1999

Intelligent Control Strategies for an Autonomous Underwater Vehicle

Craven, Paul Jason

UNIVERSITY OF PLYMOUTH

Intelligent Control Strategies for an Autonomous Underwater Vehicle

Paul Jason Craven

A thesis submitted for the degree of
Doctor of Philosophy

in the
Faculty of Technology
Department of Mechanical and Marine Engineering

in collaboration with the Sea Systems and Platform Integration Sector,
Defence Evaluation and Research Agency, Winfrith.

January 1999

UNIVERSITY OF PLYMOUTH
Faculty of Technology
Department of Mechanical and Marine Engineering

Intelligent Control Strategies for an Autonomous Underwater Vehicle

Paul Jason Craven

Abstract

The dynamic characteristics of autonomous underwater vehicles (AUVs) present a control problem that classical methods cannot often accommodate easily. Fundamentally, AUV dynamics are highly non-linear, and the relative similarity between the linear and angular velocities about each degree of freedom means that control schemes employed within other flight vehicles are not always applicable. In such instances, intelligent control strategies offer a more sophisticated approach to the design of the control algorithm. Neurofuzzy control is one such technique, which fuses the beneficial properties of neural networks and fuzzy logic in a hybrid control architecture. Such an approach is highly suited to development of an autopilot for an AUV.

Specifically, the adaptive network-based fuzzy inference system (ANFIS) is discussed in Chapter 4 as an effective new approach for neurally tuning course-changing fuzzy autopilots. However, the limitation of this technique is that it cannot be used for developing multivariable fuzzy structures. Consequently, the co-active ANFIS (CANFIS) architecture is developed and employed as a novel multivariable AUV autopilot within Chapter 5, whereby simultaneous control of the AUV yaw and roll channels is achieved. Moreover, this structure is flexible in that it is extended in Chapter 6 to perform on-line control of the AUV leading to a novel autopilot design that can accommodate changing vehicle payloads and environmental disturbances.

Whilst the typical ANFIS and CANFIS structures prove effective for AUV control system design, the well known properties of radial basis function networks (RBFN) offer a more flexible controller architecture. Chapter 7 presents a new approach to fuzzy modelling and employs both ANFIS and CANFIS structures with non-linear consequent functions of composite Gaussian form. This merger of CANFIS and a RBFN lends itself naturally to tuning with an extended form of the hybrid learning rule, and provides a very effective approach to intelligent controller development.

Acknowledgements

The work detailed within this thesis would not have been possible without the encouragement of my supervisory team. I would sincerely like to thank Dr Robert Sutton (Director of Studies) for his technical input, advice and constant good humour which made my Ph.D program a smooth and enjoyable one. I would also like to express my gratitude towards Professor Roland S. Burns for providing continual encouragement and advice, and Dr Yong-Ming Dai for his expertise in mathematical modelling with MATLAB/Simulink. They are both acknowledged for their careful reading of this thesis.

Thanks are extended to Simon Corfield and Don Cowling of the Defence Evaluation and Research Agency, Winfrith; the provision of their underwater vehicle model created the impetus for this work, and their technical support has proved very useful. The financial support of the Engineering and Physical Sciences Research Council is also acknowledged.

I would also like to express my gratitude towards Dr Miroslaw Kwiesielewicz and Professor Antonio Tiano for their continuing help, advice and friendship.

In particular I thank my family for their continuing support. Without them this thesis would not have been possible.

Finally, the unfaltering support of my closest friend Lucy to whom this thesis is dedicated is greatly appreciated; a special thank you goes out to her.

Declaration

- The work presented within this thesis is original; no part of this work has been used for any other award or degree at any time.

- During the candidature, the author has not been registered for any other award at any other institution.

Paul Jason Craven
15th January 1999.

Contents

List of Figures

List of Tables

Nomenclature

UUV Dynamics

$[u, v, w, p, q, r]$	-	*linear and angular velocities*
$[x, y, z, \phi, 6, \psi]$	-	*position and euler angles*
X, Y, Z	-	*hydrodynamic forces*
K, M, N	-	*hydrodynamic moments of forces*
X_0, Y_0, Z_0	-	*earth-fixed frame of reference*
m	-	*vehicle mass in air*
W	-	*vehicle weight in air*
B	-	*vehicle buoyancy in air*
l	-	*vehicle length*
g	-	*force due to gravity*
x_G, y_G, z_G	-	*position of the vehicle centre of gravity*
x_B, y_B, z_B	-	*position of the vehicle centre of buoyancy*
ρ	-	*density of water*
I_x, I_y, I_z	-	*moments of inertia about body-fixed coordinate system*
$G_X, ..., G_N$	-	*hydrostatic forces and moments*
$X_{wave}, ..., N_{wave}$	-	*wave forces and moments*
$\delta_{bp}, ..., \delta_{sp}$	-	*forces arising from control surfaces*

$T_p,....,T_s$	-	*forces arising from thrusters*
$X_{uu}^{\cdot},....,N_{r\|r\|}^{\cdot}$	-	*velocity dependent hydrodynamic coefficients*
$X_{current}, Y_{current}, Z_{current}$	-	*current forces along earth-fixed axes*
$X_{velocity}, Y_{velocity}, Z_{velocity}$	-	*constant current component velocities of the total current*
ψ_ε	-	*yaw integral of time squared error*
δ_ψ	-	*canard integral of time squared error*
ϕ_ε	-	*roll integral of time squared error*
δ_ϕ	-	*stern hydroplane integral of time squared error*
T_R	-	*course-change rise time*
$M_p(t)$	-	*peak course overshoot*
sse	-	*steady-state course error*

Fuzzy Inference Systems

$A_i, B_i,...., P_i$	-	*premise membership functions*
f_i	-	*the i_{th} fuzzy rule output (consequent function)*
$p_i, q_i,....., v_i$	-	*linear coefficients within the i_{th} fuzzy consequent function*
w_i	-	*weight of the i_{th} fuzzy rule*
$\overline{w}_i$	-	*normalized weight of the i_{th} fuzzy rule*
u_0	-	*the crisp control output*
x_i, y_i	-	*inputs and outputs to a FIS*
Π	-	*product or T-norm operator*
N	-	*ratio of the i_{th} rule to all other rules*
Σ	-	*summation of incoming signals*
$\mu_{\alpha,i}$	-	*the i_{th} premise membership function*
S	-	*the total parameter set of the FIS*
O_i^k	-	*the output of the i_{th} node within layer k*
$\#(k)$	-	*the number of nodes within layer k*
η	-	*learning rate during gradient transition*
κ	-	*step size during gradient transition*
N, Z, P	-	*negative, zero and positive fuzzy sets*
$T_{m,p}$	-	*the m_{th} component of the p_{th} target output vector*
$O_{m,p}^L$	-	*the m_{th} component of the p_{th} output layer node*

∇_{x^*}	-	*gradient with respect to* x^*
$\|*\|$	-	*L2-norm of vector denoted by* *

Artificial Neural Networks

x_i, y_i	-	*inputs and outputs*
w_{ji}	-	*weighted connection between* i_{th} *and* j_{th} *layers*
Δw_{ji}	-	*change in weighted connection between* i_{th} *and* j_{th} *layers*
E	-	*error measure at the output of the ANN*
t_k	-	*training pattern k*
y_k	-	*output k of the ANN*
o_k	-	*output of the* k_{th} *layer neuron*

Gaussian Inference Systems

η_k	-	*linear weighting coefficient for the* k_{th} *fuzzy rule*
ξ_k	-	k_{th} *non-linear radial function of the input variables*
f_j	-	*summation of the radial consequent functions*
σ_k	-	*width of the* k_{th} *radial basis function*
c_k	-	*centre of the* k_{th} *radial basis function*

On-Line Control

t	-	*discrete sample time*
k	-	*sample number*
h	-	*sampling interval width*
$x(t_0 + k \times h)$	-	*actual state of the plant at* $t = t_0 + k \times h$
$x_d(t_0 + k \times h)$	-	*desired state of the plant at* $t = t_0 + k \times h$

Acronymns

ANFIS	-	*Adaptive Network-based Fuzzy Inference System*
ANN	-	*Artificial Neural Network*
AUV	-	*Autonomous Underwater Vehicle*

BP	-	*Back-Propagation*
CANFIS	-	*Co-active ANFIS*
FIS	-	*Fuzzy Inference System*
FLC	-	*Fuzzy Logic Controller*
ITSE	-	*Integral Square Error over Time*
LOS	-	*Line Of Sight*
MATLAB	-	MAT*rix* LAB*oratory*
MIMO	-	*Multi Input – Multi Output*
MISO	-	*Multi Input – Single Output*
NGC	-	*Navigation, Guidance and Control*
PID	-	*Proportional plus Integral plus Derivative*
RBF	-	*Radial Basis Function*
RBFN	-	*Radial Basis Function Network*
ROV	-	*Remotely Operated Vehicle*
SANN	-	*Stage Adaptive Neural Network*
SISO	-	*Single Input – Single Output*
SNR	-	*Signal to Noise Ratio*
TSK	-	*Takagi-Sugeno-Kang*
UUV	-	*Unmanned Underwater Vehicle*

