneously in relation to the dynamic characteristics of the vessel and the direction is chosen to be fixed. McGookin [McGookin, 1997] has suggested the following constant values of the thruster angles:  $\alpha_1 = \alpha_2 = \pi$  radians,  $\alpha_3 = \alpha_4 = \pi/2$  radians. Substituting these values in (7.6) and neglecting very small components, matrix  $G(\alpha)$  becomes,

$$G(\alpha) = \begin{bmatrix} -1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0.407 & 0.527 \end{bmatrix} \quad (7.7)$$

This indicates that the surge motion is governed by thrusters 1 and 2 at the stern and the sway and yaw motion is governed by thrusters 3 and 4.

### 7.3 Sliding Mode Controller Design:

We have already described in Chapter 3 how a non-linear sliding mode controller (SMC) can be designed for a course changing control system. For *CyberShip1* the controller can be designed on the basis of equation (7.4). This equation has three inputs  $\tau_{11}$ ,  $\tau_{12}$  and  $\tau_{13}$ . McGookin [1997] has demonstrated

that for a course changing autopilot, only the yaw rate thrust input ( $\tau_{13}$ ) can be used to control the course. The remaining two inputs ( $\tau_{11}$  and  $\tau_{12}$ ) do not affect the heading motion significantly. The resulting controller has the following form:

$$\tau_{13} = -\mathbf{k}_s^T \mathbf{x}_s + (\mathbf{h}_s^T \mathbf{b}_s)^{-1} (\mathbf{h}_s^T \dot{\mathbf{x}}_{sd} - \eta_s \tanh(\sigma_s(\hat{\mathbf{x}}_s) / \phi_s)) \quad (7.8)$$

This controller has the same form as equation (3.30). However, there is a difference. Here the governing input is a force vector rather than a rudder deflection. McGookin [1997] optimized the parameters of the controller by means of Genetic Algorithms. These parameters are given in Table 7.1 where ps1 and ps2 are the two heading closed loop poles.

Table 7.1: Optimized parameter values of SMC

Ps1	-2.2092
ps2	-0.2059
$\eta_s$	8.1620
$\phi_s$	1.2851

The overall simulation set-up adopted by McGookin [1997] is illustrated in Figure 7.3. In the closed loop system, the surge motion is induced by a  $\tau_{11}$  step command of +1.2 N and the sway thrust force  $\tau_{12}$  is set to 0.0 N. The reason that sway velocity is made available for this vessel is due to the role it fulfils. Since a supply vessel provides support for oil platforms it must be able to maintain a constant position while loading and unloading takes place. In other words, a full velocity information is required to operate it effectively.

McGookin [McGookin, 1997] tested the performance of his course changing sliding mode controller on the physical model of *CyberShip1* at GNC, Department of Engineering Cybernetics, The Norwegian University of Science and Technology, Trondheim. He was kind enough to provide us with the real data which he gathered while testing his sliding mode controller on the physical model. The data sets include a  $20^{\circ}/-20^{\circ}$  manoeuvre of the physical model, its desired response, and the output  $\tau_{13}$  of the sliding mode controller.

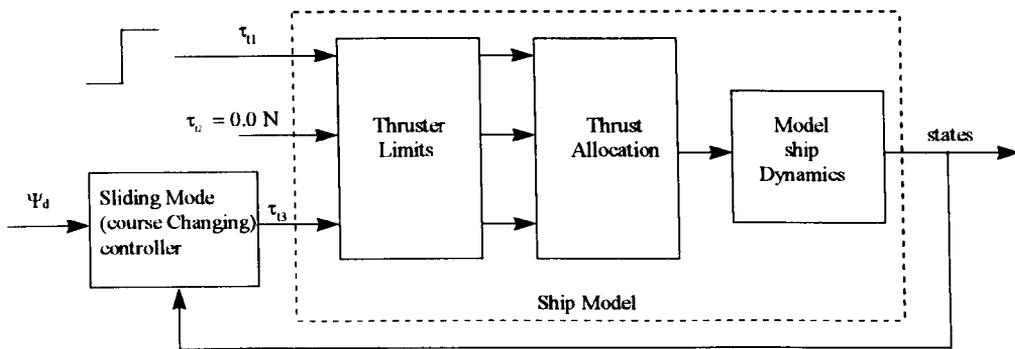


Figure 7.3: Ship Model and Course changing Controller Configuration

The data were gathered at a fixed speed only. Probably due to the limited size of the tank, it was not possible to gather the data at varying speeds.

## 7.4 Development of ANN controller:

Here our main objective is to develop an ANN controller that could mimic the dynamics of the course changing sliding mode controller of Figure 7.3. We developed an RBF controller by using the desired states and the error signal (i.e. difference between the actual and desired states) as inputs and the signal  $\tau_{13}$  as the output. The network was trained by using the orthogonal least squares algorithm. Gaussian functions were used as the radial basis

functions. The matching to the training data is shown in Figure 7.4. This matching was achieved when 40 neurons were used in the hidden layer.

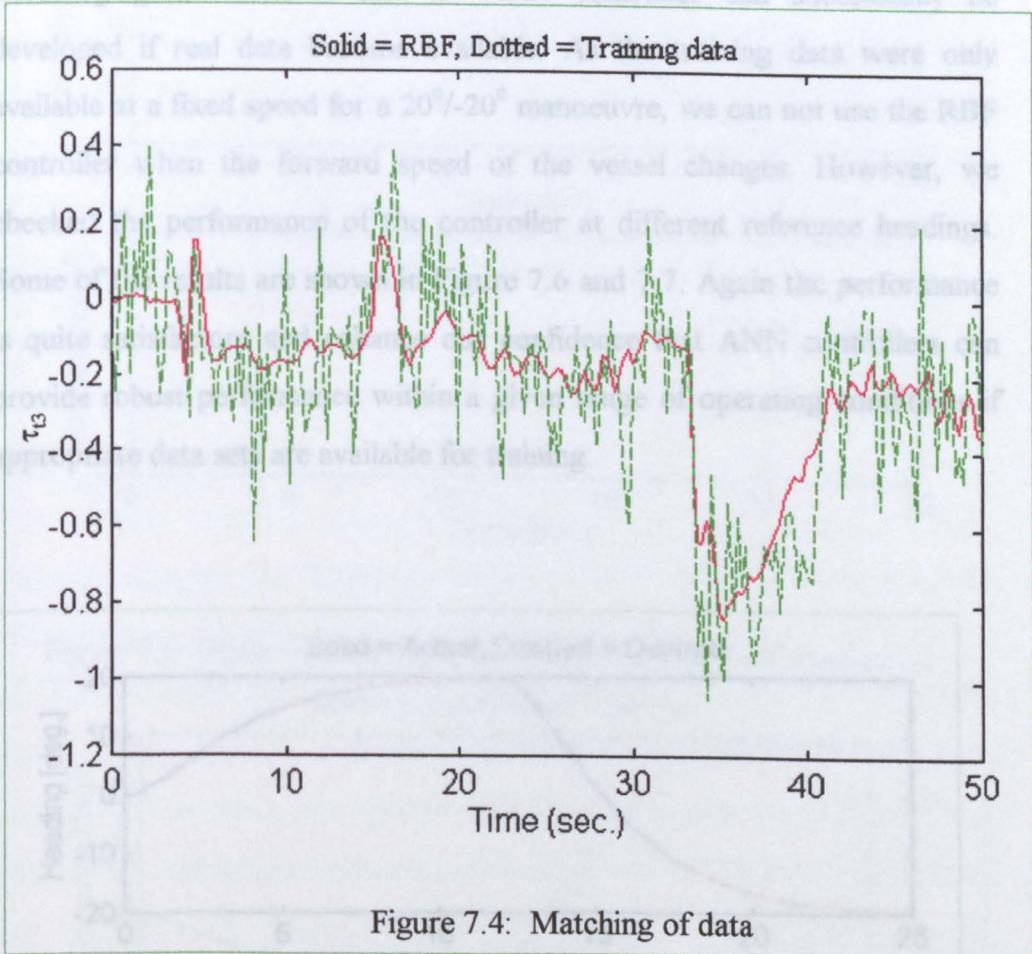


Figure 7.4: Matching of data

The above Figure shows that the training data is quite noisy and involves a small offset. There could be several reasons of this noise and offset.

- Very little consideration is given to the drag caused by the position of the thrusters. Since the bow thrusters are perpendicular to the flow over the hull they cause maximum drag and this affects the motion of the model. This could account for the slight offset in the thruster plot.
- The roll motion and water disturbance effects are neglected. They can alter the motion in the lab's manual basin.

The performance of the trained controller is shown in Figure 7.5 for a reference heading of  $20^{\circ}/-20^{\circ}$ . The performance is quite satisfactory and encouraging. It indicates that an ANN controller can successfully be developed if real data become available. As the training data were only available at a fixed speed for a  $20^{\circ}/-20^{\circ}$  manoeuvre, we can not use the RBF controller when the forward speed of the vessel changes. However, we checked the performance of the controller at different reference headings. Some of the results are shown in Figure 7.6 and 7.7. Again the performance is quite satisfactory and enhance our confidence that ANN controllers can provide robust performance within a given range of operating conditions if appropriate data sets are available for training.

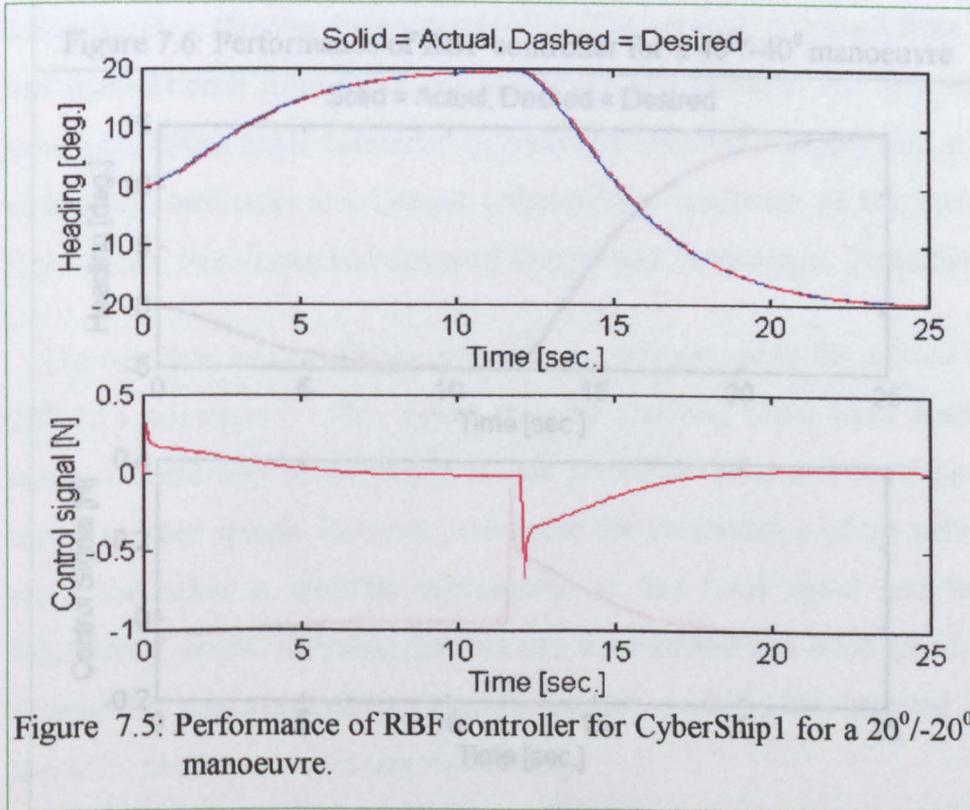


Figure 7.5: Performance of RBF controller for CyberShip1 for a  $20^{\circ}/-20^{\circ}$  manoeuvre.

Figure 7.7: Performance of RBF controller for a  $-5^{\circ}/+15^{\circ}$  manoeuvre.

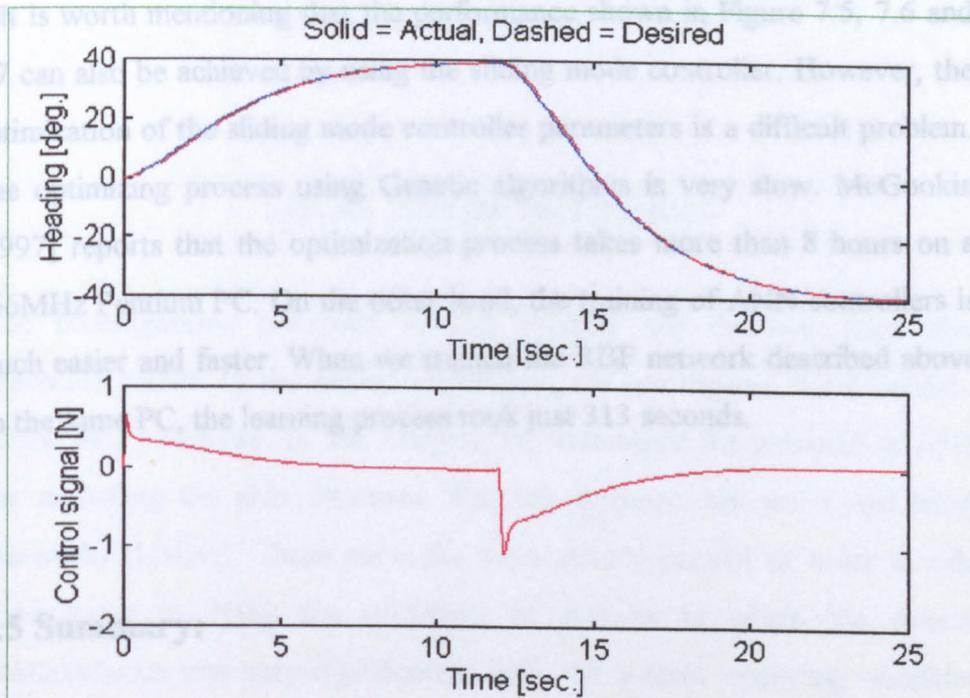


Figure 7.6: Performance of RBF controller for a  $40^\circ/-40^\circ$  manoeuvre

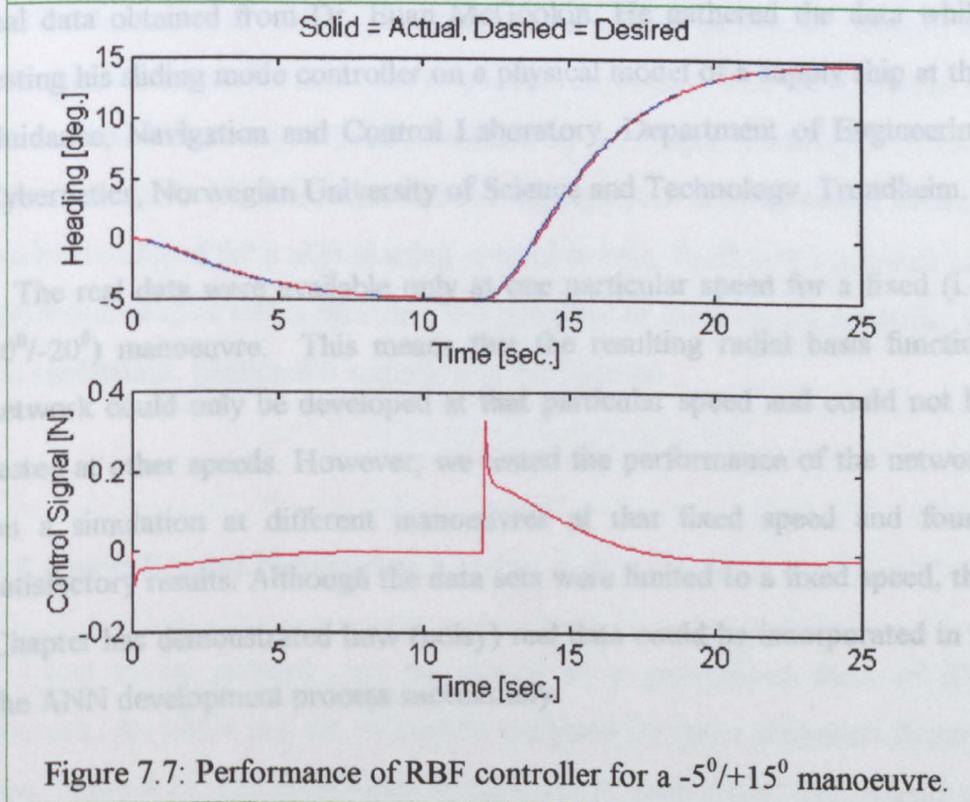


Figure 7.7: Performance of RBF controller for a  $-5^\circ/+15^\circ$  manoeuvre.

It is worth mentioning that the performance shown in Figure 7.5, 7.6 and 7.7 can also be achieved by using the sliding mode controller. However, the optimization of the sliding mode controller parameters is a difficult problem. The optimizing process using Genetic algorithms is very slow. McGookin [1997] reports that the optimization process takes more than 8 hours on a 166MHz Pentium PC. On the other hand, the training of ANN controllers is much easier and faster. When we trained the RBF network described above on the same PC, the learning process took just 313 seconds.

## **7.5 Summary:**

In this Chapter we have developed a radial basis function network controller for a thruster driven supply ship. The network is trained from the real data obtained from Dr. Euan McGookin. He gathered the data while testing his sliding mode controller on a physical model of a supply ship at the Guidance, Navigation and Control Laboratory, Department of Engineering Cybernetics, Norwegian University of Science and Technology, Trondheim.

The real data were available only at one particular speed for a fixed (i.e.  $20^{\circ}/-20^{\circ}$ ) manoeuvre. This means that the resulting radial basis function network could only be developed at that particular speed and could not be tested at other speeds. However, we tested the performance of the network on a simulation at different manoeuvres at that fixed speed and found satisfactory results. Although the data sets were limited to a fixed speed, this Chapter has demonstrated how (noisy) real data could be incorporated in to the ANN development process successfully.

## *Chapter 8*

### **LOCAL MODEL NETWORKS**

The investigations presented in this Chapter are different from those presented earlier. In the previous Chapters, we investigated the potential of ANNs as a controller. In this Chapter, we investigate the potential of ANNs for modelling the ship dynamics. For this purpose, we use Local Model Networks (LMNs). These networks have already proved of value in other applications involving the modelling of systems in which the dynamic characteristics can vary significantly with the system operating conditions [Johansen and Fosss, 1992a; Murray-Smith, 1992,1994; Murray-Smith et al., 1992; Hunt et al., 1996b; Gollee and Murray-Smith, 1997, Gøttsche et al. 1998; Johansen et al. 1998]. The Chapter is organized as follows. Section 8.1 introduces local model networks in general. In Section 8.2, the literature on local models is surveyed briefly. In Section 8.3 major advantages of local model networks are listed. Section 8.4 explains how a local model network can be developed for a ship steering control system. Section 8.5 presents two simulation studies which illustrate the potential of local model networks for the application. Section 8.6 summarizes the Chapter.

#### **8.1 Local Model Networks:**

Local model network can be viewed as a generalized form of RBF network. An LMN is a set of models weighted by some activation function (see Figure 8.1). The same input signal is fed to each model and outputs are weighted according to some variable or variables,  $\phi$ ,

$$y(t) = \sum_{i=1}^n y_i(t) \rho_i(\phi) \quad (8.1)$$

where  $y(t)$  is the model network output,  $\rho_i(\phi)$  is the validity function (basis function) of the  $i$ th model,  $n$  is the number of models, and  $y_i(t)$  is the output of the  $i$ th local model  $f_i(\phi)$ . The weighting or activation of each local model is calculated using an activation function which is a function of the

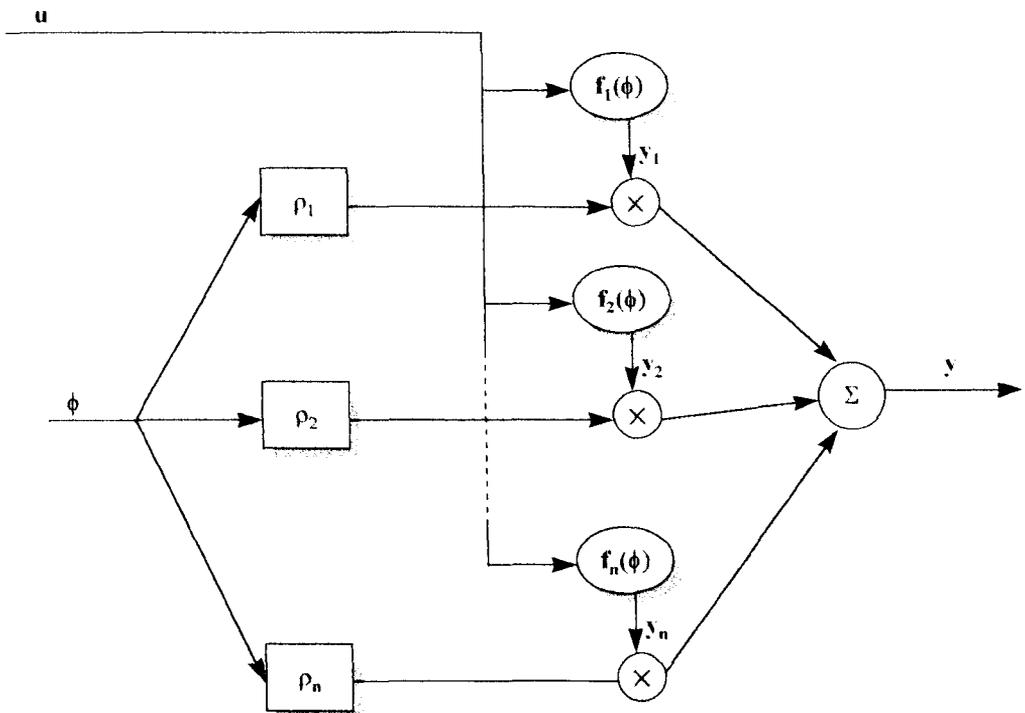


Figure 8.1: General architecture of LMN

scheduling variable. The scheduling variable could be a system state variable, an input variable or some other system parameter. It is also feasible to schedule on more than one variables and to establish a multi-dimensional local model network [Murray-Smith, 1994]. Although any function with a locally limited activation might be applied as an activation function, Gaussian functions are applied most widely. Other popular validity functions include B-Splines [Friedmann, 1991; Kavli, 1993] and Kernel functions [Hlavàcková,

1995]. In this work we restrict ourselves to the Gaussian function. For modelling tasks the validity functions should form a partition of unity for the input space, i.e. at any point in the input space, the sum of all basis function activations should be 1. This is a necessary requirement for the network to be able to globally approximate systems as complex as the basis function' local models. Werntges [1993] discusses the advantages of normalization in RBF nets, promoting the advantages of a partition of unity produced by normalization. The basis functions can be normalized as follows:

$$\rho_i(\phi) = \frac{\rho_i(\phi)}{\sum_{j=1}^n \rho_j(\phi)} \quad (8.2)$$

The individual component (local) models  $f_i$  of an LMN can be of any form; they can be non-linear or linear, have a state-space or input-output description, or be discrete or continuous time. They can be of different character, using physical models of the system for operating conditions where they are available, and parametric models for conditions where there is no physical description available. These can also be ANN models such as MLP or RBF networks. The individual local models are smoothly interpolated by the validity functions  $\rho_i$  to produce the overall model.

The learning process in local model networks can be divided into two tasks:

1. Find the optimal number, position and shape of the validity functions, i.e. define the structure of the network.
2. Find the optimal set of parameters for the local models, i.e. define the parameters of the network. These parameters could be the complete set of coefficients for a linear model, numerical parameters of a non-linear model, or even switches which altered the local model structure. The parameters

are usually optimized by using a least squares output error criterion. The details can be found in [Murray-Smith, 1994; Murray-Smith and Johansen, 1997].

## **8.2 Literature of local model methods in learning and modelling:**

Jones et al. [1989] suggested that the representational abilities of the normal basis function (BF) networks can be extended to a generalized form of BF network, where the basis functions are used to weight other functions of the inputs as opposed to straightforward weights. This suggestion was followed up by Stokbro et al. [1990] and Barnes et al. [1991]. The work of Johansen and Foss [1992a, 1992b, 1992c, 1993] and Murray-Smith [1992, 1994] is a generalization of these ideas. The adaptive expert networks of Jacobs et al. [1991] are also local model networks, where the local models are called expert systems and the integration of the various experts is made by so called gating networks. These were developed into hierarchical models in Jordan and Jacobs [1991, 1993].

The idea of using locally accurate models is also described in the statistical literature in Cleveland et al. [1988], where local linear or quadratic models are weighted by smoothing functions. Priestley's *State Dependent Models* (SDM) [Priestley, 1988] have many similarities to LMNs.

The use of local linear models without interpolation, that is, piecewise linear models, have been suggested by a number of researchers including [Tong and Lim, 1980; Billings and Voon, 1987; Skeppstedt, 1988; Hilhorst et al. 1991; Skeppstedt et al. 1992].

Local model networks have also links with fuzzy logic systems. The methods used in [Takagi and Sugeno, 1985] are effectively overlapping

piecewise linear models, with the interpolation between models provided by the membership functions. Similar works are reported in [Sugeno and Kang, 1988; Foss and Johansen, 1993; Harris et al. 1993; and Wang, 1994].

Local model networks for modelling and control of dynamic systems have been applied by many authors including [Johansen and Foss, 1992a, 1992b, 1992c, 1993, 1997; Murray-Smith, 1992, 1994; Murray-Smith and Hunt, 1995; Foss et al. 1995; Gollee and Hunt, 1997; Gollee and Murray-Smith, 1997; Gawthrop 1996; Hunt et al. 1996a; Hunt et al. 1996b; Hunt and Johansen, 1997; Sbarbaro and Johansen, 1997].

### 8.3 Advantages of LMNs

LMNs offer many advantages. These include the following:

- The LMN has a transparent structure which allows a direct analysis of local model properties.
- The LMN architecture is less sensitive to the curse of dimensionality than other local representations, such as RBF networks.
- Non-linear models based on LMNs are able to capture the non-linear effects and provide accuracy over a wide operational range.
- The LMN framework allows the integration of *a priori* knowledge to define the model structure for a particular problem. This leads to more interpretable models which can be more reliably identified from a limited amount of observed data.

## 8.4 LMNs for modelling the ship dynamics:

As discussed in earlier Chapters, the ship steering dynamics change significantly with operating conditions, such as with the forward speed of the vessel. If we derive various linear (e.g. Nomoto's first-order or second-order models) or non-linear models (e.g. Bech's model or Norrbin's model) at different forward speeds of the ship, then an LMN can easily be developed that could represent the ship model for the range of forward speeds. The derivation of linear or non-linear ship models at a particular forward speed is well established and such models are already available in the literature for most commercial ships. An LMN developed on the basis of these physically oriented models has several advantages over other ANN architectures such as MLP or RBF networks. First, the conventional ANN architectures are based on the non-transparent, black box approach that makes it difficult to incorporate *a priori* system information, and to interpret the final structure in terms of the physical characteristics of the process under consideration. Secondly, conventional neural network modelling fails to exploit the significant theoretical results available in the conventional modelling and control domain, making it difficult to analyse their behaviour.

The training of LMNs for modelling ship dynamics is not difficult. As we use physically oriented models as the local models which are already available in the literature, the problem of parameter estimation and optimization is automatically solved. The individual local models could also be developed directly from ship sea trials data.

In general, there is no straight forward rule to choose optimal number of local models for a particular application. This number is usually decided on the basis of the range of scheduling variable  $\phi$ . For example, if we desire to develop a ship model for a range of forward speeds from 5 m/sec to 10 m/sec., then we can choose several local models derived separately at these

speeds, and possibly at some other speed(s) within this range. However, our experience shows that only two local models will be sufficient for the above range of forward speeds. This will be illustrated with the help of simulation studies presented in the following Section.

The selection of centres is a crucial problem in RBF networks when a Gaussian function is used as a validity function. This is not a problem in LMNs. These can be selected at the operating point, the local model is developed about. For example, if a local model network consists of two local models derived at 5 m/sec and 10 m/sec respectively, then Gaussian functions centred at 5m/sec and 10m/sec respectively can be used as validity functions  $\rho_i$ . The width  $\sigma_i$  of the Gaussian function can be found by means of simulation studies looking at the desired and actual responses. It can also be found by using the following formula [Murray-Smith and Gollee, 1994; Gollee et al. 1997]:

$$\sigma_i = k_\sigma \frac{1}{n} \sum_{j=1}^n |c_i - c_{i,j}| \quad (8.3)$$

where  $c_i$  is the current centre and  $c_{i,j}$  is the  $j$ th nearest neighbour to  $c_i$ . The scaling factor  $k_\sigma$  defines the degree of overlap between the validity functions.

## 8.5 Simulation Studies:

In this Section we present two examples which illustrate the worth of LMNs for ship steering control. In both examples, the scheduling variable is the forward speed of the ship and the activation functions used are the normalized Gaussian functions.

### Example 8.1:

This example involves the simulation results carried out from the model of ROV Zeefakkel. We have already developed MLP and RBF controllers for this ship in Chapter 5 and 6 respectively. In these Chapters, the ship was represented by the Norrbin's non-linear model of equation (2.30). Here our purpose is to develop an LMN which could replace the Norrbin's model and yields the same performance that can be achieved by the Norrbin's model for a given range of forward speeds. The network is developed as follows:

The first step in developing an LMN is to derive local models at various operating points. In the present study, our aim is to derive Norrbin's models at various forward speeds of the ship. These models at a speed of 5 m/sec and 10 m/sec are given in equation (8.4) and (8.5) respectively [See Table 5.1]:

$$\text{Norrbin's model at a speed of 5 m/sec: } \delta = 62\ddot{\Psi} + 2\dot{\Psi} + 0.8\Psi^3 \quad (8.4)$$

$$\text{Norrbin's model at a speed of 10 m/sec: } \delta = 15.5\ddot{\Psi} + \dot{\Psi} + 0.1\Psi^3 \quad (8.5)$$

The above (local) models can be interpolated as shown in Figure 8.1. The scheduling variable  $\phi$  of Figure 8.1 is the forward speed  $U$  and the controller  $u = \delta$  is the RBF controller of Chapter 6. The activation functions  $\rho_i$  ( $i = 1,2$ ) are the two normalized Gaussian functions having centres at 5 m/sec and 10 m/sec respectively. The overall closed loop system is shown in Figure 8.2. In the Figure, LMN1 and LMN2 are the local models given in equations (8.4) and (8.5) respectively. The same reference model is used here that was used in Section 5.3 for the development of MLP network. Similarly, the model of steering machine is the same as that of Section 5.3. Figure 8.3 illustrates the performance of the LMN at two different speeds, i.e. at 10 m/s and 7 m/s. As can be seen the performance of the LMN is same as can be

achieved by the non-linear Norrbinn's model (conventional model) at these speeds.