Chapter 1

Introduction

1.1 Motivation

Although Bourne can be credited with producing the first conceptual design for a submarine in 1578, the first one built was constructed in 1620 by Van Drebbel. Nevertheless, it was not until 1776 that a submarine was specifically launched to take part in naval operations. Bushnell's submarine the *Turtle* was designed to destroy the Royal Navy men-of-war, which were participating in naval blockades during the American War of Independence. Fortunately for the British fleet, the attacks by the human powered *Turtle* (Figure 1.1) were unsuccessful. The *Turtle's* single crew member blamed the ineffectiveness of the assaults on the inability to lay 150 pound charges against the hulls of the ships owing to their reputed copper sheathing. In actual fact, the British warships were not sheathed. A more probable explanation has been postulated by Coverdale and Cassidy (1987) who propose it was due to the crew member

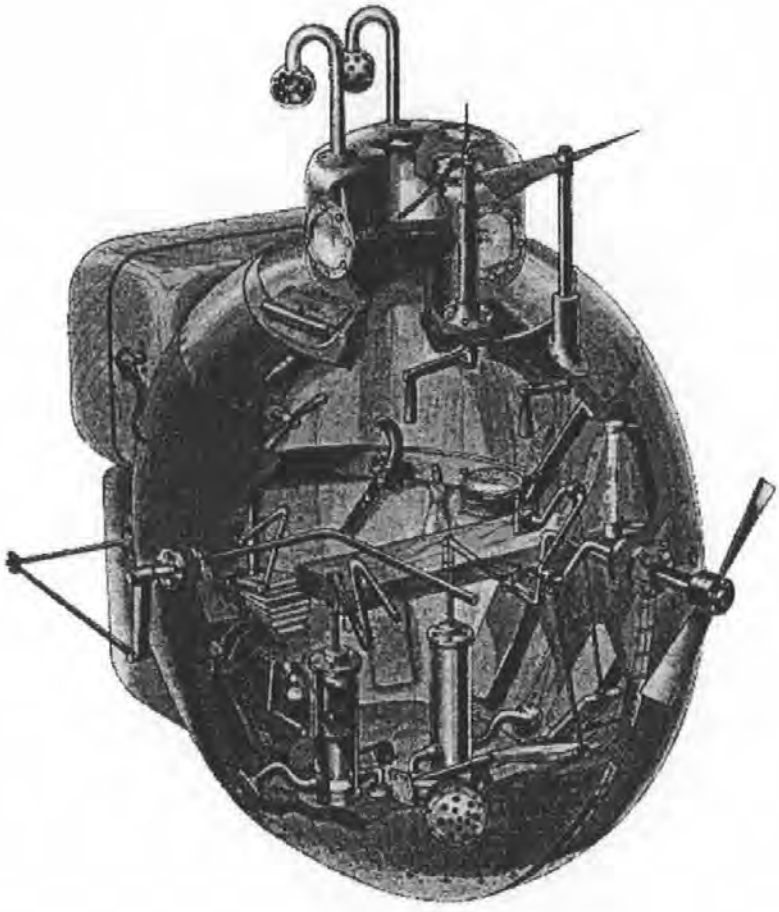


Figure 1.1: The *'Turtle'* of Bushnell (after Batchelor et *al.* (1979)).

being physically exhausted and affected by the build up of unacceptable carbon dioxide levels in the vessel by the time it reached an intended target. Reader et *al.* (1989) light-heartedly suggested that this may have been the initial impetus for the search for unmanned underwater vehicles (UUVs)! Clearly since those pioneering days, manned submarine technology has advanced dramatically. However, the common potential weakness throughout their evolution has been the reliance on humans to perform operational tasks.

Some of the earliest developments in UUV technology can be attributed to the cable-controlled underwater recovery vehicle design and construction programme instigated by the US Navy in 1958. In 1963 one of these craft was used in the search for the ill-fated USS *Thresher* which tragically sank off the New England coast in 1400 fathoms of water. Later another was used to help recover the US Navy hydrogen bomb lost off the coast of Palomares, Spain, in 1966. Notwithstanding those successes and the accompanying publicity, the commercial potential of UUVs was not recognized until the discovery of offshore oil and gas in the North Sea. More specifically, remotely operated vehicles (ROVs) began and continue to be used extensively throughout the offshore industry. Whereas, both in the naval and commercial sectors, autonomous underwater vehicle (AUV) usage was limited. Even so, more recently, interest in the possible use of both types of vehicle has been heightened. This has been prompted by the needs of the offshore industry to operate and explore in extreme depths in a continuously hostile environment and the requirements of navies to have low cost vehicles capable of undertaking covert surveillance missions and performing mine laying and disposal operations. This revival is also coupled with the current and ongoing advances being made in control techniques.

More specifically, the dynamic characteristics of UUVs present a control system design problem which classical linear design methodologies cannot accommodate easily. Fundamentally, UUV dynamics are non-linear in nature and are subject to a variety of disturbances such as vorticity effects and currents. Therefore they offer a challenging task in the development of suitable algorithms for motion and position control in the six degrees of freedom in which they operate, and are required to be robust in terms of disturbance rejection and varying vehicle speeds and dynamics. It should be noted that the term "unmanned underwater vehicle" as used here is a generic expression to describe both an AUV and an ROV. An AUV is regarded as a marine craft which fulfils a mission or task without being constantly monitored and supervised by a human operator, whilst an ROV is a marine vessel that requires instructions from an operator via a tethered cable or an acoustic link.

1.2 Research Objectives

The objectives of the programme of research were to:

(a) Critically review the current UUV control literature

(b) Define non-linear models pertaining to the yaw and roll degrees of freedom

(c) Develop traditional single-input single-output (SISO) control algorithms for the yaw and roll degrees of freedom

(d) Investigate various neuro-fuzzy algorithms and structures for yaw and roll control

(e) Produce candidate neuro-fuzzy control algorithms for each degree of freedom

(f) Employ suitable neuro-fuzzy algorithms within a multi-input multi-output (MIMO) configuration to control yaw and roll simultaneously – this configuration should be flexible to allow full six degree of freedom control

(g) Critically assess the performance of the chosen multivariable controller configuration and provide flexible alternatives to the underlying algorithm

1.3 Thesis Overview

Intelligent control and intelligent systems have recently received a great deal of interest within the control community. However, it is still not apparent as to what these phrases mean in the context of control theory. Many definitions of intelligence are given throughout the literature, yet no one definition provides a complete picture of the factors which influence a systems' intelligence.

Essentially, all definitions have certain key factors in common. An intelligent system or controller must possess the ability to learn a process, adapt its behaviour in light of process changes, store and recall relevant information and autonomously improve its performance when required to do so. Indeed a control system which can retrieve information about a situation that has previously occurred is highly desirable, yet in addition it is important that such a system is able to adapt to register new information as well. The previous sentence summarizes the distinction between a learning system and an adaptive system respectively. The aim of any intelligent approach to control or system design is to incorporate aspects from each area to enhance the overall performance when faced with non-linear dynamic process changes and/or the requirement to interpolate between known situations. Subsequently, the design technique should be transportable to reduce the cost of re-employing the algorithms within novel plant architectures.

Chapter 2 of this thesis provides the interested reader with a critical review of artificial intelligence approaches to control system design for unmanned underwater vehicles. This review is sub-divided into artificially intelligent techniques, and modern and classical approaches to UUV control. The review of classical and modern control techniques as applied to UUVs is included in Appendix B for brevity. The attractiveness of neuro-fuzzy approaches to control system design is discussed, clarifying the motivation behind the work submitted within this thesis.

The dynamic characteristics of the underwater vehicle are detailed in Chapter 3, along with results of preliminary open loop experiments for the yaw and roll channels. Various performance specifications used within the ensuing work are outlined, including the design of suitable benchmark autopilots and a line of sight algorithm for autonomous vehicle guidance. The chapter concludes by suggesting two autopilots from a possible five, to be used on a comparative basis with the results of the following chapter.

The neuro-fuzzy concepts developed within this work are detailed in Chapter 4. Specifically, the adaptive network-based fuzzy inference system (ANFIS) of Jang (1993) is explained fully as the architecture for the application of various parameter tuning algorithms. Initially, the backpropagation, chemotaxis and simulated annealing algorithms are employed to tune the input fuzzy sets of a 9 rule fuzzy course-changing autopilot designed in Chapter 3. The results pertaining to these tuning regimes lead to a clear strategy for autopilot tuning. Subsequently, the full parameter set of the 9 rule fuzzy autopilot of Chapter 3 is considered for adaptation, using the hybrid learning rule, and the chemotaxis and simulated annealing algorithms. Comprehensive results concerning autopilot robustness, generalization and autonomous guidance are included, leading to a definitive strategy for multi input – single output (MISO) autopilot tuning.