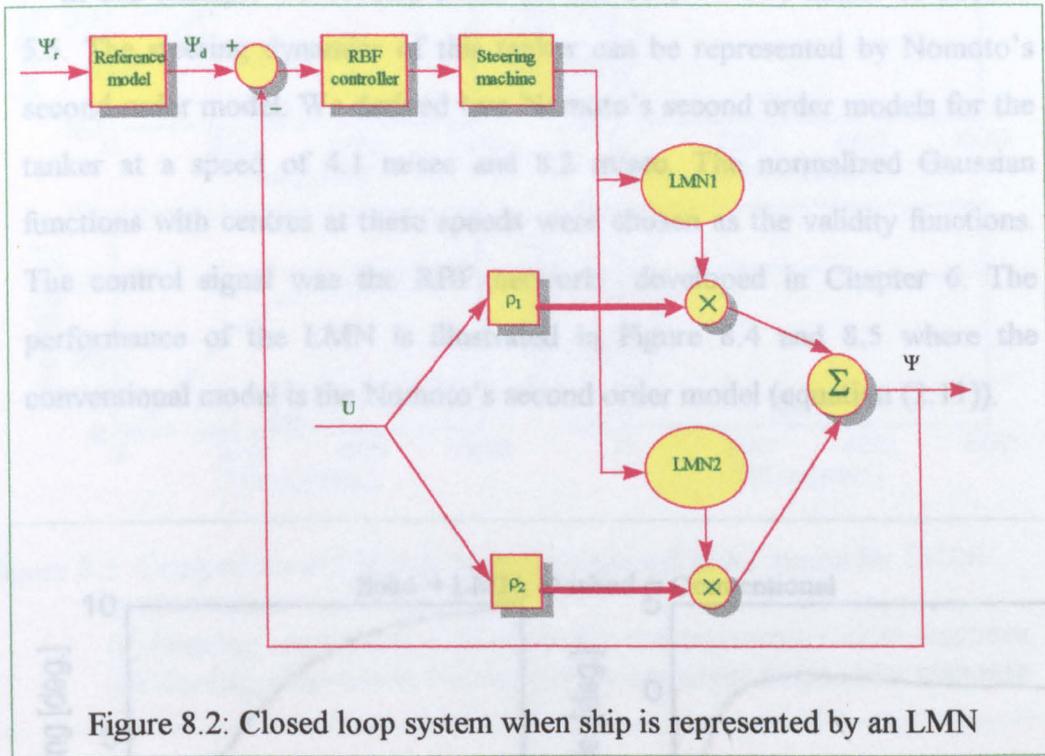


Figure 8.2: Closed loop system when ship is represented by an LMN

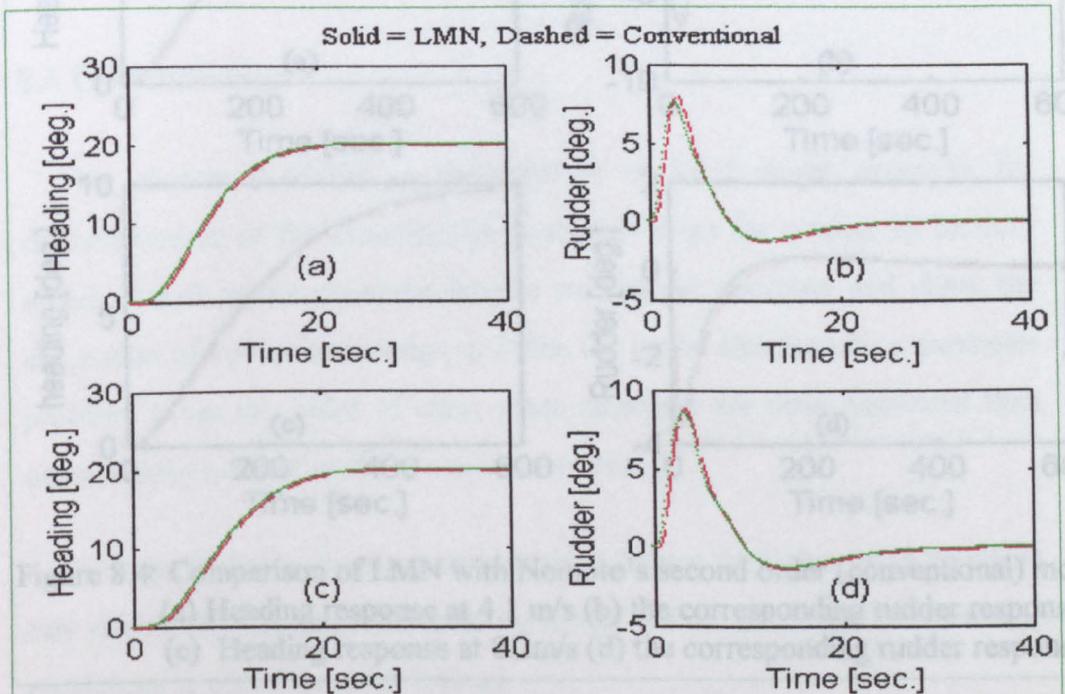
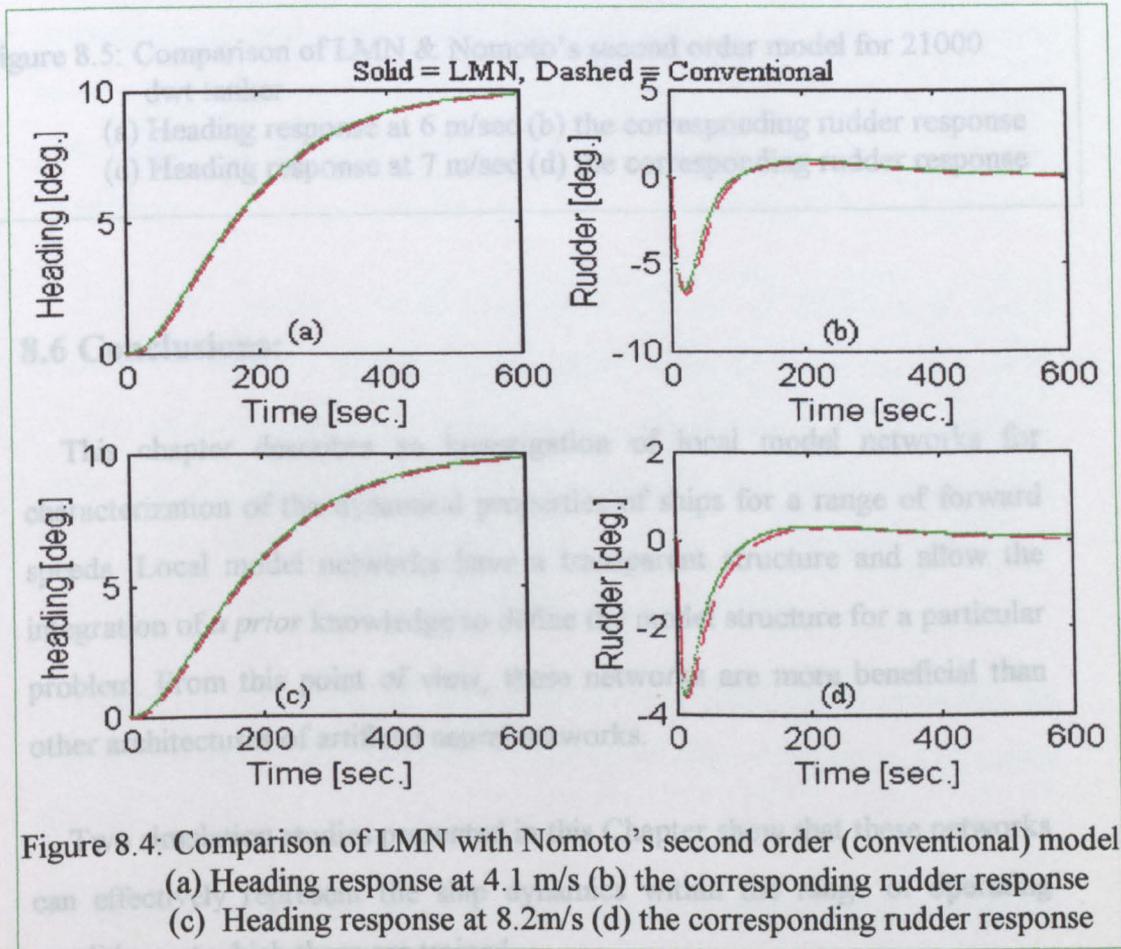


Figure 8.3: Performance of LMN for ROV Zeefakkel

(a) Heading response at 10 m/s (b) the corresponding rudder response  
(c) Heading response at 7 m/s (d) the corresponding rudder response

**Example 8.2:**

In this example we develop LMN for the 210,000 dwt tanker of Section 5.5. The steering dynamics of this tanker can be represented by Nomoto's second order model. We derived two Nomoto's second order models for the tanker at a speed of 4.1 m/sec and 8.2 m/sec. The normalized Gaussian functions with centres at these speeds were chosen as the validity functions. The control signal was the RBF network developed in Chapter 6. The performance of the LMN is illustrated in Figure 8.4 and 8.5 where the conventional model is the Nomoto's second order model (equation (2.11)).



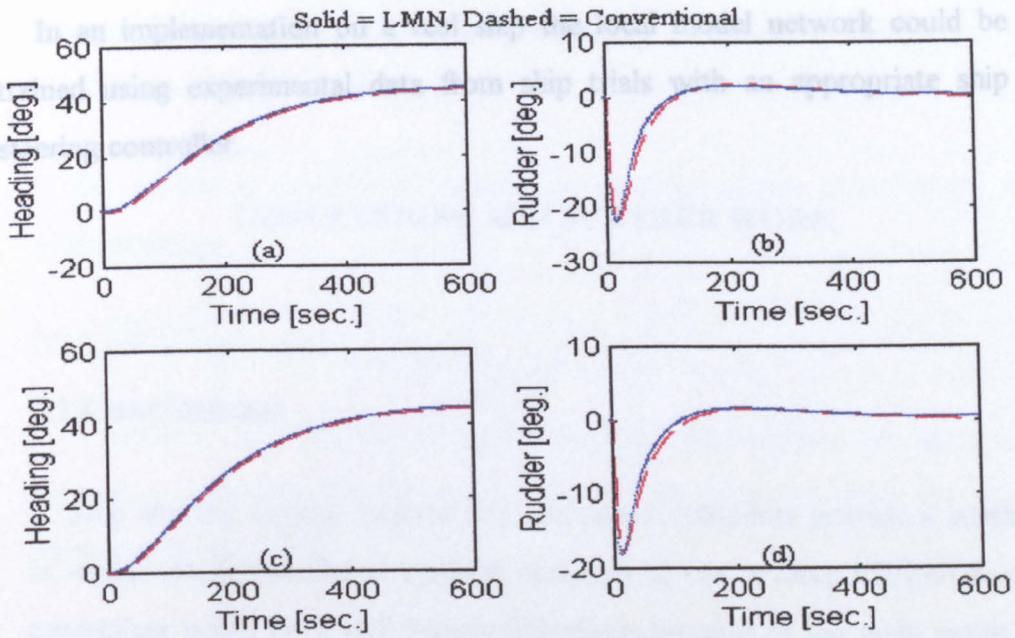


Figure 8.5: Comparison of LMN & Nomoto's second order model for 21000 dwt tanker

- (a) Heading response at 6 m/sec (b) the corresponding rudder response  
 (c) Heading response at 7 m/sec (d) the corresponding rudder response

## 8.6 Conclusions:

This chapter describes an investigation of local model networks for characterization of the dynamical properties of ships for a range of forward speeds. Local model networks have a transparent structure and allow the integration of *a priori* knowledge to define the model structure for a particular problem. From this point of view, these networks are more beneficial than other architectures of artificial neural networks.

Two simulation studies presented in this Chapter show that these networks can effectively represent the ship dynamics within the range of operating conditions at which these are trained.

In an implementation on a real ship the local model network could be trained using experimental data from ship trials with an appropriate ship steering controller.

## **CONCLUSIONS AND FURTHER WORK**

### **9.1 Conclusions:**

Ship steering control systems and associated autopilots provide a number of design challenges. Most systems currently in use involve PD/PID based controllers which have well known limitations because of the wide range of dynamical behaviour which can be exhibited by the vessel. Other approaches based on adaptive control techniques can give important benefits in terms of performance, but they can also suffer from disadvantages such as potential instabilities.

This thesis presents results concerning the use of feedforward neural networks to obtain a controller which incorporates the properties of a series of conventional controllers. Two types of feedforward neural networks, namely the multilayer perceptron (MLP) networks and radial basis function (RBF) networks have been used. These networks have already found many successful applications in the field of systems and control.

The thesis also describes an investigation of local model networks for characterization of the dynamical properties of ships for a range of forward speeds.

The main conclusion which we can draw from this work is that all the above architectures of neural networks are suitable for this application and

can yield robust performance within the range of operating conditions, for which they are trained.

The particular conclusions concerned with the above architectures are outlined below:

### **9.1.1 MLP Networks:**

A number of conclusions concerning the applicability of multilayer perceptron networks for the application can be extracted from the work presented in this thesis:

- MLP networks having only one hidden layer are sufficient for satisfactory performance, provided an appropriate number of neurons is used in the hidden layer. In other words, MLP networks containing more than one hidden layer are not necessary for this application.
- The performance of a control system incorporating an MLP network is robust within the range of forward speeds for which it is trained. It is observed that the performance could be satisfactory beyond the upper extreme of this range. However, it may not be satisfactory below the range for training. For example, if we train an ANN for a range of forward speeds from 5 m/s to 10 m/s, its performance will be satisfactory from 5 m/s to 10 m/s and also at speeds above 10 m/s. However, it may not yield good performance at speeds below 5 m/s.
- A large data set is not essential for successful training. A large data set may cause the learning process to be very slow. An appropriate subset of data for training can improve the speed of training without having a serious effect on the performance.

- The training time of an MLP network can be improved significantly by incorporating the Levenberg-Marquardt algorithm into the standard back-propagation learning algorithm.
- The performance of an MLP network trained with back-propagation algorithm with adaptive learning rate and momentum is almost the same in this application as that obtained through training by means of back-propagation with the Levenberg-Marquardt algorithm.
- A significant disadvantage of MLP networks is that there is no straight forward rule of choosing an appropriate number of hidden layer neurons for an optimal performance. This number is chosen by trial and error methods, starting with two or three neurons, and then increasing the number gradually, until satisfactory performance is achieved. This procedure is of course, tedious and time consuming. However, it is observed that only a very limited number of hidden layer neurons is needed for the MLP networks developed for ship steering control systems. This number is less than 15 for the ships considered in this thesis.

### **9.1.2 Radial Basis Function Networks:**

Because of their distinctive properties of best approximation, simple network structure and efficient learning procedure, radial basis function networks can have favourable characteristics in control applications when compared with MLP networks. All radial basis function networks presented in this thesis are trained using the orthogonal least squares algorithm. This algorithm selects an appropriate number of the radial basis function centres from input data, hence the problem of selecting an optimal number of hidden layer neurons is automatically solved.

The results presented in Chapter 6 and 7 demonstrate that the performance of radial basis function networks is as good as can be achieved by the MLP networks. Moreover, they have faster training time than MLP networks although they require more hidden layer neurons than MLP networks for the solution of the same problem.