The inability of the ANFIS technique to control multiple input – multi output (MIMO) plant is discussed within Chapter 5. A novel approach to multivariable control is presented which accounts for the cross-coupling between vehicle degrees of freedom. This autopilot design approach is considered as a natural extension to the ANFIS technique. Results are presented which compare the developed tuning architecture to the ANFIS approach and a traditional non-interacting control system design approach. The effectiveness of this new design technique is illustrated through numerous AUV simulations examining autopilot robustness, generalization and autonomous guidance. Notwithstanding, the inability of the developed approach to adapt to incoming data signals prompts further development of the neuro-fuzzy algorithm used for tuning.

Consequently, Chapter 6 discusses a natural progression of the autopilot designs of Chapters 4 and 5 towards on-line control. Essentially, the resulting autopilots employ a sequential least squares algorithm fused with gradient descent to adapt their parameter sets to represent incoming data. The respective autopilot structures are encoded as stage adaptive networks within the modelling framework. The results demonstrate the feasibility of the new autopilot schema, but detail the inadequacies of the sequential least-squares algorithm for AUV control. Thus an improved control algorithm, employing a switching facility during transient periods of motion, is developed as a novel control scheme.

The results of Chapter 7 document original work carried out within the context of neuro-fuzzy AUV control, and indeed control theory. The traditional method of radial basis function approximations is combined with the ANFIS modelling approach to create a new control architecture. By replacing the typical linear rule consequents within the ANFIS regime with composite non-linear rules of Gaussian form a non-linear fuzzy inference system (FIS) is produced. The resulting non-linear gain-scheduling controller proves effective, particularly for multivariable control of the AUV yaw and roll channels. However, the new consequents are more computationally expensive to implement than the original linear rules of the ANFIS regime.

Consequently, section 7.5.2 is devoted to computationally efficient implementation of these consequent functions. The inherent multiplication of each scaling coefficient within the consequent functions is replaced with a series of additions to produce a smaller computational burden within the network structure. The result is an autopilot that produces an almost identical AUV response but with a lower computational cost.

Concluding remarks and a summary of the thesis' objectives are presented in Chapter 8 for completeness.

1.4 Contributions of the Thesis

The major contributions of this work are seen as:

- The adaptive network-based fuzzy inference system (ANFIS) technique has been applied directly to the problem of autopilot design for an AUV.

- A novel multivariable control scheme based on the co-active ANFIS (CANFIS) regime of Mizutani and Jang (1995) has been developed. Consequently, a MATLAB dependent C library has been produced for application to a wider range of modelling problems.

- On-line control of an AUV has been investigated via the use of a neuro-fuzzy control algorithm based on the above work.

- A more sophisticated modelling approach has been introduced which employs Gaussian consequent functions as opposed to the linear rule outputs of the ANFIS technique. Results gained through use of this approach are superior to those achieved previously.

References

Batchelor, J., Preston, A. and Casey, L. S. (1979). Sea Power. Exeter Books, New York.

Coverdale, A. and Cassidy, S. (1987). The Use of Scrubbers in Submarines. Journal of Naval Engineering, pp528-544.

Jang, J.-S. R. (1992). Neuro-Fuzzy Modelling: Architecture, Analyses and Applications, Ph.D Thesis, Department of Electrical Engineering, University of California, Berkeley, CA 94720, United States of America.

Mizutani, E. and Jang, J.-S. R. (1995). Co-Active Neuro-Fuzzy Modelling. Proceedings of the International Conference on Neural Networks, pp760-765, Perth, Western Australia, Australia.

Reader, G.T., Walker, G. and Hawley, J.G. (1989). Non-Nuclear Powerplants for AUVs. Proceedings of the 6th International Symposium on Untethered Submersible Technology, University of New Hampshire, pp88-99.

Chapter 2

Artificially Intelligent Control Strategies for Unmanned Underwater Vehicles

2.1 Introduction

The purpose of this chapter is to review a number of artificially intelligent approaches that have been adopted to control the dynamic behaviour of UUVs. For the interested reader Appendix B contains a review of classical and modern techniques for UUV control system design. Where relevant, additional references will be reviewed within the appropriate chapters. Craven et *al.* (1998) details many classical, modern and artificially intelligent aspects of UUV control.

Classical linear control system design methods are adequate to control linear systems. However, such approaches are often notably lacking in robustness when the system to

be controlled exhibits characteristics of non-linearity, time dependence and high complexity. Notwithstanding, human operators still manage to control dynamical systems displaying such characteristics. The emergence of fuzzy logic has enabled the vagueness of human language to be mathematically quantified. Consequently, the control decisions of an experienced plant operator could be formulated into an algorithm to control the desired plant. Such an approach may therefore be capable of controlling an UUV very successfully.

2.2 Fuzzy Logic Control Schemes

Basically, a fuzzy inference system (FIS) consists of five key elements as shown in Figure 2.1. The *fuzzification unit* receives crisp input signals and transforms them into fuzzy values based upon their degree of match with the pre-defined fuzzy input membership functions (sets) of the *data base*. Similarly, the *defuzzification unit* converts the values of the consequent terms into crisp output signals, again using the information contained within the *database*. The *decision-making element* provides a logical inference of the linguistic rules contained within the *rule base*. This *rule base* typically consists of a number of linguistic rules. Within the context of an UUV autopilot and its internal structure, these rules could take the form:

If *yaw error* is *positive small* and *yaw rate* is *positive big*
then *rudder demand* is *zero*

where the terms "*positive small*", and "*positive big*" are fuzzy sets defined within the input space. The term "*zero*" in the output space can also be represented by a fuzzy set, yet in control applications is often written as a linear combination of the input variables:

If *yaw error* is *positive small* and *yaw rate* is *positive big*
then *rudder demand* is $p_\alpha \psi_\varepsilon + q_\alpha \dot{\psi} + r_\alpha$.

Depending upon the choice of consequent (output) set, various defuzzification strategies are commonly used. Figures 2.2 and 2.3 depict FISs based upon Mamdani and Takagi-Sugeno-Kang (TSK) (Takagi and Sugeno, (1985)) inference styles respectively. Other methods of inference can be employed such as Tsukamoto style fuzzy inference. However, the methods shown herein represent the more popular styles of inference for control applications.

In order to elicit a deterministic value from the resultant fuzzy control output set, the centre of area method (Eqn(2.1)) is often employed in Mamdani FISs:

$$u_o = \frac{\sum_{i=1}^{N} u_i U(u_i)}{\sum_{i=1}^{N} U(u_i)} \tag{2.1}$$

where u_o represents the crisp output of the FIS due to the N fuzzy control rules, and u_i is the consequent fuzzy subset in question. Alternatively, due to the crisp nature of TSK consequent sets, the weighted average defuzzifier is typically employed:

$$u_o = \frac{\sum_{i=1}^{N} w_i f_i}{\sum_{i=1}^{N} w_i} \tag{2.2}$$

where w_i is the weight of the i_{th} fuzzy rule and f_i is the i_{th} linear function of the inputs.

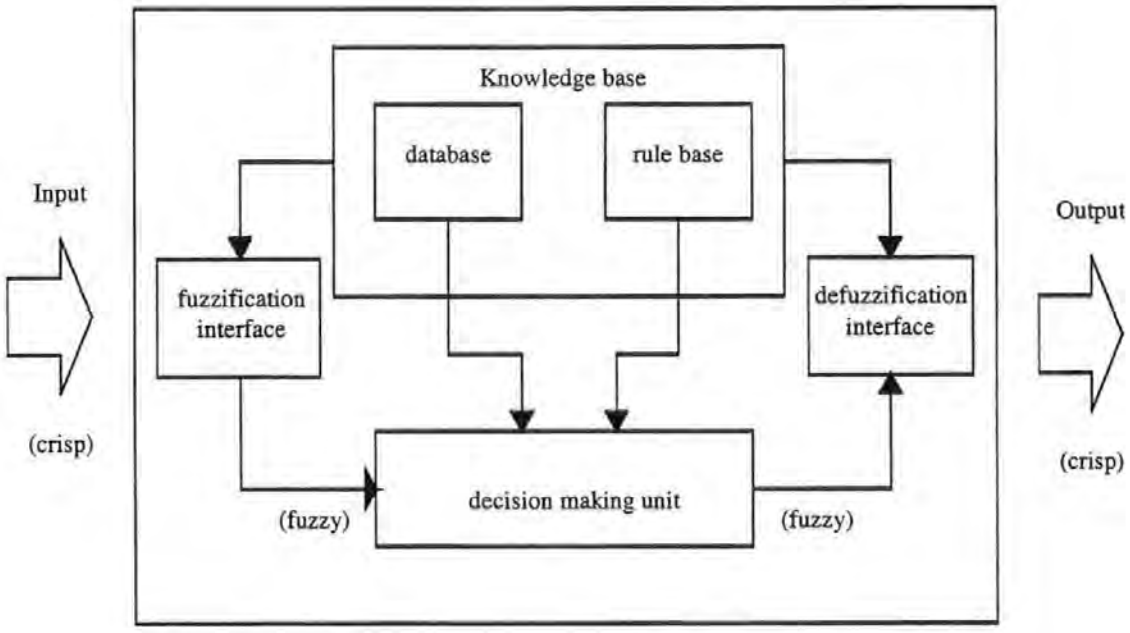


Figure 2.1: Fuzzy inference system structure (after Jang(1993)).

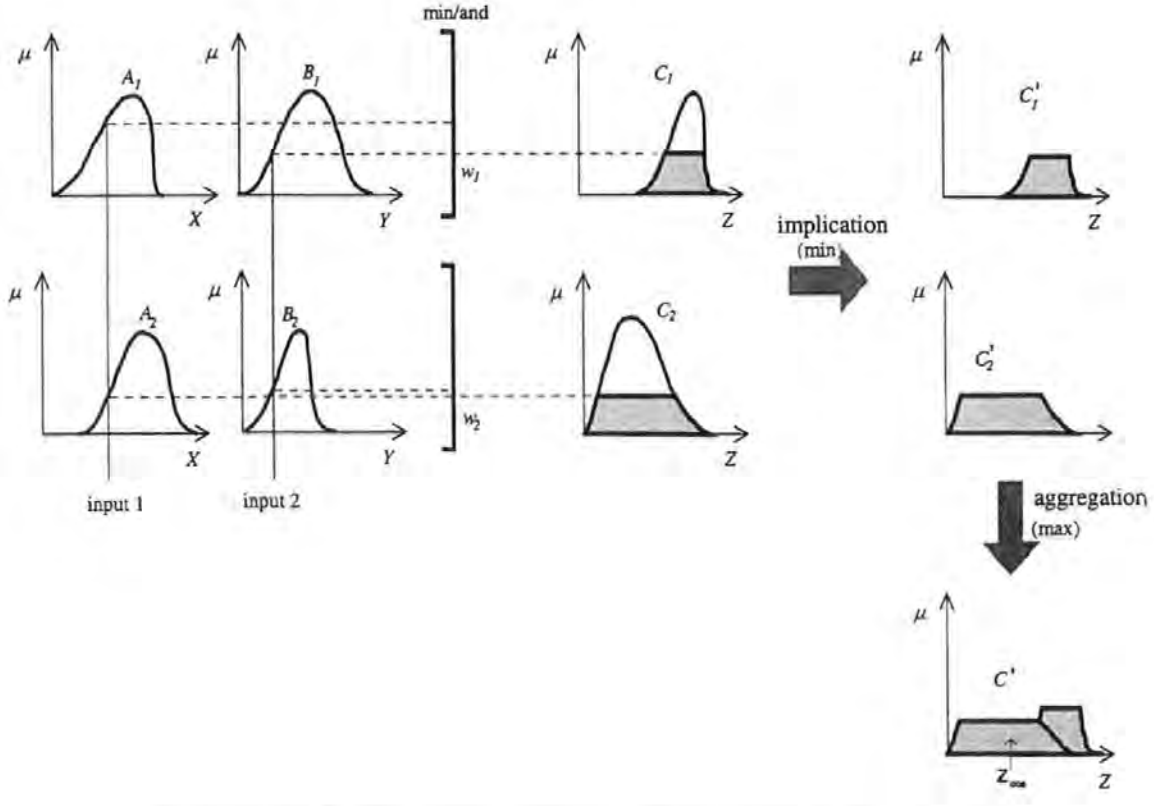


Figure 2.2: Mamdani fuzzy inference diagram (after Jang(1993)).

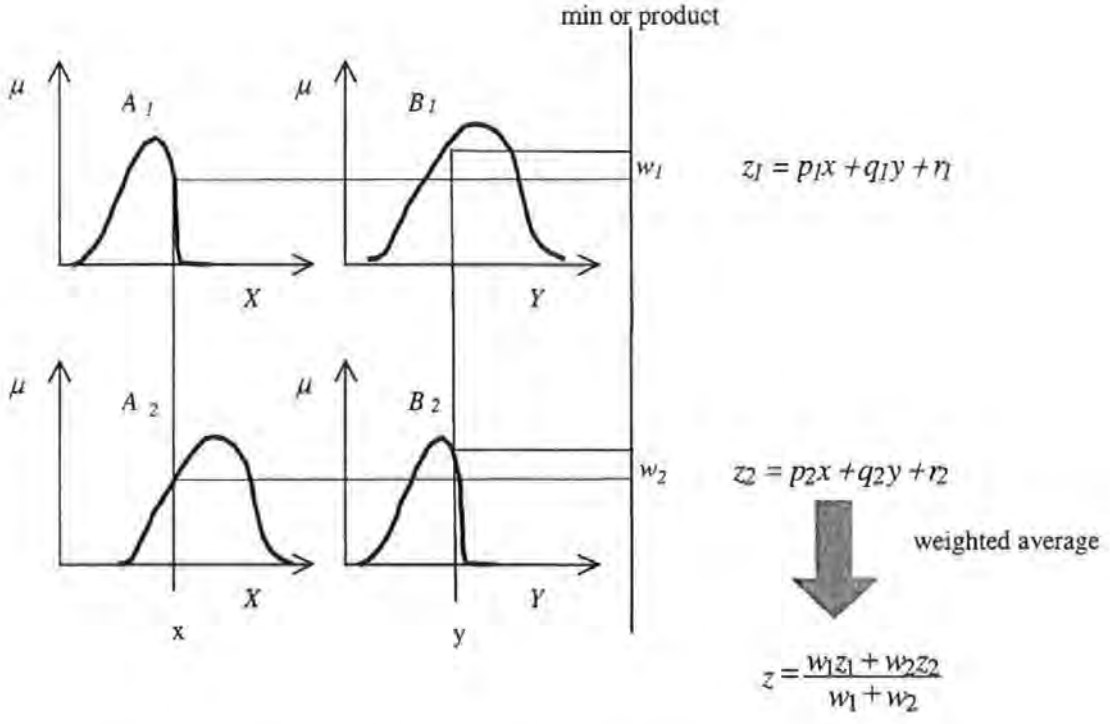


Figure 2.3: Takagi-Sugeno-Kang fuzzy inference diagram (after Jang (1993)).

Control systems that are based on fuzzy logic are often robust to parametric uncertainties and can deal with the inherent non-linearities of the plant under consideration. They therefore offer the potential to control UUVs in an effective manner.

2.2.1 Fixed Rule-Based Fuzzy Control

One particular implementation of interest here is the study conducted by de Bitetto (1995), who applied fuzzy logic to the depth and pitch control of an UUV. A fixed rule base is used containing fourteen rules for depth, pitch and ballast control. Good overall performance is achieved, although simulation results are obtained at a forward speed that is considered too slow for cross-coupling effects between the yaw and pitch channels to be influential. However, due to the use of fuzzy rules, allowing an insight into the control strategy, the control rule base could be easily modified.

Smith et *al.* (1993) applied fixed rule based TSK style fuzzy controllers simultaneously to the channels yaw, heave and pitch of the *Ocean Voyager* AUV. Two problems were addressed: (i) the low level control problem of developing a control system which could reliably and efficiently manoeuvre the AUV, and (ii) the high level problem of docking the AUV in a confined space. Promising results were achieved using this approach although a more detailed analysis and comparison of the fuzzy controller to more conventional methods was not forthcoming.

Although fixed rule base control strategies have received a great deal of attention in past years, adaptive fuzzy control schemes are more common in the UUV control literature. The following section details various self-organizing fuzzy controller studies.

2.2.2 Fuzzy Self-Organizing Control

In some circumstances it may be difficult to obtain a clear set of fuzzy rules which describe the controller action required, particularly in the case of non-linear, time-varying processes. To overcome this problem, fuzzy self-organizing controllers (FSOC) have been developed [Shao (1988), Daley and Gill (1986), Mandic et *al.* (1985), Tanscheit and Scharf (1988), Procyk and Mamdani (1979), and Farbrother (1991)] which generate their own fuzzy rule-base by continual performance feedback, thus assessing the rule bases' effectiveness. A schematic of a self-organizing fuzzy controller is shown in Figure 2.4.

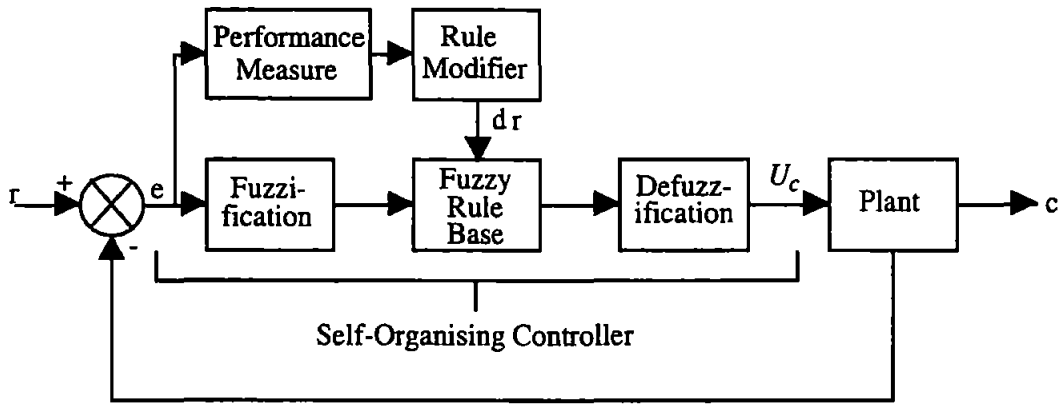


Figure 2.4: The fuzzy self-organizing controller.