### **9.1.3 Experience with real data:**

In Chapter 7, we have successfully developed radial basis function networks from the real data gathered from a scale model of a supply ship called *Cybership1*. Supply ships are used for oil platform support and use movable thrusters to maintain their position and heading accurately. The input and target data sets were only available at a fixed forward speed of the ship for a  $20^0/-20^0$  manoeuvre. Hence we could only develop RBF network controller at that particular speed and could not test the performance of the network at other speeds. However, we tested the performance of the network (on a simulation) at different manoeuvres and found good results.

In principle, if experimental data sets become available at different operating conditions, then the development of robust controllers involving radial basis function networks should not be difficult. This can be an interesting area for future research.

Controller networks could also be developed from ship trials data which mimic the action of an experienced helmsman in course changing manoeuvres. Such data might well be more readily available for a range of operating conditions than data from optimized ship steering control systems and could lead to interesting investigations of controllers which incorporate the essential features of the human operator in this specialized control task. It might also be possible using such data to generate local model representations of human

operator dynamics. Such a “grey box” approach to human operator studies could open up new areas of research on manual control problems.

#### **9.1.4 Local Model Networks:**

Local model networks have been developed successfully in Chapter 8 to represent ship dynamics for a range of operating conditions. Local model networks have several advantages over other artificial neural network (ANN) architectures. The conventional ANN architectures are based on the black box approach that makes it difficult to incorporate *a priori* system information and interpret the final structure in terms of the physical characteristics of the process under consideration. The simulation studies presented in this thesis indicate that these models can effectively represent ship dynamics for a range of operating conditions at which they are trained.

## **9.2 Suggestions for further work:**

This thesis will in future be regarded as being among the early literature on the applicability of feedforward neural networks for ship steering control systems. The thesis has reported successful development of artificial neural networks for modelling and control of ships. However, there are many aspects which need further attention. These are outlined below:

### **9.2.1 Development of Neuro autopilots in presence of disturbances:**

This work has clearly shown that neuro autopilots can successfully be developed for course changing operation under varying operating conditions such as the forward speed of a ship. It would be interesting to check the performance of these networks in presence of environmental disturbances such as waves, winds, and ocean currents generated by winds. It is well

established that, in general, artificial neural networks are robust to noisy data and perform well in presence of external disturbances. This suggests that neuro autopilots of ships should also perform well in noisy environments but experimental assessment of the effect of waves and other disturbances is essential. Such investigations could be carried out using model facilities such as those described in Chapter 7..

### **9.2.2 Development of ANNs from real data:**

We have successfully developed RBF network from real data in Chapter 7. A limitation of the work is that the training data were only available at a fixed speed for a  $20^0/-20^0$  manoeuvre. If data become available at different forward speeds of a ship at different depths of water under different loading conditions, then a powerful neuro controller could be developed. This may be difficult using the experimental facilities at Norwegian University of Science and Technology Trondheim because the relatively small size of the ship tank makes large manoeuvres at higher speeds impossible.

### **9.2.3 On-Line Training:**

The approach considered in this thesis for the development of ship autopilots is based on the off line training of neural networks. The on-line training of neural network controllers for the application will be another interesting and important development. Some researchers [e.g. Zhang et. al., 1995, 1996] are already working in this area but there are many opportunities for further research which could have implications for many other control applications of artificial neural networks.

### **9.2.4 Modelling the Ship Dynamics:**

We have developed local model networks to represent the ship dynamics under varying forward speeds. A possible development in this area will be to

develop the networks when the forward speed of the ship, the depth of water and the loading conditions change simultaneously. Complex changes in ship dynamics can also occur when vessels approach each other in restricted water ways. Such changes present considerable challenges in terms of conventional modelling methods and traditional mathematical description.

#### **9.2.5 Rudder Roll Stabilization:**

In some cases, an autopilot is not only used to control the heading of a ship, but it is used to reduce the roll motion as well. The development of neural networks for rudder roll stabilization (RRS) systems is another area of future research.

#### **9.2.6 Track Keeping Autopilots:**

As the name suggests, the track keeping autopilots enable a ship to follow a pre-specified route. Such autopilots are specially important in the areas where there are many other ships in the surroundings. The applicability of ANNs for such autopilots will also be a good topic of future research.

## REFERENCES

- Abkowitz, M.A., 1964. Lectures on ship hydrodynamics - steering and manoeuvrability, *Technical Report Hy-5*, Hydro- and Aerodynamics Laboratory, Lyngby, Denmark.
- Abkowitz, M.A., 1975. System identification techniques for ship manoeuvring trials, *Proceedings Symposium on Control Theory and Navy Applications*, Monterey, CA, 337-393.
- Abkowitz, M.A., 1980. Measurement of hydrodynamic characteristics from ship manoeuvring trials by system identification, *Transactions SNAME* **88**, 283-318.
- Ackley, D.H., G.E. Hinton, and T.J. Sejnowski, 1985. A learning algorithm for Boltzmann machines. *Cognitive Science* **9**, 147-169.
- Aizerman, M.A., E.M. Braverman, and L.I. Rozonoer, 1964a. Theoretical foundations of the potential function method in pattern recognition learning, *Automation and Remote Control* **25**, 821-837.
- Aizerman, M.A., E.M. Braverman, and L.I. Rozonoer, 1964b. The probability problem of pattern recognition learning and the method of potential functions, *Automation and Remote Control* **25**, 1175-1193.
- Amari, S., 1967. A theory of adaptive pattern classifiers, *IEEE Transactions on Electronic Computers* **EC-16**, 299-307.
- Amari, S., 1972. Characteristics of random nets of analog neuron like elements, *IEEE Transactions on Systems, Man and Cybernetics* **SMC-2**, 643-657.
- Amari, S., 1977. Neural theory of association and concept formation, *Biological Cybernetics* **26**, 175-185.
- Anderson, B.D.O., and J.B. Moore, 1971. *Linear Optimal Control*, Prentice Hall.

- Anderson, J.A., 1968. A memory storage model utilizing spatial correlation functions, *Kybernetik* **5**, 113-119.
- Anderson, J.A., 1972. A simple neural network generating an interactive memory, *Mathematical Biosciences* **14**, 197-220.
- Anderson, J.A., 1983. Cognitive and psychological computation with neural models, *IEEE Transactions on Systems, Man and Cybernetics* **SMC-13**, 799-815.
- Anderson, J.A., 1985. What Hebb synapses build. In *Synaptic Modification, Neuron selectivity, and Neuron System Organization* (W.B. Levy, J.A. Anderson, and S. Lehmkuhle, Eds.), 153-173, Erlbaum, Hillsdale, NJ.
- Anderson, J.A., 1995. *An Introduction to Neural Networks*, Bradford Books.
- Arbil, J., J. Salom, and O. Calvo, 1997. Fuzzy control of a sail boat, *International Journal of Approximate Reasoning* **16**, 359-375.
- Arie, T., M. Itoh, A. Senoh, N. Takahashi, S. Fujii, and N. Mizuno, 1986. An adaptive steering system for a ship, *IEEE Control Systems Magazine* **6**, 3-8.
- Åström, K.J., 1970. *Introduction to Stochastic Control Theory*, Academic Press.
- Åström, K.J., 1980. Why use adaptive techniques for steering large tankers? *International Journal of Control* **32**(4), 689-708.
- Åström, K.J., and T. Hägglund, 1995. *PID Control*, Research Triangle Park, N.C., Instrument Society of America.
- Åström, K.J., T. Hägglund, C.C. Hang, and W.K. Ho, 1993. Automatic tuning and adaptation for PID Controllers - A survey, *Control Engineering Practice* **1**, 699-714.
- Åström, K.J., and C.G. Källström, 1976. Identification of ship dynamics, *Automatica* **12**(9), 9-22.

- Åström, K.J., C.G. Källström, N.H. Norbbin, and L. Byström, 1975. The identification of linear ship steering dynamics using maximum likelihood parameter estimation, *SSPA Report No. 75*, Gothenburg, Sweden.
- Åström, K.J., and B. Wittenmark, 1995. *Adaptive Control*, Addison-Wesley.
- Balasuriya, B.A.A.P., and P.R.P. Hoole, 1995. Feedforward neural network controller for ship steering, *Proceedings 3<sup>rd</sup> IFAC Workshop on Control Applications in Marine Systems*, Trondheim, Norway, 400-404.
- Barnard, E., 1992. Optimization for training neural nets, *IEEE Transactions on Neural Networks* **3**(2), 232-240.
- Barnes, C., S. Brown, G. Flake, R. Jones, M. O'Rourke, Y.C. Lee, 1991. Applications of neural networks to process control and modelling, *Proceedings of 1991 International Conference on Artificial Neural Networks (ICANN'91)*, Amsterdam, Netherlands. Vol. 1, 823-828.
- Bashkirov, O.A., E.M. Braverman, and I.B. Muchnik, 1964. Potential function algorithms for pattern recognition learning machines. *Automation and Remote Control* **25**, 629-631.
- Battitti, R., 1992. First and second order methods for learning: Between steepest descent and Newton's method, *Neural Computation* **4**(2), 141-166.
- Bavarian, B., 1988. Introduction to neural networks for intelligent control, *IEEE Control Systems magazine* **8**(2), 3-7.
- Baxt, W., 1992. The applications of the artificial neural network to clinical decision making, *Conference on Neural Information Processing Systems - Natural and Synthetic*, Denver, CO.
- Bech, M.I., 1966. Alternative procedure for carrying out spiral tests, *Scandinavian Ship Technical Symposium*.
- Bech, M.I., 1968. The reverse spiral test as applied to large ships, *Shipping World and Shipbuilder*, 1753-1754.

- Bech, M.I., and L.W. Smith, 1969. Analogue simulation of ship manoeuvres, *Technical Report Hy-14*, Hydro- and Aerodynamics Laboratory, Lyngby, Denmark.
- Bech, M.I., 1972. Some aspects of the stability of automatic course control of ships. *The Journal of Mechanical Engineering Science* **14**(7), Supplementary Issue, 123-131.
- Bennet, S., 1993. *A History of Control Engineering 1930 - 1955*. Peter Peregrines.
- Billings, S.A., and S. Chen, 1989. Extended model set, global data and threshold model identification of severely nonlinear systems, *International Journal of Control* **50**(5), 1897-1923.
- Billings, S.A., H.B. Jamaluddin, and S. Chen, 1992. Properties of neural networks with applications to modelling nonlinear dynamical systems, *International Journal of Control* **55**(1), 193-224.
- Billings, S.A., and W.S.F. Voon, 1987. Piecewise linear identification of nonlinear systems, *International Journal of Control* **46**, 215-235.
- Björck, A., 1967. Solving linear least squares problems by Gram-Schmidt orthogonalization, *Nordisk Tidskr. Information-Be-Handling* **7**, 1-21.
- Blanke, M., 1981. *Ship Propulsion Losses Related to Automatic Steering and Prime Mover Control*, PhD Thesis, The Technical University of Denmark, Lyngby.
- Brink, A.W., G. Baas, A. Tiano, and E. Volta, 1978. Adaptive course keeping of a super tanker and a container ship - A simulation study, *Proceedings of the 5<sup>th</sup> Ship Control Symposium*, Annapolis, Maryland.
- Broome, D.R., and T.H. Lambert, 1978. An optimizing function for adaptive ships and autopilots, *Proceedings of the 5<sup>th</sup> Ship Control Symposium*, Annapolis, Maryland.
- Broomhead, D.S., and D. Lowe, 1988. Multivariable functional interpolation and adaptive networks, *Complex Systems* **2**, 321-355.

- Bryson, A.E., and Ho, Y.C., 1975. *Applied Optimal Control*, Hemisphere Publishing, New York.
- Bulsari, A., and H. Saxon, 1991. System identification of a biochemical process using feedforward neural networks, *Neurocomputing* **3**(3), 125-133.
- Burcher, R.K., 1972. Developments in ship manoeuvrability, *Transactions of the Royal Institute of Naval Architects* **114**, 1-32.
- Burns, R.S., 1990. The design, development and implementation of an optimal guidance system for ships in confined waters, *Proceedings of the 9<sup>th</sup> Ship Control Symposium*, Vol. 3, Bethesda, USA, 3.386-3.401.
- Burns, R.S., 1995. The use of artificial neural networks for the intelligent optimal control of surface ships, *IEEE Journal of Oceanic Engineering* **20**(1), 65-72.
- Charalambous, C., 1992. Conjugate gradient algorithm for efficient training of artificial neural networks, *IEE Proceedings* **139**(3), 301-310.
- Chen, S., and S.A. Billings, 1992. Neural networks for nonlinear dynamic system modelling and identification, *International Journal of Control, Special Issue on Intelligent Control* **52**, 1327-1350.
- Chen, S., and S.A. Billings, 1994. Neural networks for nonlinear dynamic system modelling and identification. In *Advances in Intelligent Control*, (C.J. Harris, Ed.), Taylor & Francis, London, 85-112.
- Chen, S., S.A. Billings, and P.M. Grant, 1990. Non-linear system identification using neural networks, *International Journal of Control* **51**(6), 1191-1214.
- Chen, S., S.A. Billings, and W. Luo, 1989. Orthogonal least squares methods and their application to non-linear system identification, *International Journal of Control* **50**(5), 1873-1896.
- Chen, S., C.F.N. Cowan, and P.M. Grant, 1991. Orthogonal least squares algorithm for radial basis function networks, *IEEE Transactions on Neural Networks* **2**(2), 302-309.

- Chislett, M.S., and J. Strøm-Tejsen, 1965a. Planar motion mechanism tests and full scale steering and manoeuvring predictions for a mariner class vessel, *Technical Report Hy-5*, Hydro- and Aerodynamics Laboratory, Lyngby, Denmark.
- Chislett, M.S., and J. Strøm-Tejsen, 1965b. Planar motion mechanism tests and full scale steering and manoeuvring predictions of a mariner class vessel, *Technical Report Hy-6*, Hydro- and Aerodynamics Laboratory, Lyngby, Denmark.
- Cichocki, A., and R. Unbehauen, 1993. *Neural Networks for Optimization and Signal Processing*, Wiley.
- Clarke, D., 1987. Assessment of manoeuvring performance, *Ship Manoeuvrability, Proceedings of the International Conference on Ship Manoeuvrability, Prediction and Achievement*, Vol.1, Paper 2, The Royal Institution of Naval Architects, London.
- Cleveland, W.S., S.J. Devlin, and S. Grosse, 1988. Regression by local fitting, *Journal of Econometrics* **37**, 87-114.
- Cohen, M., H. Franco, N. Morgan, D. Rumelhart, and V. Abrash, 1993. Context- dependent multiple distribution phonetic modelling with MLPs. In *Advances in Neural Information Processing Systems* (S.J. Hanson, J.D. Cowan, and C.L. Giles, Eds.), 649-657, San Mateo, CA.
- Comstock, J.P. (Ed.), 1967. *Principles of Naval Architecture*, SNAME, Jersey City, USA.
- Crane, C.L., H. Eda, and A. Landsburg, 1989. Controllability. In (E.V. Lewis, Ed.): *Principles of Naval Architecture*, Vol. 3, SNAME.
- Cristi, R., F.A. Popoulias, and A.J. Healey, 1990. Adaptive sliding mode control of autonomous underwater vehicles in the dive plane, *IEEE Journal of Oceanic Engineering* **15**(3), 152-160.
- Cybenko, G., 1989. Approximation by superposition of a sigmoidal function, *Mathematics of Control, Signals and Systems* **2**, 303-314.