Essentially the architecture of a FSOC is similar to that of a fixed rule based fuzzy controller but has the added refinement of a higher hierarchical level superimposed upon it. The hierarchical structure consists of three elements, namely, an appropriate performance index (PI), a simplified model of the plant and a rule modification algorithm. An excellent explanation of FSOC is detailed in Harris et *al.* (1993) including a review of FSOC applied to ship yaw control.

The learning mechanism in this control system uses the values of error and change in error to initiate any improvements to the rules driven by the PI. Assuming the plant is in an undesired state, the PI relates the measured values to a rule correction (dr), which is then related to the rule modification algorithm via the simplified plant model. Essentially, dr represents the magnitude of the proposed rule modification within the PI. For a single-output system, the simplified plant model can be a sign change of unity, whereas for a multivariable system, a matrix of its steady-state gains will suffice.

Clearly the above is a simplified description of a FSOC. A detailed exposition of a FSOC adapted to form the basis of a ship autopilot can be found in Sutton and Jess (1991). The fuzzy algorithm within the autopilot operates as in the case of a fixed rule base fuzzy controller with the exception that the compositional rule of inference is interpreted differently. The composition rule of inference is written as:

$$U = (E \times CE) \circ R \tag{2.3}$$

where U is the control signal, E is the error, CE is change in error, R represents the rulebase and "o" normally denotes the max-min product. However, Yamazaki (1982) has found that better control responses result from using the max-max product. Thus, for this autopilot Eqn.(2.3) is rewritten as:

$$U(u_k) = \bigvee_{\substack{e \in E \\ ce \in CE}} \left[E(e_i) \wedge CE(ce_i) \vee R(e_\ell, ce_j, u_k) \right] \tag{2.4}$$

Farbrother and Stacey (1990) developed and applied a fuzzy logic fixed rule base controller to the yaw channel of an ROV. This study provided encouraging results, but achieved limited success. This was due to the changing dynamics and external disturbances (such as noise) when applied to mine-counter measures. Encouraged by these simulations, and those of Sutton and Jess, Farbrother et *al.* (1991) have developed and applied a self organizing fuzzy logic controller to the same problem. Consequently, the controller achieved a much more robust performance in the presence of external disturbances.

Polkinghorne et *al.* (1996) reviewed the criteria to be considered in the performance assessment of such a control scheme with respect to full scale sea trials obtained on an 11 metre vessel capable of approximately 20-knots. Results highlighted the somewhat heuristic nature of the selection of suitable gains for scaling the penalisation of poor performance signals.

2.3 Artificial Neural Network Methods

The artificial neural network (ANN) is a biologically inspired computing technique (Figure 2.5) that in its simplest form is a fully connected structure of basic units which are themselves based on the McCulloch and Pitts (1943) neuron model shown in Figure

2.6. This model forms the basis for the early perceptron learning algorithm [Rosenblatt (1962)] and the later, and now widely known, backpropagation feedforward algorithm of Rumelhart and McClelland (1986) for training the multi-layer perceptron (MLP). With an ability to approximate non-linear functions such feedforward networks have been used extensively for classification and recognition problems [Sejnowski and Rosenberg (1987), Beale and Jackson (1990)] for which more conventional techniques would have been less tractable. Figure 2.7 illustrates the form of the feedforward, fully connected multi-layer perceptron.

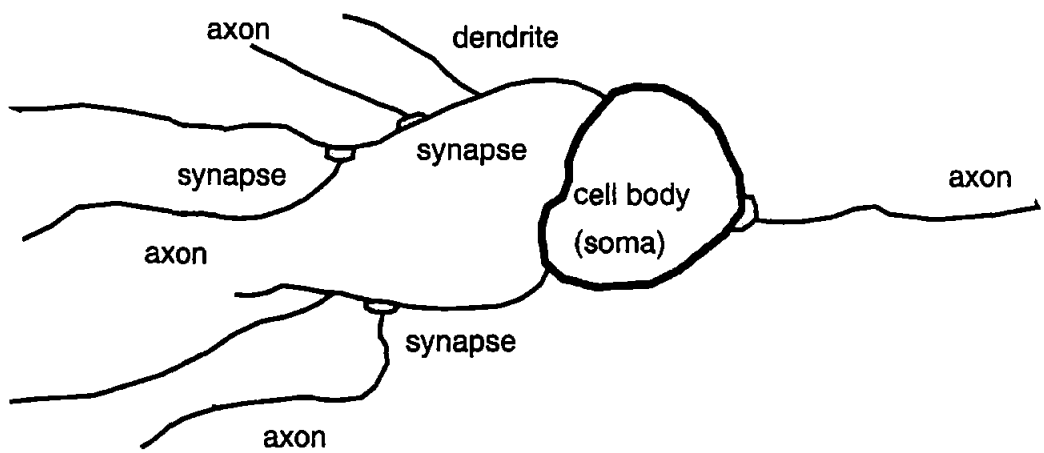


Figure 2.5: The basic features of a biological neuron.(after Beale and Jackson(1990)).

Here the input vector x is mapped to the output vector y via the nodes in the hidden layer j and the weighted connections w_{ji} and w_{kj}:

$$y = f\left(x, w_{ji}, w_{kj}\right) \qquad x \in R^n, y \in R^m \tag{2.5}$$

There are three methods generally used in order to train neural networks, these being supervised learning, reinforcement learning and unsupervised learning.

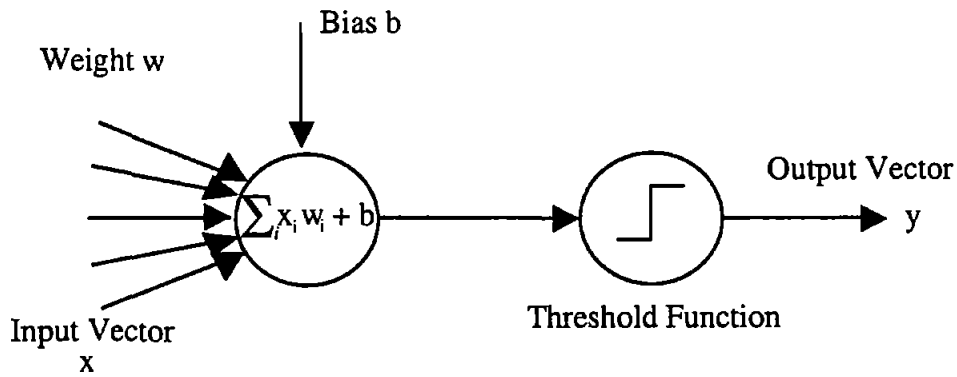


Figure 2.6: The McCulloch and Pitts neuron model (after McCulloch and Pitts (1943)).

Supervised learning depends upon a target being available for each input pattern to compare to the actual output of the network. Reinforcement learning does not require a target output to be available but only a cost function signal to indicate whether changes in weight connections provide better or worse performance. Unsupervised learning is used when no target pattern and no cost function are easily available to perform training. This requires that the network itself has the ability to recognize common features across the range of input patterns and modifies its internal state to model the features found in the training data.

One commonly used method of supervised learning is the aforementioned backpropagation (BP) rule. For each set of input data there is a corresponding output set, thus enabling the computation of an error measure between actual and desired output data sets on presentation of an input data set. The alteration of weights and biases within the ANN is therefore possible. The BP algorithm aims to alter the ANN weights and biases so that progression is made in the direction of the greatest rate of change of error reduction. To allow this behaviour, a function based upon the derivative of the error at the output of the previous layer is backpropagated through the ANN on completion of each training epoch or iteration. This principle is highlighted in Figure 2.8. Further details can be found in Beale and Jackson (1990).