- Davidson, K.S.M., and L.I. Schiff, 1946. Turning and course keeping qualities, *Transactions SNAME* **54**, 152-200.
- Desanj, D.S., M.J. Grimble, and M.R. Katebi, 1997. State-space adaptive  $H_\infty$  controller with application to a ship control system, *Transactions of the Institute of Measurement and Control* **19**(3), 139-153.
- Dieudonné J., 1953. Collected French papers on the stability of route of ships at sea 1949-1950. (Translated by H.E. Saunders and E.N. Labouvie), *Technical Report DTMB-246*, Naval Ship Research and Development Centre, Washington, DC.
- Donaldson, N de N., H. Gollee, K.J. Hunt, J.C. Jarvis, and M.K.N. Kwende, 1995. A radial basis function model of muscle stimulated with irregular inter-pulse intervals, *Medical Engineering Physics* **17**(6), 431-441.
- Dougherty, F., and G. Woolweaver, 1990. At-Sea testing of an unmanned underwater vehicle flight control system, *Symposium on Autonomous Underwater Technology*, Washington, DC, 65-68.
- Dyne, G., and P. Trägårdh, 1975. Simuleringsmodell för 350000 tdw tanker i fullast- och ballastkonditioner på djupt vatten, *Report 2075-1*, Swedish State Shipbuilding Experimental Tank (SSPA), Gothenburg, Sweden.
- Edwards, C., and S.K. Spurgeon, 1998. *Sliding Mode Control: Theory and Applications*, Taylor & Francis, London.
- Ekdahl, K.I., and O. Henriksson, 1970. *Om regulatorer för maxima ekonomisk styrning av fartyg*, MSc Thesis, Report RE-83, Division of Automatic Control, Lund Institute of Technology.
- Elanayar, S., and Y.C. Shin, 1994. Radial basis function neural network for approximation and estimation of non-linear stochastic dynamic systems, *IEEE Transactions on Neural Networks* **5**(4), 594-603.
- Endo, M., J. Van Amerongen, and A.W.P. Bakkers, 1989. Applicability of neural networks to ship steering, *Proceedings of the IFAC Workshop on Control Applications in Marine Systems (CAMS'89)*, The Technical University of Lyngby, 221-232.

- Erives, H., Thompson, W., and R. Parra-Loera, 1996. Evolved radial basis function networks for identification and control of dynamical systems, *Proceedings of the SPIE* **2755**, 375-383.
- Fathala, G., and M. Farsi, 1997. Modelling and control of a brushless DC motor using RBF neural network, *Proceedings of the 12<sup>th</sup> International Conference on Systems Engineering (ICSE'97)*, Coventry, UK, 248-251.
- Flobakk, T., 1983. *Parameter estimation techniques applied to ship manoeuvring tests in model scale*, PhD Thesis, Department of Engineering Cybernetics, The Norwegian Institute of Technology, Trondheim, Norway.
- Flower, J.O., and J.R. Sparrius, 1986. The design of autopilot using the pseudo derivative feedback algorithm. *International Shipbuilding Progress* **33**(337), 10-20.
- Foss, B.A., and T.A. Johansen, 1993. On local and fuzzy modelling, *Proceedings 3<sup>rd</sup> International Conference on Industrial Fuzzy Control and Intelligent Systems*, Houston, Texas.
- Foss, B.A., T.A. Johansen, and A.V. Sørensen, 1995. Nonlinear predictive control using local models - applied to a batch fermentation process, *Control Engineering Practice* **3**, 389-396.
- Fossen, T.I., 1993. High performance autopilot with wave filter, *10<sup>th</sup> International Ship Control Systems Symposium (SCSS'93)*, Ottawa, Canada, 2\_271 - 2\_285.
- Fossen, T.I., 1994. *Guidance and Control of Ocean Vehicles*, John Wiley & Sons.
- Fossen, T.I., and Å.Grøven, 1998. Nonlinear output feedback control of dynamically positioned ships using vectorial observer backstepping, *IEEE Transactions on Control Systems Technology* **TCST-6**(1), 121-128.
- Fossen, T.I., and M.J. Paulsen, 1992. Adaptive feedback linearization applied to steering of ships. *Proceedings of the 1<sup>st</sup> IEEE Conference on Control Applications*, Dayton, Ohio, 1088-1093.

- Fossen, T.I., and M.J. Paulsen, 1993. Adaptive feedback linearization applied to steering of ships, *Modelling, Identification and Control* **14**(4), 229-237.
- Friedmann, J.H., 1991. Multivariable adaptive regression Splines, *The Annals of Statistics* **19**, 1-141.
- Fujino, M., 1968. Experimental studies on ship manoeuvrability in restricted water - part I. *International Ship Building Progress* **15**, 279-301.
- Fukushima, K., 1975. Cognitron: A self-organizing multilayered neural network, *Biological Cybernetics* **20**, 121-136.
- Fukushima, K. 1980. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position, *Biological Cybernetics* **36**, 193-202.
- Funahashi, K., 1989. On the approximate realization of continuous mappings by neural networks, *Neural Networks* **2**, 183-192.
- Garcia, R.F., 1993. Fuzzy rule based adaptive control method applied to ship steering, *Proceedings of the International Conference on Systems, Man and Cybernetics. Systems Engineering in the Service of Humans*. 668-673.
- Garcia, R.F., and F.J. Castelo, 1995. Adaptive PID controller applied to marine DP control using frequency analysis, *Proceedings 3<sup>rd</sup> IFAC Workshop on Control Applications in Marine Systems*, Trondheim, Norway, 356-361.
- Gawthrop, P.J., 1996. Continuous time local model networks. In *Neural Adaptive Control Technology* (R.W. Zbikowski and K.J. Hunt, Eds.), 41-70, World Scientific, Singapore.
- Girosi, F., and T. Poggio, 1989. Representation properties of networks: Kolmogorov's theorem is irrelevant, *Neural Computation* **1**, 465-469.
- Girosi, F., and T. Poggio, 1990. Networks and the best approximation property, *Biological Cybernetics* **63**(3), 169-176.

- Goclowski, J., and A. Gelb, 1966. Dynamics of an automatic ship steering system, *IEEE Transactions on Automatic Control* **AC-11**(3), 513-524.
- Gollee, H., and K.J. Hunt, 1997. Nonlinear modelling and control of electrically simulated muscle: a local model approach, *International Journal of Control* **68**(6), 1259-1288.
- Gollee, H., K.J. Hunt, N. de N. Donaldson, and C.J. Jarvis, 1997. Modelling of electrically simulated muscle. In *Multiple Modelling Approaches to Modelling and Control* (R. Murray-Smith and T.A. Johansen, Eds.), Chapter 3, Taylor and Francis.
- Gollee, H., and D.J. Murray-Smith, 1997. Validation of local model networks for electrically simulated muscle, *Proceedings 3<sup>rd</sup> International Conference on Engineering Applications of Neural networks (EANN'97)*, Stockholm, Sweden, 191-194.
- Gorinevsky, D., A. Kapitanovsky, and A. Goldenberg, 1996. Radial basis function architecture for non holonomic motion planning and control of free flying manipulators, *IEEE Transactions on Robotics and Automation* **12**(3), 491-496.
- Gøttsche, Th., K.J. Hunt, and T.A. Johansen, 1998. Nonlinear dynamics modelling via operating regime decomposition, *IMACS Journal on Mathematics and Computers in Simulation* **46**(5/6), 543.
- Grossberg, S., 1972. Neural expectation: Cerebellar and retinal analog of cells fired by learnable or unlearned pattern classes, *Kybernetik* **10**, 49-57.
- Grossberg, S., 1976. Adaptive pattern classification and universal recording: I. Parallel development and coding of neural detectors, *Biological Cybernetics* **23**, 121-134.
- Grossberg, S., 1980. How does a brain build a cognitive code? *Psychological Review* **87**, 1-51.
- Grossberg, S., 1982. *Studies of Mind and Brain*, Reid.

- Guyon, I., 1991. Applications of neural networks to character recognition, *International Journal of Pattern Recognition and Artificial Intelligence* **5**, 353-382.
- Hagan, M.T., and M.B. Minhaj, 1994. Training feedforward networks with the Marquardt algorithm, *IEEE Transactions on Neural Networks* **5**(6), 989-993.
- Harris, C., C.G. Moore, and M. Brown, 1993. *Intelligent Control: Aspects of Fuzzy Logic and Neural Nets*. World Scientific.
- Hartman, E., and J.D. Keeler, 1991. Predicting the future: advantages of the semilocal units, *Neural Computation* **3**(4), 566-578.
- Hartman, E., J.D. Keeler, and J.M. Kowalski, 1990. Layered neural networks with Gaussian hidden units as universal approximations, *Neural Computation* **2**(2), 210-215.
- Haykin, S., 1994. *Neural Networks - A Comprehensive Foundation*, Macmillan.
- Hazarika, N., A.C. Tsoi, and Sergejew, 1998. Modelling and classification of EEG signals in psychiatric disorders using multilayer perceptrons, *Proceedings of the 4<sup>th</sup> International Conference on Engineering Applications of Neural Networks (EANN'98)*, Gibraltar, 284-287.
- Healey, A.J., and D. Lienard, 1993. Multivariable sliding mode control for autonomous diving and steering of unmanned underwater vehicles, *IEEE Journal of Oceanic Engineering* **18**, 327-339.
- Healey, A.J. and D.B. Marco, 1992. Slow speed flight control of autonomous underwater vehicles: Experimental results with the NPS AUV II, *Proceedings of the 2<sup>nd</sup> International Offshore and Polar Engineering Conference (ISOPE)*, San Francisco, CA, 523-532.
- Hebb, D.O., 1949. *The Organization of Behaviour: A Neuropsychological Theory*, Wiley.
- Hecht-Nielsen, R., 1990. *Neurocomputing*, Addison-Wesley.

- Herther, J.C., F.E. Wanock, K.W. Howard, and W. Vanvelde, 1980. Digipilot - A self-adjusting digital autopilot for better manoeuvring and improved course and track keeping, *Proceedings of the International Symposium on Ship Steering Automatic Control*, Genoa, Italy.
- Hertz, J., A. Krogh, and R.G. Palmer, 1991. *Introduction to the Theory of Neural Computation*, Addison-Wesley.
- Hilhorst, R.A., J. Van Amerongen, and P. Löhnberg, 1991. Intelligent adaptive control of mode switch processes. *Proceedings IFAC International Symposium on Intelligent Tuning and Adaptive Control*, Singapore.
- Hlaváčková, K., 1995. An upper estimate of the error of approximation of continuous multivariable functions by KBF networks, *Proceedings of the 3<sup>rd</sup> European Symposium on Artificial Neural Networks*, Brussels, 333-340.
- Holzhüter, T., 1989. Robust identification in an adaptive track controller for ships. *Proceedings of the 3<sup>rd</sup> IFAC Symposium on Adaptive Systems in Control and Signal Processing*, Glasgow, 275-280.
- Holzhüter, T., and R. Schultze, 1996. Operating experience with a high precision track controller for commercial ships, *Control Engineering Practice* 4(3), 343-350.
- Honderd, G., and J.E.W. Winkelman, 1972. An adaptive autopilot for ships, *Proceedings of the 3<sup>rd</sup> Ship Control Symposium*, Bath, UK, 1-16.
- Hopfield, J.J., 1982. Neural networks and physical systems with emergent collective computational abilities, *Proceedings of the National Academy of Science of the USA* 79, 2554-2558.
- Hornik, K, M. Stinchcombe, and H. White, 1989. Multilayer feedforward networks are universal approximators, *Neural Networks* 2, 359-366.
- Hornik, K., M. Stinchcombe, and H. White, 1990. Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks, *Neural Networks* 3, 551-560.

- Hotland, A.G., A.J. Morris, and G.A. Montague, 1992. Radial basis function networks applied to process control, *American Control Conference*, Vol. 1, Evanston, IL, USA, 480-484.
- Hunt, K.J., R. Haas, and R. Murray-Smith, 1996a. Extending the functional equivalence of radial basis function networks and fuzzy inference systems, *IEEE Transactions on Neural Networks* **7**(3), 776-781.
- Hunt, K.J., J.C. Kalkkuhl, H. Fritz, and T.A. Johansen, 1996b. Constructive empirical Modelling of longitudinal vehicle dynamics using local model networks, *Control Engineering Practice* **4**(2), 167-178.
- Hunt, K.J., and T.A. Johansen, 1997. Design and analysis of gain scheduled control using local model networks, *International Journal of Control* **66**, 619-651.
- Hunt, K.J., D. Sbarbaro, R. Zbikowski, and P.J. Gawthrop, 1992. Neural networks for control systems - a survey, *Automatica* **28**, 1083-1112.
- Hush, D.R., and B.G. Horne, 1993. Progress in supervised neural networks, *IEEE Signal Processing Magazine*, 8-38.
- Jacobs, R.A., M.I. Jordan, S.J. Nowlan, and G.E. Hinton, 1991. Adaptive mixtures of local experts, *Neural Computation* **3**(1), 79-87.
- Jia, X.L., and Q. Song, 1987. Self-tuning regulators used for ship course keeping, *Proceedings of the IFAC Symposium on Adaptive Systems in Control and Signal Processing*, Lund, Sweden.
- Johansen, T.A., and B.A. Foss, 1992a. A NARMAX model representation for adaptive control based on local models, *Modelling, Identification and Control* **13**(1), 25-39.
- Johansen, T.A., and B.A. Foss, 1992b. Nonlinear local model representation for adaptive systems, *Proceedings International Conference on Intelligent Control and Instrumentation*, Vol. 2, 677-682, Singapore.
- Johansen, T.A., and B.A. Foss, 1992c. Representing and learning unmodelled dynamics with neural network memories, *Proceedings American Control Conference*, Chicago, IL, 3037-3043.

- Johansen, T.A., and B.A. Foss, 1993. Constructing NARMAX models using ARMAX models, *International Journal of Control* **58**, 1125-1153.
- Johansen, T.A., and B.A. Foss, 1997. Operating regime based process modelling and identification, *Computers and Chemical Engineering* **21**, 159-176.
- Johansen, T.A., K.J. Hunt, and H. Fritz, 1998. Off-equilibrium linearization and design of gain scheduled control with application to vehicle speed control, *Control Engineering Practice* **6**, 167-180.
- Jones, R.D. et al., 1989. Function approximation and time series prediction with neural networks, *Technical Report No. 90-21*, Los Alamos National Lab., New Mexico.
- Jordan, M.I., and R.A. Jacobs, 1991. Hierarchies of adaptive experts. In *Advances in Neural Information Processing Systems 4* (J.E. Moody, S.J. Hanson, and R.P. Lippmann, Eds.), Morgan Kaufmann Publishers, San Mateo, CA.
- Jordan, M.I., and R.A. Jacobs, 1993. Hierarchical mixtures of experts and the EM algorithm, *Technical Report 9301*, MIT, Computational Cognitive Science.
- Kaasen, K.E., 1986. *Estimation of parameters in ship manoeuvring models from hull scale measurements*, Dr. ing Thesis, Department of Engineering Cybernetics, The Norwegian Institute of Technology, Trondheim, Norway.
- Kanamaru, H., and T. Sato, 1979. Adaptive autopilot system with minimum energy consumption, *Proceedings of the International Symposium on Ship Operation Automation*, Tokyo, Japan.
- Kartalopoulos, S.V., 1996. *Understanding Neural Networks and Fuzzy Logic: Basic Concepts and Applications*, IEEE.
- Katebi, M.R., and J.C. Byrne, 1988. LQG adaptive ship autopilot. *Transactions Inst. MC* **10(4)**, 187-197.

- Kaveli, T., 1993. ASMOD - an algorithm for adaptive spline modelling of observation data, *International Journal of Control* **58**(4), 947-967.
- Källström, C.G., 1979. *Identification and adaptive Control applied to Ship Steering*, PhD Thesis, Department of Automatic Control, Lund Institute of Technology, Sweden.
- Källström, C.G., and K.J. Åström, 1981. Experiences of system identification applied to ship steering, *Automatica* **17**, 187-198.
- Källström, C.G., K.J. Åström, N.E. Thorell, J. Eriksson, and L. Sten, 1979. Adaptive autopilots for tankers, *Automatica* **15**, 241-254.
- Kempf, G., 1932. Measurements of the propulsive and structural characteristics of ships, *Transactions of SNAME* **40**, 42-57.
- Kim, S.W., and J.J. Lee, 1996. Filter-error-learning neural networks for stable trajectory tracking control, *Mechatronics* **6**(2), 181-192.
- Kirkpatrick, S., C.D. Gelatt, Jr., and M.P. Vecchi, 1983. Optimization by Simulated Annealing, *Science* **220**, 671-680.
- Kohonen, T., 1982. Self-organized formation of topologically correct feature maps, *Biological Cybernetics* **43**, 59-69.
- Kollias, S., and D. Anastassiou, 1989. An adaptive least squares algorithm for the efficient training of artificial neural networks, *IEEE Transactions on Circuits and Systems* **36**(8), 1092-1101.
- Koyama, T., 1967. On the optimum automatic steering system of ships at sea, *J. Soc. Nav. Archit. Japan* **122**, 18-35.
- Koyama, T., 1972. Improvement of course stability and control by a subsidiary automatic control, *The Journal of Mechanical Engineering Science* **14**(7), Supplementary Issue, 132-141.
- Kruger, L., and D. Naunin, 1996. Advanced control of an induction motor based on neural and stochastic models, *Proceedings 7<sup>th</sup> International Power Electronics and Motion Control Conference*, Budapest, Hungary, 182-186.

- Kwakernaak, H., and R. Sivan, 1972. *Linear Optimal Control Systems*, Wiley.
- Lauvdal, T., and T.I. Fossen, 1995. A globally stable adaptive ship autopilot with wave filter using only yaw angle measurements, *Proceedings 3<sup>rd</sup> IFAC Workshop on Control Applications in Marine systems (CAMS'95)*, Trondheim, Norway.
- Layne, J.R., 1992. *Fuzzy Model Reference Learning Control*, Master's Thesis, Department of Electrical Engineering, The Ohio State University.
- Layne, J.R., and K.M. Passino, 1993. Fuzzy model reference learning control for cargo ship steering. *IEEE Control Systems Magazine* **13**(6), 23-34.
- Lee, S., and R.M. Kil, 1991. A Gaussian potential function network with hierarchically self-organizing learning, *Neural Networks* **4**, 207-224.
- Levenberg, K., 1944. A method for the solution of certain non-linear problems in least squares, *Quart. Appl. Math.* **2**, 164-168.
- Light, W., 1992. Ridge functions, sigmoidal functions and neural networks. In E.W. Cheney, C.K. Chui, and L.I. Schumaker (Eds.), *Approximation Theory*, Academic Press, Boston.
- Lightbody, G., P. O'Reilly, G.W. Irwin, K. Kelly, and J. McCormick, 1997. Neural modelling of chemical plant using MLP and B-Spline networks, *Control Engineering Practice* **5**(11), 1501-1515.
- Lim, C.C., and W. Forsythe, 1983. Autopilot for ship control, *Proceedings of the IEE, Part-D*, **130**, 281-294.
- Lippmann, R.P., 1987. An introduction to computing with neural nets, *IEEE ASSP Magazine* **4**, 4-22.
- Lu, X.R., and B.S. Zhang, 1987; An adaptive algorithm for ship's adaptive autopilot design, *Proceedings 10<sup>th</sup> World Congress on Automatic Control*, IFAC, FRG, 232-240.
- Luke, R.M., and F. West, 1960. An integrated steering system, *New England Section of SNAME*.

- Marquardt, D.W., 1963. An algorithm for least squares estimation of nonlinear parameters, *SIAM Journal of Appl. Math.* **11**, 431-441.
- McCallum, I.R., 1991. The effect of ship controllability on port design. In (M.M.A. Pourzanjani and G.N. Roberts, Eds.), *Modelling and Control of Marine Craft*, 252-266, Elsevier Applied Science, London.
- McCulloch, W.S., and W. Pitts, 1943. A logical calculus of the ideas immanent in nervous activity, *Bulletin of Mathematical Biophysics* **5**, 115-133.
- McGookin, E.W., 1993. *Sliding Mode Control of a Submarine*, MSc Thesis, Department of Electronics and Electrical Engineering, University of Glasgow, Glasgow, Scotland.
- McGookin, E.W., 1997. *Optimization of Sliding Mode Controllers for Marine Applications: A Study of Methods and Implementation Issues*, PhD Thesis, Department of Electronics and Electrical Engineering, University of Glasgow, Glasgow, Scotland.
- Merlo, P., and A. Tiano, 1975. Experiments about computer controlled ship steering. *Semana Internacional Sobre la Automatica en la Marina*, Barcelona, Spain.
- Millers, H.T., 1973. Modern control theory applied to ship steering, *Proceedings of the 1<sup>st</sup> IFAC/IFIP Symposium on Ship Operation Automation*, Oslo, Norway.
- Minorsky, N., 1922. Directional stability of automatically steered bodies, *Journal American Society of Naval Engineers* **34**, 280-309.
- Minorsky, N., 1930. Automatic steering tests, *Journal American Society of Naval Engineers* **42**, 285-310.
- Minsky, M.L., 1954. *Theory of neural-analog reinforcement systems and its application to the brain model problem*, PhD Thesis, Princeton University, Princeton, NJ.

- Minsky, M.L., and S.A. Papert, 1969. *Perceptrons*, MIT Press.
- Moody, J.E., 1991. Note on generalization, regularization and architecture selection in nonlinear learning systems, In (B.H. Juang, S.Y. Kung, and C.A. Kamm, Eds.), *Neural Networks for Signal Processing: Proceedings of the 1991 IEEE Workshop*, 1-10, IEEE Press.
- Moody, J.E., and C.J. Darkin, 1989. Fast learning in networks of locally tuned processing units, *Neural Computation* **1**, 281-294.
- Morari, M., and E. Zafiriou, 1989. *Robust Process Control*, Prentice Hall.
- Mort, N., and D.A. Linkins, 1981. Self-tuning controllers for surface ship course keeping and manoeuvring. In *Self-Tuning and Adaptive Control*, (C.J. Harris and S.A. Billings, Eds.), The Institute of Electrical Engineers.
- Motora, S., 1967. On the automatic steering and yawing in rough seas, *Journal of Soc. Nav. Archit. Japan* **122**.
- Motora, S., 1972. Manoeuvrability, state of the art, *Proceedings of the International Jubilee Meeting on the occasion of the 40<sup>th</sup> Anniversary of the Netherlands Ship Model Basin*, Netherlands Ship Model Basin, Wageningen, The Netherlands.
- Motora, S., and T. Koyoma, 1968. Some aspects of the stability of automatic steering of ships, *Japan Shipbuilding and Marine Engineering* **3**(4), 5-18.
- Mozayyani, N., and G. Vaucher, 1997. A spatio-temporal perceptron for on line hand written character recognition, *Proceedings International Conference on Artificial Neural Networks (ICANN'97)*, France, 325-330.
- Murray-Smith, R., 1992. A fractal radial basis function network for modelling, *Proceedings International Conference on Automation, Robotics, and Computer Vision*, 2.6.1 - 2.6.5.
- Murray-Smith, R., 1994. *A Local Model Network Approach to non-linear modelling*, PhD Thesis, University of Strathclyde, Glasgow, Scotland.
- Murray-Smith, R., and H. Gollee, 1994. A constructive learning algorithm for local model networks, *Proceedings of the IEEE Workshop on Computer*

*Intensive Methods in Control and Signal Processing*, Prague, Czech Republic, 21-29.

- Murray-Smith, R., and K.J. Hunt, 1995. Local model architectures for nonlinear modelling and control. In *Neural Network Engineering in Dynamic Control Systems* (K.J. Hunt, G.R. Irwin, and K. Warwick, Eds.), Advances in Industrial Control, 61-82, Springer-Verlag.
- Murray-Smith, R., and T.A. Johansen (Eds.), 1997. *Multiple Model Approaches to Modelling and Control*, Taylor & Francis.
- Murray-Smith, R., D. Neumerkel, and D. Sbarbaro-Hofer, 1992. Neural networks for modelling and control of a non-linear dynamic system, *IEEE Symposium on Intelligent Control*, Glasgow, 404-409.
- Müller, B., J. Reinhardt, M.T. Strickland, 1995. *Neural Networks - An Introduction*, Springer-Verlag.
- Narendra, K.S., and S. Mukhopadhyay, 1992. Intelligent control using neural networks, *IEEE Control Systems Magazine*, 11-18.
- Narendra, K.S., and K. Parthasarathy, 1990. Identification and control of dynamical systems using neural networks, *IEEE Transactions on Neural Networks*, 1(1), 4-27.
- Ng, K., and R.P. Lippmann, 1991. Practical characteristics of neural networks and conventional pattern classifiers. In *Advances in Neural Information Processing Systems 3* (R.P. Lippmann, J.E. Moody, and D.S. Touretzky, Eds.), 970-976, Morgan Kaufmann, San Mateo, CA.
- Nomoto, K., 1966. Response analysis of manoeuvrability and its application to ship design. *Research on the Manoeuvrability of Ships in Japan*, 60<sup>th</sup> Anniversary Series of the Society of Naval Architects of Japan, Vol. 11.
- Nomoto, K., 1972. Problems and requirements of directional stability and control of surface ships. *The Journal of Mechanical Engineering Science* 14(7), Supplementary Issue, 1-5.

- Nomoto, K., and T. Motoyama, 1966. Loss of propulsive power caused by yawing with particular reference to automatic steering. *Japan Shipbuilding and Marine Engineering* **94**.
- Nomoto, K., T. Tagushi, K. Hunda, and S. Hirano, 1957. On the steering qualities of ships, *International Ship Building Progress* **4**(35), 354-370.
- Norrbin, N.H., 1963. On the design and analysis of the zigzag test on base of quasi linear frequency response, *Technical Report B104-3*, The Swedish State Shipbuilding Experimental Tank (SSPA), Gothenburg, Sweden.
- Norrbin, N.H., 1965. Zig-Zag Provets Teknik och Analys, *Technical Report No. 12*, The Swedish State Shipbuilding Experimental Tank (SSPA), Gothenburg, Sweden.
- Norrbin, N.H., 1970. Theory and observation on the use of a mathematical model for ship manoeuvring in deep and confined waters, *Proceedings of the 8<sup>th</sup> Symposium on Naval Hydrodynamics*, Pasadena, California.
- Oguri, K., and A. Iwata, 1998. Biomedical applications of neural networks - analysis of changing oligosaccharide profiles in several diseases, *Proceedings of the 4<sup>th</sup> International Conference on Engineering Applications of Neural networks (EANN'98)*, Gibraltar, 601-608.
- Ohtsu, K., M. Horigome, and G. Kitagawa, 1979. A new ship's autopilot design through a stochastic model, *Automatica* **15**, 255-268.
- Ohtsu, K., and G. Kitagawa, 1978. An advanced ship autopilot system by a stochastic model, *Proceedings of the 5<sup>th</sup> Ship Control Symposium*, Annapolis, Maryland, USA, C2\_1 - C2\_11.
- Oldenburg, J., 1975. Experiments with a new adaptive autopilot intended for controlled turns as well as for straight course keeping, *Proceedings of the 4<sup>th</sup> Ship Control System Symposium*, The Hague, The Netherlands.
- Pantaleòn-Prieto, C.J., F.D. de Maria, and A. Figueiras-Vidal, 1993. On training RBF networks. In *Neural Networks and their Industrial & Cognitive Applications*, Conference proceedings, Nimes, France, 279-288.

- Pao, Y.H., 1989. *Adaptive Pattern Recognition and Neural Networks*, Addison-Wesley.
- Park, J., and I.W. Sandberg, 1991. Universal approximation using radial basis function networks, *Neural Computation* **5**, 305-316.
- Park, J., and I.W. Sandberg, 1993. Universal approximation using radial basis function networks, *Neural Computation* **3**(2), 246-257.
- Parker, D.B., 1985. Learning logic: Casting the cortex of the human brain in Silicon. *Technical Report TR-47*, Centre for Computational Research in Economics and Management Science, MIT, Cambridge, MA.
- Parson, M.G., A.C. Chubb, and Y. Cao, 1995. An assessment of fuzzy logic vessel path control, *IEEE Journal of Oceanic Engineering* **20**(4), 276-284.
- Paulsen, M.J., O. Egeland, and T.I. Fossen, 1994. An output feedback controller with wave filter for marine vehicles, *Proceedings of the American Control Conference*, Baltimore, Maryland, 2202-2206.
- Poggio, T., and F. Girosi, 1990. Networks for approximation and learning, *Proceedings of the IEEE* **78**, 1481-1497.
- Pottman, M., and H.P. Jorgl, 1993. Radial basis function networks for nonlinear process control - an internal model control approach, *Elektrotechnik and Informationstechnik* **110**(7,8), 336-341.
- Powell, M.J.D., 1985. Radial basis functions for multivariable interpolation: A review, *Proceedings IMA Conference on Algorithms for the Approximation of Functions and Data*, RMCS, Shrivenham, UK, 143-167.
- Priestley, M.B., 1988. *Non-linear and Non-stationary Time Series Analysis*, Academic press.
- Rawson, K.J., and E.C. Tupper, 1983. *Basic Ship Theory*, Longman, London.
- Reid, R.E., and V.E. Williams, 1978. A new ship autopilot design criterion for improved heavy weather steering, *Proceedings of the 5<sup>th</sup> Ship Control Systems Symposium*, Annapolis, Maryland, C1-1 - C1-65.

- Richter, R., and R.S. Burns, 1993. An artificial neural network autopilot for small vessels, *Proceedings of the United Kingdom Simulation Society (UKSS'93)*, Edinburgh, 168-172.
- Rivera, D.E., M.Morari, and S. Skogestad, 1986. Internal Model Control. 4. PID Controller design, *Ind. Eng. Chem. Process Des. Dev.* **25**, 252-265.
- Robinson, D.A., 1992. Signal processing by neural networks in the control of eye movements. In *Computational Neuroscience Symposium*, 73-78, Indiana University-Purdue University at Indianapolis.
- Rosenblatt, F., 1958. The Perceptron: A probabilistic model for information storage and organization in the brain, *Psychological Review* **65**, 386-408.
- Rosenblatt, F., 1960a. Perceptron simulation experiments, *Proceedings of the Institute of Radio Engineers* **48**, 301-309.
- Rosenblatt, F., 1960b. On the convergence of reinforcement procedures in simple perceptrons, *Report VG-1196-G-4*, Cornell Aeronautical Laboratory, Buffalo, NY.
- Rosenblatt, F., 1962. *Principles of Neurodynamics*, Spartan Books, Washington, DC.
- Röscheisen, R. Hoffmann, and V. Tresp, 1992. Neural control of rolling mills: Incorporating domain theories to overcome data deficiency. In *Advances in Neural Information Processing*, Vol. 4, 659-666, Morgan Kaufmann Publishers, San Mateo CA.
- Rubio, F.R., and M.J. López, 1993. Adaptive optimal control of ship steering autopilots, *Proceedings of the 12<sup>th</sup> Triennial World Congress*, Sydney, Australia, 1017-1020.
- Rumelhart, D.E., and J.L. McClelland, (Eds.), 1986a. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations*. Vol. 1, MIT Press.
- Rumelhart, D.E., and J.L. McClelland, (Eds.), 1986b. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Psychological and Biological Models*. Vol. 2, MIT Press.