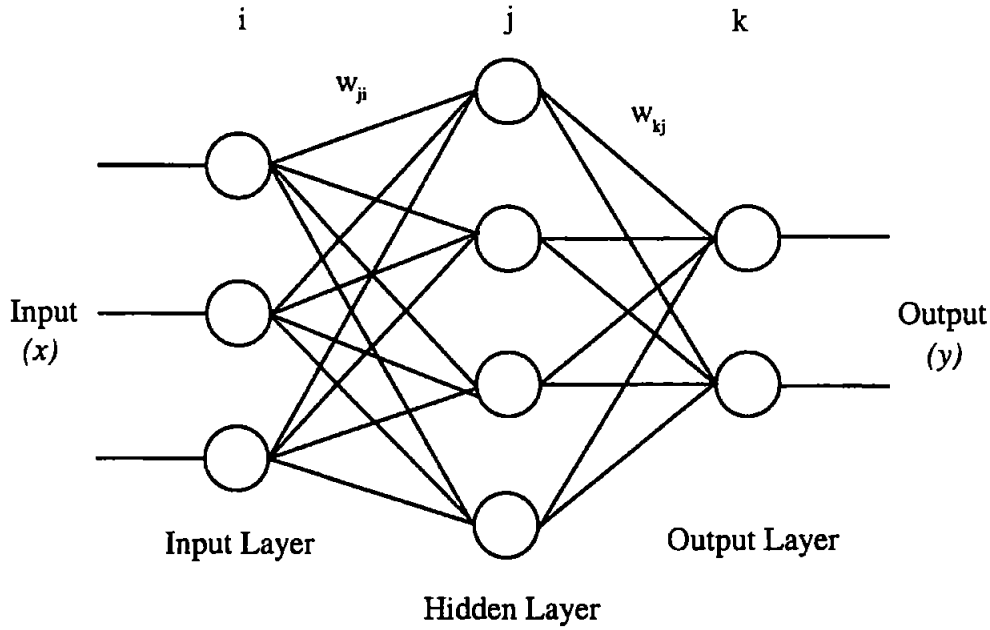


Figure 2.7: The feedforward multi-layer perceptron.

2.3.1 Applications

Owing to the ability of ANNs to represent non-linear mappings of systems for which the underlying rules are unknown, they have been applied extensively to the control of marine vehicles. In the majority of applications to date, ANNs have been employed as a robust controller, where the network is generated through a period of learning and reinforcement. Usually the network is then frozen at this point because to perform on-line calculations generally requires a large amount of computer power. This problem has limited most controllers to date to be non-adaptable once a suitable level of control has been achieved.

Yuh (1990), in his paper on the application of an ANN controller for an AUV, describes the application of two and three layer architectures to the problem of trajectory control. The two layered architecture is seen to provide a limited degree of success, proving unreliable in the advent of unknown vehicle dynamic situations and environmental uncertainties. A three layered network, conversely, gave much more

robust performances but was insufficiently documented as concerns simulation results to make satisfactory conclusions on its overall performance.

Waldock et *al.* (1995) used a BP algorithm to train architectures of differing structure to control an UUV to follow the terrain of the seabed. The chosen architecture, based on the smallest sum squared error after testing, was the single hidden layer three neuron feedforward network. Initial training was then improved upon by using an alopex algorithm to find a minimum global error solution.

Johnson (1995) performed a similar study again using differing architectures, but using a chemotaxis algorithm over the commonly employed BP algorithm. The resulting two hidden layered ten neuron per layer network was seen to give better overall control than a classical PID controller with which it was compared in terms of reduced thruster revolutions.

An example of a neural network controller for an AUV which uses an on-line learning technique to update nodal weights has been designed and implemented by Venugopal et *al.* (1992). A four layered neural network is used which is trained by the BP algorithm. This implementation differs from many on-line strategies in that it employs a gain factor (proportional to the inverse of the Jacobian of the dynamics) which adapts to changes in the vehicle dynamics in order that the controller accounts for dynamical change in its control action. Simulation results are divided into three distinct categories: (i) maintaining a desired pitch, (ii) maintaining a desired heading and (iii) maintaining a desired depth.

In case (i), the controller (beginning from a random start) soon achieves a good control action. Perturbations are introduced into the vehicles forward speed dynamics to assess the performance of the adaptive gain network. Better simulation results were achieved

when the learning rate of the network was increased, suggesting that the ANN be better equipped to generalize to disturbances in vehicle dynamics at a higher learning rate.

In case (ii), an increased learning rate provided faster convergence to the desired heading but also increased oscillatory behaviour about the desired heading. However, good simulation results were achieved and as time increased the on-line learning capabilities of the ANN controller reduced the oscillatory behaviour. The vehicle parameters were again disturbed and it was noted that the higher learning rate again improved the controller's ability to adapt to varying dynamics.

Finally in case (iii), the controller achieved adequate control action even at a small learning rate.

It is obvious from this study that the learning rate is a critical feature of the ANN controllers performance in the event of varying vehicle dynamics. A criticism of this study is the lack of attention devoted to this important feature. For example, could the learning rate be increased incrementally, perhaps less initially to avoid oscillatory behaviour and then in larger increments as on-line learning improves controller performance in latter stages? Also the paper mentions the applicability of the chosen architecture and technique to multi input - multi output (MIMO) controller design, but no simulations were even mentioned. Obviously it is a difficult task to choose an appropriate ANN learning rate for a MIMO controller which will optimize the learning rate of all three degrees of freedom simultaneously. However, this paper presents an alternative approach to the usual ANN AUV implementations and provides encouraging results for single input – single output (SISO) control of the vehicle in the presence of varying vehicle dynamics.

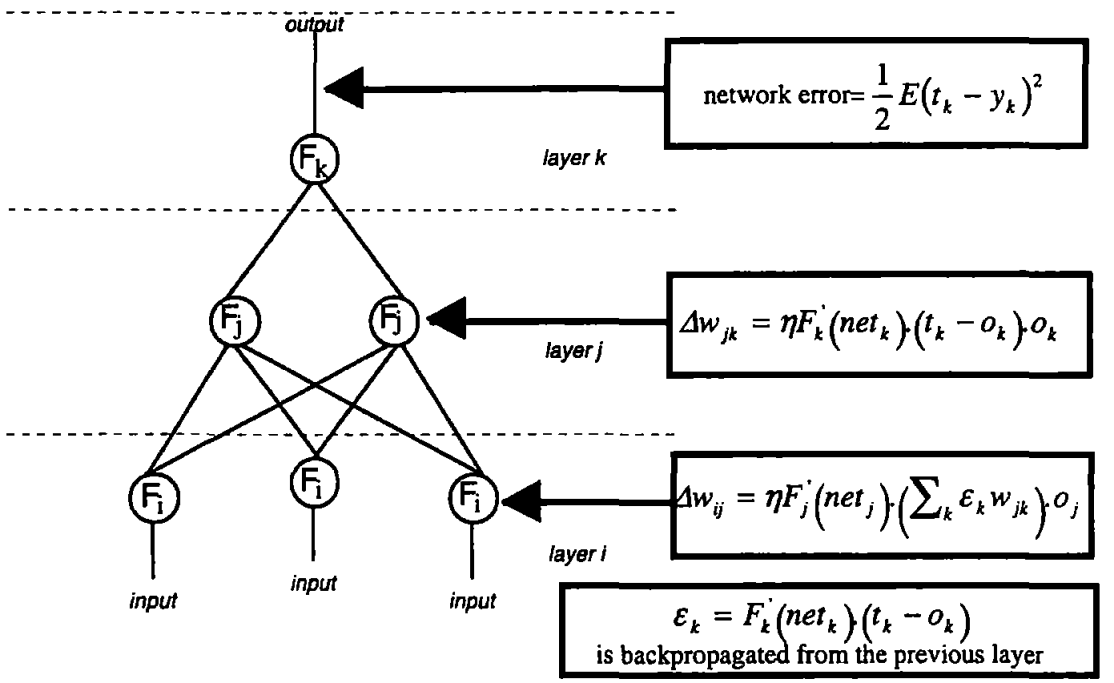


Figure 2.8: Error backpropagation (after Waldock(1995)).

Another interesting application of an ANN control strategy for an UUV is that given by Ishii et *al.* (1993). Again this is an adaptive on-line control scheme called 'imaginary training' applied to a 'self organizing neural-net controller' (SONCS) which attempts to improve or shorten the training times involved in the learning phase of the control scheme. The SONCS consists of real world ANN and imaginary world ANN sections that are connected to form an adaptive link. In the real world ANN section the AUV is operated according to its objective, and in the imaginary world ANN section the imaginary training takes place based upon simulated state variables which are calculated without knowledge of actual simulation data. The imaginary training is executed independently of the actual operations of the AUV via an identification ANN. The controller of the real world part adjusts its network weights based on the adjusted values obtained in the imaginary world part. This ANN is chosen for its specific structure and consequent ability to identify dynamical systems via a forward model network.

To assess the effectiveness of the imaginary training, the proposed scheme was implemented on the *Twin-Burger* AUV for the yaw channel only. The resulting simulations show that the SONCS can control the heading of the AUV sufficiently well, even in the event of noise corruption in the teaching period. Also the controller shows good performance in the event of dynamical changes to the vehicle and its environment. One criticism proposed is the relatively infrequent sampling of the teaching data from the state variables, for the adaptation of the weights in the imaginary training controller network. It is thought that a vehicle with a relatively small time constant, such as an AUV, will be very susceptible to environmental and dynamical perturbations. Thus twenty seconds per sample of data is seen as a rather long time period in which to let the teaching of the real world controller weights remain 'untaught', especially in the initial stages of learning by the controller.