- Saha, A., J. Christian, D.S. Tang, and C.L. Wu, 1991. Oriented non-radial basis functions for image coding and analysis. In *Advances in Neural Information Processing Systems 3*, (R.P. Lippmann, J.E. Moody, and D.S. Touretzky, Eds.), 728-734, Morgan Kaufmann Publishers, San Mateo, CA.
- Sbarbaro, D., 1992. *Connectionist Feedforward Networks for Control of Nonlinear Systems*, PhD Thesis, Department of Mechanical Engineering, University of Glasgow, Glasgow, Scotland.
- Sbarbaro, D., and T.A. Johansen, 1997. Multiple local Laguerre models for modelling nonlinear dynamic systems of the wiener class, *Proceedings IEE*, Part-D.
- Schilling, A.C., 1976. Economics for autopilot steering using an IBM system/7 computer. *Proceedings International Symposium on Ship Operation Automation*, Washington, DC.
- Simensen, R., and D.J. Murray-Smith, 1995. Ship steering control by neural networks using feedback linearization control laws, *Proceedings of IFAC/IMACS International Workshop on Artificial Intelligence in Real Time Control*, Bled, Slovenia, 269-274.
- Singhal, S., and L. Wu, 1989. Training multilayer perceptrons with the extended Kalman algorithm. In *Advances in Neural Information Processing Systems I*, (D.S. Touretzky, Ed.), 133-140, Morgan Kaufmann Publishers, San Mateo, CA.
- Skeppstedt, A., 1988. *Construction of Composite Models from large data sets*, PhD Thesis, University of Linköping.
- Skeppstedt, A., L. Ljung, and M. Millnert, 1992. Construction of composite models from large data sets, *International Journal of Control* **55**(1), 141-152.
- Sperry, E., 1922. Automatic Steering, *Transactions SNAME*, 53-57.

- SNAME, 1950. The Society of Naval Architects and Marine Engineers. Nomenclature for Treating the Motion of a Submerged Body Through a Fluid, *Technical and Research Bulletin No. 1-5*.
- Stinchcombe, M., and H. White, 1989. Universal approximation using feedforward networks with non-sigmoidal hidden layer activation functions, *Proceedings of the International Joint Conference on Neural Networks*, Washington, DC, New York, 613-617.
- Stokbro, K., D.K. Umberger, and J.A. Hertz, 1990. Exploiting neurons with localized receptive fields to learn chaos, *Complex Systems* **4**, 603-622.
- Strøm-Tejse, J., and M.S. Chislett, 1966. A model testing technique and method of analysis for the prediction of steering and manoeuvring qualities of surface ships, *Proceedings 6<sup>th</sup> Symposium on Naval Hydrodynamics*.
- Sugeno, M., and G.T. Kang, 1988. Structure identification of fuzzy models, *Fuzzy Sets and Systems* **26**, 15-33.
- Sugimoto, A., and T. Kojima, 1978. A new autopilot system with condition adaptivity, *Proceedings of the 5<sup>th</sup> Ship Control System Symposium*, Annapolis, Maryland.
- Tadeusiewicz, R., W. Wszolek, and M. Modrzejewski, 1998. The evaluation of speech deformation treated for larynx cancer using the neural network and pattern recognition methods, *Proceedings 4<sup>th</sup> International Conference on Engineering Applications of Neural Networks (EANN'98)*, Gibraltar, 613-616.
- Takagi, T., and M. Sugeno, 1985. Fuzzy identification of systems and its applications for modelling and control, *IEEE Transactions on Systems, Man and Cybernetics* **15**, 116-132.
- Tetley, L., and D. Calcutt, 1991. *Electronic Aids to Navigation*, Edward Arnold.
- Thompson, R.F., 1993. *The Brain: A Neuroscience Primer*, W.H. Freeman and Company, San Francisco.

- Tiano, A., 1976. Identification and control of the ship steering process, *Ship Operation and Simulation, 2<sup>nd</sup> International Symposium*, Washington, 573-580.
- Tiano, A., E. Volta, A.W. Brinks, and T.W. Verbruggen, 1980. Adaptive control of large ships in nonstationary conditions - A simulation study. *Proceedings of the International Symposium on Ship Steering Automatic Control*, Genoa, Italy.
- Tollenaere, T., 1990. SuperSAB: Fast adaptive back-propagation with good scaling properties, *Neural Networks* **3**(5), 561-573.
- Tomera, M., and L. Morawski, 1996. Neural network based fuzzy logic marine autopilot. *Proceedings 3<sup>rd</sup> International Symposium on Methods and Models in Automation and Robotics*, Miedzyzdroje, Poland, 1207-1212.
- Tong, H., and K.S. Lim, 1980. Threshold autoregression, limit cycles and cyclical data, *J. Royal Stat. Soc.* B42, 245-292.
- Tulunay, E., 1991. Introduction to neural networks and their application to process control. In *Neural Networks: Advances and Applications* (E. Gelenbe, Ed.). 241-273, Elsevier Science Publishers B.V. (North Holland).
- Tulunay, E., S. Aslan, I. Topalli, E. Cetinkaya, and B. Kuyucu, 1998. Neural net based controller for juice extraction process in sugar industry, *Proceedings 4<sup>th</sup> International Conference on Engineering Applications of Neural Networks (EANN'98)*, 629-631, Gibraltar.
- Unar, M.A., 1995. *Multi-Loop PID Control Design*, MSc Thesis, University of Glasgow, Glasgow, Scotland.
- Unar, M.A., and D.J. Murray-Smith, 1997a. Artificial neural networks for ship steering control systems, *Proceedings 3<sup>rd</sup> International Conference on Engineering Applications of Neural Networks (EANN'97)*, Stockholm, Sweden, 87-90.

- Unar, M.A., and D.J. Murray-Smith, 1997b. Radial basis function networks for ship steering Control, *Proceedings of the 12<sup>th</sup> International Conference on Systems Engineering*, Vol. 2, 700-705, Coventry, UK.
- Unar, M.A., and D.J. Murray-Smith, 1999. Automatic steering of ships using neural networks, *International Journal of Adaptive Control and Signal Processing - Special Issue on Neural Network Applications in Control*, To Appear.
- Unar, M.A., D.J. Murray-Smith, and G.J. Gray, 1998. Neural network applications in ship steering control, *Proceedings 4<sup>th</sup> International Conference on Engineering Applications of Neural Networks (EANN'98)*, Gibraltar, 122-125.
- Unar, M.A., D.J. Murray-Smith, and Syed F.M. Shah, 1996. Designing and tuning of fixed structure PID controllers - A survey, *Technical Report No. CSC-96016*, Centre for Systems and Control and Department of Electronics and Electrical Engineering, University of Glasgow, Glasgow, Scotland.
- Utkin, V.I., 1992. *Sliding Mode in Control and Optimization*, Springer-Verlog.
- Vahedipour, A., J.O. Flower, and M.M.A. Pourzanjani, 1990. Pseudo derivative feedback design study of an autopilot with reference to nonlinearities and time delay effects, *Proceedings of the International Conference on Modelling and Control of Marine Craft*, Exeter, UK, 88-105.
- Van Amerongen, J., 1982. *Adaptive Steering of Ships - A Model Reference Approach to Improved manoeuvring and Economical Course Keeping*, PhD. Thesis, Delft University of Technology, The Netherlands.
- Van Amerongen, J., 1984. Adaptive steering of ships - A model reference approach, *Automatica* **20**(1), 3-14.
- Van Amerongen, J., and A.J. Udink Ten Cate, 1975. Model reference adaptive autopilots for ships, *Automatica* **11**, 441-449.

- Van Berlekom, W.B., and T.A. Goddard, 1972. Manoeuvring of large tankers, *Transactions SNAME* **80**, 264-298.
- Van Leeuwen, G., 1972. Some aspects of prediction and simulation of manoeuvres, *The Journal of Mechanical Engineering Science* **14**(7), Supplementary Issue, 108-144.
- Wang, L.X., 1994. *Adaptive Fuzzy Systems and Control: Design and Stability Analysis*, Prentice Hall.
- Werbos, P.J., 1974. *Beyond regression: New tools for prediction and analysis in the behavioural sciences*, PhD Thesis, Harvard University, Cambridge, MA.
- Werntges, H.W., 1993. Partition of unity improves neural function approximation, *Proceedings of the IEEE International Conference on Neural networks*, Vol. 2, 914-918, San Francisco, CA.
- Widrow, B., and M.E. Hoff, Jr., 1960. Adaptive switching circuits, *IRE WESCON Convention Record*, 96-104.
- Widrow, B., and M.A. Lehr, 1990. 30 years of adaptive neural networks: Perceptron, madaline, and back-propagation, *Proceedings of the IEEE* **78**, 1415-1442.
- Witt, N.A.J., R. Sutton, and K.M. Miller, 1994. Recent technological advances in the control and guidance of ships, *Journal of Navigation* **47**(2), 236-258.
- Witt, N.A.J., R. Sutton, and K.M. Miller, 1995. A track keeping neural network controller for ship guidance. *Proceedings 3<sup>rd</sup> IFAC Workshop on Control Applications in Marine Systems (CAMS'95)*, Trondheim, Norway, 385-392.
- Yoerger, D.R., and J.J.E. Slotine, 1984. Nonlinear trajectory control of autonomous underwater vehicles using the sliding methodology, *Proceedings of the ROV'84 Conference*, 245-251.

- Yoerger, D.R., and J.J.E. Slotine, 1985. Robust trajectory control of underwater vehicles, *IEEE Journal of Oceanic Engineering* **10**(4), 462-470.
- Yoerger, D.R., and J.J.E. Slotine, 1991. Adaptive sliding control of an experimental underwater vehicle, *Proceedings of IEEE International Conference on Robotics and Automation*, Sacramento, California, 2746-2751.
- Yoerger, D.R., and J.J.E. Slotine, 1995. Robust trajectory control of underwater vehicles, *IEEE Journal of Oceanic Engineering* **10**(4), 462-470.
- Zhang, Y., P. Sen, and G.E. Hearn, 1995. An on-line trained adaptive neural controller, *IEEE Control Systems*, 67-75, October 1995.
- Zhang, Y., P. Sen, and G.E. Hearn, 1996. Neural network approach to ship track keeping control, *IEEE Journal of Oceanic Engineering* **21**(4), 513-527.
- Ziegler, J.G., and N.B. Nichols, 1942. Optimum settings for automatic controllers, *Transactions ASME* **64**, 759-768.
- Ziegler, J.G., and N.B. Nichols, 1943. Process lags in automatic control circuits, *Transactions ASME* **65**(5), 433-444.
- Zhou, G., and J. Si, 1998. Advanced neural network training algorithm with reduced complexity based on Jacobian deficiency, *IEEE Transactions on Neural Networks* **9**(3), 448-453.
- Zuidweg, J.K., 1970. *Automatic Guidance of Ships as a Control Problem*, PhD Thesis, Delft University of Technology, The Netherlands.

## Appendix A: Internal Model Control

Internal model control was introduced by Morari and co-workers in the 1980s [Morari and Zafiriou, 1989; Rivera et al., 1986]. This control system design method is based on an assumed process model and relates the controller settings to the model parameters in a straight forward manner.

The IMC approach is based on the simplified block diagram shown in Figure A2. Transfer function  $G$  denotes the actual process. A process model

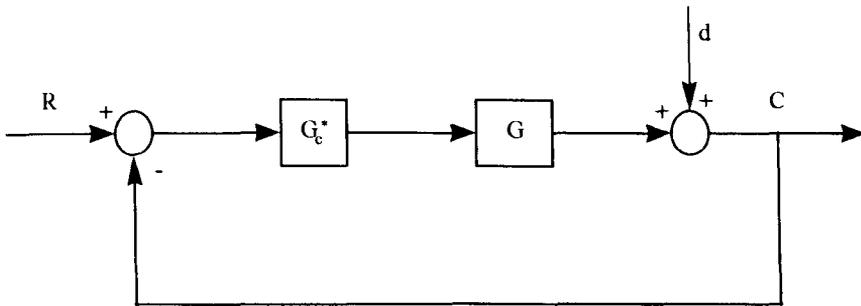


Figure A1: Conventional feedback control

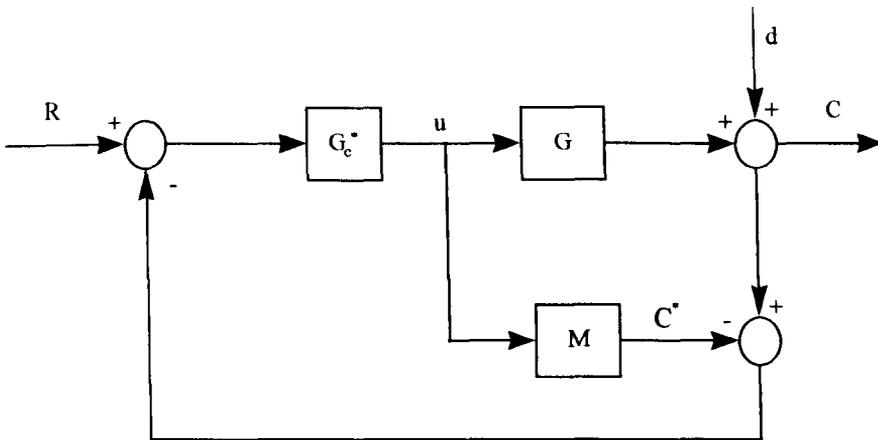


Figure A2: Internal model control

$M$  and the controller output  $u$  are used to calculate a model response  $C^*$ . The model response is subtracted from the actual response  $C$  and the difference  $C - C^*$  is used as the input signal to the controller which has the transfer

function  $G_c^*$ . In general,  $C \neq C^*$  due to modelling errors (i.e.  $M \neq G$ ) and unknown disturbances ( $d \neq 0$ ) that are not accounted for in the model.

Comparing IMC scheme of Figure A2 and the conventional feedback control system of Figure A1, we see that the diagrams will be identical if controller  $G_c$  and  $G_c^*$  satisfy the relation

$$G_c = \frac{G_c^*}{1 - G_c^*M} \quad (\text{A.1})$$

Thus, an IMC controller  $G_c^*$  can be replaced as a standard feedback controller  $G_c$  using (A.1).

From block diagram of Figure A2, the following closed loop relation for IMC can be derived.

$$C = \frac{G_c^*G_c}{1 + G_c^*(G - M)}R + \frac{1 - G_c^*M}{1 + G_c^*(G - M)}d \quad (\text{A.2})$$

For the special case of perfect control (i.e.  $G = M$ ), we can have

$$C = G_c^*GR + [1 - G_c^*M]d \quad (\text{A.3})$$

The IMC controller is designed in two steps:

Step 1: The plant model is factored as

$$M = [M \quad \parallel M_+ ] \quad (\text{A.4})$$

where  $[M_+ ]$  contains any delays and right half plane zeros. It is specified so that the steady state gain is 1.