Yuh and Lakshmi (1993) document a multi-layered ANN controller that estimates the control error using a 'critic' or punish/reward system. The most appropriate architecture is again investigated in the initial stages of the paper. A three layered ANN controller is chosen and compared using three learning algorithms: the BP algorithm, the parallel recursive prediction error algorithm (PRPE) and the modified parallel recursive prediction error algorithm (MPRPE).

The critic equation upon which controller network weights are adapted is a function of the actual velocity vector and the desired velocity vector, where the desired velocity vector is computed via the desired position vector, actual position vector and sampling period. The equation is then based upon a one step performance measure of the vehicle position and velocity.

Case studies performed on the ANN controller were considered in the lateral plane, i.e. in the yaw, surge and sway degrees of freedom. Initial tests investigated the performance of each learning algorithm whilst keeping the vehicle parameters constant.

Resulting graphs indicate that the BP and MPRPE algorithms have good learning ability whilst the PRPE algorithm produces consistently high values of mean squared error in horizontal plane control. Results obtained using the MPRPE learning algorithm to train the ANN controller show that on-line training is effective and can adequately cope with the addition of random noise and also the effects of varying vehicle dynamics and parameters.

This study, although somewhat taking into account some of the cross-coupling effects of such a vehicle, does not provide any de-coupled results in this plane against which to compare these simulations. Thus no comparisons can be made in terms of controller performance as concerns reduced effectiveness in any particular channel arising due to any cross-coupling effects.

Kodogiannis et *al.* (1996) describe the application of a model predictive control (MPC) strategy whereby the model takes the form of a neural network. Real time implementation of the controller structure was assessed in both simulation experiments and within an on-line configuration for the 'Aquacube' underwater vehicle. Results were presented for SISO control of the vehicle based upon MPC with autoregressive recurrent (ARNN) and Elman neural networks, both models employing a five step ahead prediction horizon.

Although the Elman neural network controller is typically only suitable for modelling linear systems due to its simple architecture, it displayed effective trajectory following in comparison to the desired response. However, the ARNN was the more accurate of the two models and thus the preferred one. Future studies are said to include extensions to multivariable system control.

2.4 Neuro-Fuzzy Schemes

At present there is a great deal of interest concerned with the unification of neural networks and fuzzy logic to produce intelligent neuro-fuzzy controllers. The aim of this union is to retain the robustness, non-linear mapping ability and numerical data utilization of ANNs whilst incorporating the linguistic advantages of fuzzy logic. Thus a controller designed using a neuro-fuzzy approach has immediate advantages over either an ANN or fuzzy logic controller.

The structure of neuro-fuzzy controllers is invariably such that no previous knowledge of the modelled process is required for the ANN to identify the existing input/output mapping. (It should be noted however that a priori knowledge of the modelled process should improve the training time of such controllers.) It is thus possible that a controller can be designed which is applicable to other non-linear dynamic situations, even if knowledge of process dynamics is unknown.

Training of a neuro-fuzzy controller is usually based on one of two methods, either gradient descent as with the BP algorithm or reinforcement learning, although some applications have employed combinations of both methods. The hierarchical structure of such controllers means that the learning process usually consists of the ANN converging on an improved set of fuzzy parameters. Taylor (1995) suggests that using an ANN to imitate a fuzzy autopilot and thus producing a 'black box' form of controller is inferior to the aforementioned method whereby knowledge is retained of process dynamics.

The adaptive network based fuzzy inference system (ANFIS) was designed and implemented by Jang (1993). This approach uses an ANN that differs from most in that not all nodes are connected through weighted links. This has the consequence that not all the nodes are modifiable with respect to their weights. Jang has implemented the ANFIS in dynamic control of an inverted pendulum problem, whereby an

approximation to the Jacobian matrix is used for backpropagation of the overall system error as opposed to the network error, thus creating a specialized learning approach to neuro-fuzzy control. Figure 2.9 illustrates the typical layout of an ANFIS architecture.

Square nodes in this structure represent the parameter sets of the membership functions detailing the TSK fuzzy model. Circular nodes are thus static or non-modifiable and simply perform operations on the incoming signals, such as product or min calculations. Because the consequent functions are linear with respect to the network inputs, a hybrid learning rule is used for accelerating parameter adaptation based upon sequential least squares in the forward pass to identify the consequent parameters, and backpropagation in the backward pass for the premises.

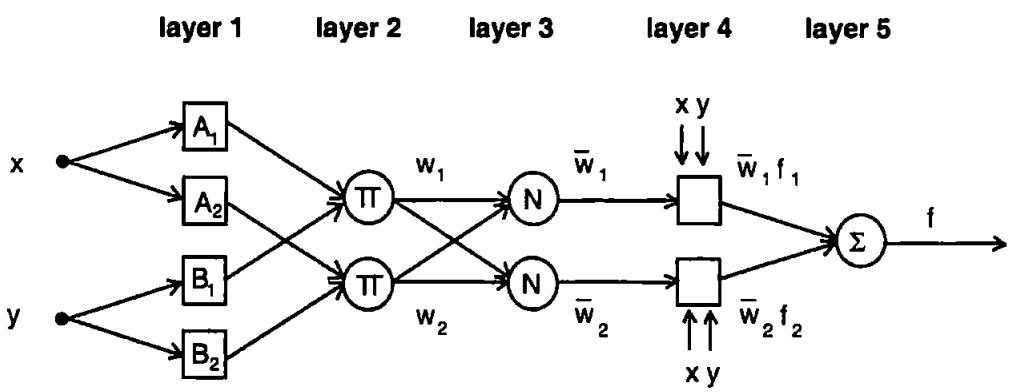


Figure 2.9: The ANFIS architecture for a Takagi-Sugeno-Kang FIS.

Jang and Sun (1995) give a review of fundamental and advanced developments in neuro-fuzzy synergisms for modelling and control. In this review the basic concepts of fuzzy logic and adaptive networks are discussed as a prelude to discussions on the hybrid learning rule of the ANFIS architecture and its superiority over the usual BP algorithm. To conclude the review a number of design techniques are given for neural and fuzzy controllers, and the common problems encountered in their implementation.

One such technique is 'BP through time and real time recurrent learning', a scheme whereby the controller, and often plant simulation blocks are replaced by adaptive ANNs which cascade to form a large single network. The parameters to be adjusted in this process are the fuzzy controller parameters in the ANFIS. This temporal learning approach was successfully employed by Nguyen and Widrow (1990), for control of a tractor-trailer vehicle in a confined zone operating environment.

Another implementation of neuro-fuzzy control by Jang and Gulley (1995) is that of gain-scheduled neuro-fuzzy control. This technique exists under certain conditions so that a first order TSK fuzzy model becomes a gain-scheduler that switches between several sets of feedback gains. The MATLAB fuzzy logic toolbox contains Jang's implementation of this technique applied to the inverted pendulum system, where the scheduling variable is the pole length and the control action is the smooth switching between three sets of feedback gains.

A recent application of neuro-fuzzy control is detailed by Tao and Burkhardt (1994), with application to the control of a flame process. In order to control the supply of the input variables gas and oxygen, a neural network based fuzzy logic controller is implemented through a personal computer. The optimal burning state is sought by the fine-tuning of the fuzzy parameters. Prior expert knowledge is incorporated into the control action through the initial fuzzy rule base. Two methods of network training were adopted; supervised learning and reinforcement learning. Supervised learning was employed in one instance when it was assumed that training data were available; reinforcement learning was otherwise used. This paper provides the reader with an alternative example of an application of neuro-fuzzy control. Although no results are presented the system is said to be able to control the flame to find its optimum state.

Taylor (1995) provides a review of neural-fuzzy algorithms for control applications. This work highlights the clear distinction between two classes of learning algorithm, those of supervised learning and those of unsupervised or reinforcement learning.

Tomera and Morawski (1996) employed the ANFIS scheme of Jang (1993) to produce an autopilot suitable for both course-keeping and course-changing behaviour. A sophisticated non-linear container ship model was used throughout the results. Fifteen fuzzy rules based on course-changing error and rate of change of course-changing error control inputs were adopted and tuned. The resulting autopilot proved equally as effective (if not more so) as a classical PID autopilot in light of simulated wind and wave disturbances for the limited simulations presented. Additionally, the ease of implementation and simplicity of the controller structure proved a strong point in favour of the method as an autopilot design technique.

Jang (1993b) has successfully applied the more computationally inexpensive temporal difference methods of Sutton (1988) in conjunction with his own ANFIS architecture to control the benchmark pole balancing on a moving cart problem. This approach falls into the supervised learning category due to the use of temporal backpropagation. The method therefore relies on a priori knowledge of the underlying model of the pole problem. This approach is seen to be capable of balancing the pole on the cart, when the cart moves unrestrictedly, after only one controller parameter set adjustment. The parameter consequents and antecedents are initially set to zero and cover the input space respectively. The controller was also seen to be robust to variations in pole lengths and initial conditions.