Step 2: The controller is specified as

$$G_c^* = \frac{1}{[M_-]} f \quad (\text{A.5})$$

where  $f$  is a low pass filter with a steady state gain of one. The IMC filter typically has the form

$$f = \frac{1}{(\Omega s + 1)^r} \quad (\text{A.6})$$

where  $\Omega$  is the desired closed loop time constant. Parameter  $r$  is a positive integer that is selected so that  $G_c^*$  is either a transfer function or if ideal derivative action is allowed,  $r$  can be chosen so that the order of the numerator exceeds the order of the denominator by one.

Note that the IMC controller in (A.5) includes the inverse of  $M_-$  rather than the inverse of the entire process model  $M$ . In contrast, if  $M$  had been used, the controller would contain a prediction term  $e^{ts}$ , (if  $M_-$  contained a time delay  $\tau$ ) or an unstable pole (if  $M_+$  contained a right half plane zero). Thus, by employing the factorization given in (A.4) and using a filter of the form of (A.6), the resulting controller  $G_c^*$  is guaranteed to be physically realizable and stable.

In general, the IMC approach does not necessarily result in a PID controller. However, Rivera et al. [1986] have shown that the IMC approach can be used to derive PID parameters for a wide variety of process models. Some of their results are shown in Table A1 where the PID controller itself is represented as

$$G_c = K_p \left( 1 + \frac{1}{T_i s} + T_d s \right) \quad (\text{A.7})$$

Table A1: IMC based PID controller parameters

Model	$K_p$	$T_i$	$T_d$
$\frac{K}{Ts+1}$	$\frac{T}{K\Omega}$	T	-----
$\frac{K}{(T_1s+1)(T_2s+1)}$	$\frac{T_1+T_2}{K\Omega}$	$T_1+T_2$	$\frac{T_1T_2}{T_1+T_2}$
$\frac{K(-ks+1)}{T^2s^2+2\zeta Ts+1}$	$\frac{2\zeta T}{K(k+\Omega)}$	$2\zeta T$	$\frac{T}{2\zeta}$
$\frac{K}{s(Ts+1)}$	$\frac{1}{K\Omega}$	-----	T

## *Appendix B: Derivation of SM controller*

Consider a system

$$\dot{x} = Ax + Bu + f(x) \quad (\text{B.1})$$

where  $A$  is the system matrix,  $B$  is the input matrix,  $u$  represents the input of the system and  $f(x)$  describes any deviations that would cause the system to deviate from its equilibrium point e.g. nonlinearities, unmodelled dynamics or external disturbances. Suppose a subsystem

$$\dot{x}_s = A_s x_s + B_s u_s + f_s(x_s) \quad (\text{B.2})$$

is selected in such a way that the dominant dynamics of the manoeuvre that is being controlled are being decoupled from the dynamics that have very little influence on the manoeuvre. Here  $x_s$  represents the subsystem states and  $u_s$  is the input to the subsystem.

The SM controller  $u_s$  has two components : (a) an equivalent control  $u_{eq}$  which provides the main control action and (b) the switching term  $u_{sw}$  that provides additional control action in order to compensate for any change in the nominal operating point the equivalent controller is designed around. That is,

$$u = u_{eq} + u_{sw} \quad (\text{B.3})$$

The equivalent control can be chosen as a state feedback gain controller of the following form:

$$u_{eq} = -k_s^T x_s \quad (\text{B.4})$$

where  $k_s$  is a feedback gain obtained from pole placement theory. Since this feedback control law is designed around a nominal linear plant it would not necessarily work well for all the operating conditions of the vessel. Therefore it requires additional control action in order to compensate for variation in the vessel's operating conditions. This additional control is provided by the non-linear switching term  $\sigma_s$  which is given by

$$\sigma_s(\hat{x}) = h_s^T \hat{x}_s = h_s^T (x_s - x_{sd}) \quad (\text{B.5})$$

where  $h_s$  is the right eigen vector of the desired closed loop matrix,  $x_{sd}$  is the desired state vector and  $\hat{x}$  is the state error.

Differentiating the above equation with respect to time yields

$$\dot{\sigma}_s(\hat{x}) = h_s^T \dot{\hat{x}} = h_s^T (\dot{x}_s - \dot{x}_{sd}) \quad (\text{B.6})$$

substituting equation (B.2) for  $\dot{x}_s$  in the above equation gives

$$\dot{\sigma}_s(\hat{x}) = h_s^T (A_s x_s + b_s u_s + f_s(x_s) - \dot{x}_{sd}) \quad (\text{B.7})$$

and substituting for  $u_s$  results in the following equation

$$\dot{\sigma}_s(\hat{x}) = h_s^T (A_s x_s + b_s u_{eq} + b_s u_{sw} + f_s(x_s) - \dot{x}_{sd}) \quad (\text{B.8})$$

Putting the value of  $u_{eq}$  from equation (B.4) yields

$$\begin{aligned} \dot{\sigma}_s(\hat{x}) &= h_s^T (A_s x_s - b_s k_s^T x_s + b_s u_{sw} + f_s(x_s) - \dot{x}_{sd}) \\ &= h_s^T (A_{cs} x_s + b_s u_{sw} + f_s(x_s) - \dot{x}_{sd}) \end{aligned}$$

or

$$\mathbf{u}_{sw} = (\mathbf{h}_s^T \mathbf{b}_s)^{-1} (\mathbf{h}_s^T \dot{\mathbf{x}}_{sd} - \mathbf{h}_s^T \mathbf{A}_{cs} \mathbf{x}_s - \mathbf{h}_s^T \mathbf{f}_s(\mathbf{x}_s) - \dot{\boldsymbol{\sigma}}_s(\hat{\mathbf{x}})) \quad (\text{B.9})$$

where  $\mathbf{A}_{cs} = \mathbf{A}_s - \mathbf{b}_s \mathbf{k}_s^T$  is the closed loop system matrix created by the feedback gain and  $(\mathbf{h}_s^T \mathbf{b}_s)^{-1}$  is assumed to be non zero. Since  $\mathbf{h}_s^T$  is chosen as the right eigen value of  $\mathbf{A}_{cs}$  it therefore corresponds to an eigen value of zero of this matrix. Hence it provides the following relationship

$$\mathbf{h}_s^T \mathbf{A}_{cs} = (\mathbf{A}_{cs}^T \mathbf{h}_s)^T = 0 \quad (\text{B.10})$$

Therefore (B.9) becomes

$$\mathbf{u}_{sw} = (\mathbf{h}_s^T \mathbf{b}_s)^{-1} (\mathbf{h}_s^T \dot{\mathbf{x}}_{sd} - \mathbf{h}_s^T \mathbf{f}_s(\mathbf{x}_s) - \dot{\boldsymbol{\sigma}}_s(\hat{\mathbf{x}})) \quad (\text{B.11})$$

Healey and Marco (1992) and Healey and Lienard (1993) define the  $\dot{\boldsymbol{\sigma}}$  as follows:

$$\dot{\boldsymbol{\sigma}}(\hat{\mathbf{x}}_s) = \mathbf{h}_s^T \Delta \mathbf{f}_s(\mathbf{x}_s) - \eta_s \text{sgn}(\boldsymbol{\sigma}(\hat{\mathbf{x}}_s)) \quad (\text{B.12})$$

where  $\eta_s$  is the switching gain which determines the amount of switching control action and

$$\Delta \mathbf{f}_s(\mathbf{x}_s) = \mathbf{f}_s(\mathbf{x}_s) - \hat{\mathbf{f}}_s(\mathbf{x}_s) \quad (\text{B.13})$$

Here  $\hat{\mathbf{f}}_s(\mathbf{x}_s)$  is the estimate of the function  $\mathbf{f}_s(\mathbf{x}_s)$ . Combining (B.4), (B.12) and (B.13) in (B.3) we get the total controller equation as

$$\mathbf{u}_s = -\mathbf{k}_s^T \mathbf{x}_s + (\mathbf{h}_s^T \mathbf{b}_s)^{-1} (\mathbf{h}_s^T \dot{\mathbf{x}}_{sd} - \mathbf{h}_s^T \hat{\mathbf{f}}_s(\mathbf{x}_s) - \eta_s \text{sgn}(\boldsymbol{\sigma}_s(\hat{\mathbf{x}}_s))) \quad (\text{B.14})$$

The estimate  $\hat{\mathbf{f}}_s(\mathbf{x}_s)$  is usually negligible and can be compensated for by making the switching gain sufficiently high. Therefore,

$$\mathbf{u}_s = -\mathbf{k}_s^T \mathbf{x}_s + (\mathbf{h}_s^T \mathbf{b}_s)^{-1} \left( \mathbf{h}_s^T \dot{\mathbf{x}}_{sd} - \eta_s \operatorname{sgn}(\sigma_s(\hat{\mathbf{x}}_s)) \right) \quad (\text{B.15})$$

To avoid the problem of chattering, the term  $\operatorname{sgn}(\sigma_s(\hat{\mathbf{x}}_s))$  may be replaced by  $\tanh\left(\frac{\sigma_s(\hat{\mathbf{x}}_s)}{\phi_s}\right)$ . Here  $\phi_s$  is called the boundary layer thickness and defines the range about the zero sliding surface where the switching term transition is smoothed. This term acts like a low pass filter and is the reason that this is called *soft switching*. Now the controller equation becomes

$$\mathbf{u}_s = -\mathbf{k}_s^T \mathbf{x}_s + (\mathbf{h}_s^T \mathbf{b}_s)^{-1} \left( \mathbf{h}_s^T \dot{\mathbf{x}}_{sd} - \eta_s \tanh\left(\frac{\sigma_s(\hat{\mathbf{x}}_s)}{\phi_s}\right) \right) \quad (\text{B.16})$$

which is equation (3.30) of Section 3.7.2

## *Appendix C: Levenberg-Marquardt Algorithm*

The Levenberg-Marquardt (L-M) algorithm is very popular in the area of numerical optimization and has been used successfully in solving non-linear least squares problems.

The L-M algorithm is an approximation to Newton's method. Suppose that we have a function  $f(\mathbf{x})$  which is to be minimized with respect to the parameter vector  $\mathbf{x}$ , then the Newton's method would be

$$\Delta \mathbf{x} = -[\nabla^2 f(\mathbf{x})]^{-1} \nabla f(\mathbf{x}) \quad (\text{C.1})$$

where  $\nabla^2 f(\mathbf{x})$  is the Hessian matrix and  $\nabla f(\mathbf{x})$  is the gradient. If we assume that  $f(\mathbf{x})$  is a sum of squares function

$$f(\mathbf{x}) = \sum_{i=1}^N e_i^2(\mathbf{x}) \quad (\text{C.2})$$

then it can be shown that

$$\nabla f(\mathbf{x}) = \mathbf{J}^T(\mathbf{x}) \mathbf{e}(\mathbf{x}) \quad (\text{C.3})$$

$$\nabla^2 f(\mathbf{x}) = \mathbf{J}^T(\mathbf{x}) \mathbf{J}(\mathbf{x}) + \mathbf{S}(\mathbf{x}) \quad (\text{C.4})$$

where  $\mathbf{J}(\mathbf{x})$  is the Jacobian matrix

$$J(\mathbf{x}) = \begin{bmatrix} \frac{\partial e_1(\mathbf{x})}{\partial x_1} & \frac{\partial e_1(\mathbf{x})}{\partial x_2} & \dots & \frac{\partial e_1(\mathbf{x})}{\partial x_n} \\ \frac{\partial e_2(\mathbf{x})}{\partial x_1} & \frac{\partial e_2(\mathbf{x})}{\partial x_2} & \dots & \frac{\partial e_2(\mathbf{x})}{\partial x_n} \\ \vdots & \vdots & & \vdots \\ \frac{\partial e_N(\mathbf{x})}{\partial x_1} & \frac{\partial e_N(\mathbf{x})}{\partial x_2} & \dots & \frac{\partial e_N(\mathbf{x})}{\partial x_n} \end{bmatrix} \quad (\text{C.5})$$

and

$$S(\mathbf{x}) = \sum_{i=1}^N e_i(\mathbf{x}) \nabla^2 e_i(\mathbf{x}) \quad (\text{C.6})$$

For the Gauss-Newton method it is assumed that  $S(\mathbf{x}) \approx 0$ , and the update (C.1) becomes

$$\Delta \mathbf{x} = \left[ J^T(\mathbf{x}) J(\mathbf{x}) \right]^{-1} J^T(\mathbf{x}) \mathbf{e}(\mathbf{x}) \quad (\text{C.7})$$

The L-M modification to the Gauss-Newton method is

$$\Delta \mathbf{x} = \left[ J^T(\mathbf{x}) J(\mathbf{x}) + \mu I \right]^{-1} J^T(\mathbf{x}) \mathbf{e}(\mathbf{x}) \quad (\text{C.8})$$

where  $I$  is the identity matrix and  $\mu$  is a scalar. This equation is similar to equation (4.30) of Chapter 4.



## Appendix D: Supply Ship Model

The dynamics of CyberShip1 of Chapter 7 can be represented by the following kinetic equation:

$$M\dot{v} + C(v)v + Dv = \tau_t \quad (D.1)$$

where  $v = [u \ v \ r]^T$ ,  $\tau_t = [\tau_{t1} \ \tau_{t2} \ \tau_{t3}]^T$ ,

$$M = \begin{bmatrix} m_{11} & 0 & 0 \\ 0 & m_{22} & m_{23} \\ 0 & m_{32} & m_{33} \end{bmatrix} = \begin{bmatrix} m - X_u & 0 & 0 \\ 0 & m - Y_v & mx_G - Y_r \\ 0 & mx_G - N_v & I_z - N_r \end{bmatrix} = \begin{bmatrix} 19.0 & 0 & 0 \\ 0 & 35.2 & 0 \\ 0 & 0 & 2.0 \end{bmatrix}$$

$$C(v) = \begin{bmatrix} 0 & 0 & -m_{22}v - m_{23}r \\ 0 & 0 & m_{11}u \\ m_{22}v + m_{23}r & -m_{11}u & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & -35.2v \\ 0 & 0 & 19.0u \\ 35.2v & -19.0u & 0 \end{bmatrix}$$

$$D = \begin{bmatrix} -X_u & 0 & 0 \\ 0 & -Y_v & 0 \\ 0 & 0 & -N_r \end{bmatrix} = \begin{bmatrix} m_{11}/T_1 & 0 & 0 \\ 0 & m_{22}/T_2 & 0 \\ 0 & 0 & m_{33}/T_3 \end{bmatrix} = \begin{bmatrix} 6.3 & 0 & 0 \\ 0 & 7.0 & 0 \\ 0 & 0 & 2.0 \end{bmatrix}$$

The values of  $T_1$ ,  $T_2$  and  $T_3$  are 3sec, 5sec and 1 sec respectively.

The kinematics equation is

$$\dot{\eta} = J(\eta)v \quad (D.2)$$

where  $v = [u \ v \ r]^T$ ,  $\eta = [x \ y \ \Psi]^T$  and

$$J(\eta) = \begin{bmatrix} \cos \Psi & -\sin \Psi & 0 \\ \sin \Psi & \cos \Psi & 0 \\ 0 & 0 & r \end{bmatrix}$$