Barto and Anderson (1983) have applied a reinforcement algorithm to the balancing pole problem, whereby the reinforcements are used to update the weights of an ANN, the action selection network (ASN) which has a partially connected architecture and is derived from a set of fuzzy conditional statements. An action evaluation network

(AEN) is also used to generate a reinforcement signal based on the effectiveness of the previous control action. The learning aim of the ASN is to increase the output from the AEN, thus providing more improvements in control action. This controller required thirteen rules in its fuzzy rule base to balance the pole. Results showed good robustness to rule omission or degradation. This method also requires a model for off-line training or that a failure signal is generated, in on-line training, by the plant.

Hiraga et *al.* (1995) applied a fuzzy neural network (FNN) to the problem of ship collision avoidance, by the acquisition of suitable fuzzy control rules. Essentially, rules based upon ship steering control and also ship steering and speed control were generated through a backpropagation tuned FNN. The results obtained proved very effective when the object in the collision path approached from an angle greater than 4°. This was considered due to a 4° heading lying at the border of the left and fore steering decision region of the fuzzy sets. The autopilot thus could not define a correct evasive rudder angle. Future research is aimed at producing FNN collision avoidance controllers for the ship in the presence of multiple collision objects under more complicated conditions. It would also prove interesting from a control point of view to consider the effects of wind and wave disturbances upon the ship, as in the work of Lloyd(1989).

Albus (1975) designed and implemented the cerebellar model articulation controller (CMAC) architecture, an associative ANN which employs piecewise 'constant' basis functions, the number of which is determined by the designer via a generalization parameter. Also a number of knot functions is then necessary to fuse together these basis functions to cover the input space. The network output can consequently be written as a function of the sum of the individual weights of the network, which are determined from a supervised training scheme such as a least mean squares law.

If the CMAC network is used to interpret a fuzzy controller, as in Pedrycz (1993), the network consists of the following layers:

(a) A sensory layer - provided with inputs fuzzy sets of perhaps 'error' and 'change in error',

(b) An association layer - this consists of logical AND nodes and is used to aggregate the individual signals of the Sensory Layer.

(c) A post association layer - this consists of logical OR neurons is used to summarize the AND aggregates above.

(d) A defuzzification layer - transforming the results of all three above layers into one single valued output.

The learning scheme is by reinforcement and a single scalar value is used to determine a group of connections in the network. Also the amount of a' priori knowledge necessary to construct or design the controller is reduced.

An alternative neural-fuzzy control strategy, the differential competitive learning (DCL) algorithm, was proposed by Kosko (1992) and co-workers [Pacini and Kosko (1992), Kong and Kosko (1992)]. Individual neurons in the network architecture are in competition with each other. Fuzzy rules containing multiple antecedents are decomposed and then reformulated to produce more fuzzy rules which can be considered as the unions of individual decomposed rules, where fuzzy sets are defined as quantization vectors of membership function values. These are termed fuzzy associative memories (FAM) and combine to produce the FIS, each FAM representing a particular area of the input/output space. The aim of DCL is to cluster the quantization vectors and consequently generate fuzzy rules. The ANN architecture has an input layer

and a 'competition layer' connected together by a weight matrix and an intra-weight matrix in the competition layer that has positive diagonal elements and negative non-diagonal elements to excite the neurons. If a neuron becomes unexcited then learning will occur, caused by the weight changes' dependence on the change in competing neurons output. Thus many fuzzy rules may be defined but only those having a number of quantization vectors assigned to them are used. Kosko (1992) demonstrates successful controller performance using the DCL method, which is not reliant on a priori knowledge of the plant model.

Richter et *al.* (1996) developed a fuzzy-neural autopilot with predictive control capability for control of a Mariner Hull type ship. A neural model of the ship was identified using a pseudo random binary sequence input signal based upon a pre-definition of the crucial input/output ship relationships to be modelled. The design of the fuzzy autopilot was presented in some detail. Results claim a performance improvement of 50% over previous autopilot designs when employed in actual sea trials, yet no corroborating evidence is provided. It is suggested that the main failing with the majority of intelligent autopilots to date is the retrospective learning aspects, which produce corrective learning action if and when degraded performance is encountered. The heuristic nature of a typical FSOC falls into this category as the performance index is developed heuristically. By introducing the predictive element into the autopilot it is suggested that the strategy will possess the ability to learn correct commands in advance. However, no actual results for ship sea trials are presented in the paper. Additionally, some concern may be centred upon the selective pre-definition of training data pairs in producing the ship neural model. Due to the vast training history associated with the model (4 million epochs), if these training data pairs are incorrectly chosen the resulting model will not provide good generalization to unseen ship behaviour. Additionally, the overall structure of the autopilot is somewhat complicated in comparison to effective design techniques already described. Whilst the approach

holds great promise, results collected over comprehensive sea trials are required for completeness.

Sutton et *al.* (1996) investigated the use of ANNs in the design of fuzzy autopilots for controlling the yaw dynamics of a modern Royal Navy Warship model. A network was chosen based on the ANFIS network and subsequently trained using the BP, alopex [Unnikrishnan and Venugopal (1994)], chemotaxis [Koshland (1980)] and simulated annealing [Kirkpatrick et *al.* (1983)] algorithms. The input fuzzy sets for the autopilot were chosen as yaw error and yaw rate. Simulation results were given and comparisons made with a traditional PD linear autopilot. Overall, it was found that the autopilot trained using the simulated annealing algorithm performed the best with respect to overshoot, rise time and the integral square error of rudder angle and yaw error. However, a significant drawback of the simulated annealing algorithm is that it required longer training periods to converge than the gradient based algorithm of BP.

Future applications of neuro-fuzzy techniques to the control of marine vehicles are expected, mainly due to the fact that such techniques do not require a model of the process dynamics to produce a control action.

2.5 Summary: Fuzzy Logic, Neural Networks or an Informed Fusion?

Fuzzy logic control systems are inherently robust to non-linear, time-varying plant but remain reliant upon a rule base. Indeed, the self-organizing fuzzy logic controller develops its own rule base but requires some initial performance criterion. Such approaches have proved to be very successful at controlling UUVs, possibly due to their transparency.

Similarly, neural network paradigms are clearly very effective as controllers when no underlying model of the plant is available, but retain their 'black box' structure upon completion of learning.

The fusion of fuzzy logic and neural network control methodologies offers a means by which the inherently robust and non-linear nature of the fuzzy controller can be combined with the powerful learning abilities of the neural network. Basically, the human understanding of linguistic control rules needed to describe the trained neural network are available and the resulting fuzzy rule base can learn to adapt its performance suitably to improve its knowledge base.

Although there are examples of such fusions as applied to ship autopilot designs, limited attention has been given to the design of AUV autopilots using these techniques. Consequently, the use of neuro-fuzzy approaches to control the dynamic behaviour of an AUV could offer significant technological advances in the field of AUV autopilot design and thus provide an excellent research area.

References

Albus, J.S. (1975). A New Aproach to Manipulator Control: the Cerebellar Model Articulation Controller (CMAC). Transactions of the ASME, Journal of Dynamical Systems, Measurement and Control. pp220-227.

Barto, A.G. and Anderson, C.W. (1983). Structural Learning in Connectionist Systems. Proceedings of the Seventh Conference of the Cognitive Science Society. pp43-53.

Beale, R. and Jackson, T. (1990). Neural Computing: An Introduction. IOP Publishing Ltd. Bristol, U.K.

Craven, P. J., Sutton, R. and Burns, R. S. (1998). Control Strategies for Underwater Vehicles. Journal of Navigation, Vol. 51, No. 1, pp79-105.

Daley, S. and Gill, K.F. (1986). A Design Study of a Self-Organising Fuzzy Logic Controller. Proceedings of the Institution of Mechanical Engineers. Part C, 1, Vol. 200. pp59-69.

DeBitetto, P. (1995). Fuzzy Logic for Depth Control of UUVs. IEEE Journal of Oceanic Engineering. Vol. 20, No. 3, pp242-248.

Farbrother, H.N.R. and Stacey, B.A. (1990). Fuzzy Logic Control of an ROV. Modelling and Control of Marine Craft. Pourzanjani, M. and Roberts, G. (Eds). pp193-210.

Farbrother, H.N.R., Stacey, B.A. and Sutton, R. (1991). A Self-Organising Controller for an ROV. Proceedings of IEE Control 91. Edinburgh. pp499-504.

Farbrother, H.N.R. (1991). Aspects of Remotely Operated Vehicle Control - A review. RNEC Control Department Research Report, RNEC-RR-91025.

Harris, C.J., Moore, C.G. and Brown, M. (1993). Intelligent Control. Aspects of Fuzzy Logic and Neural Nets. World Scientific Series in Robotics and Automated Systems, Vol. 6, World Scientific Publishing Co. Pte. Ltd, Singapore.

Hiraga, I., Furuhashi, T., Uchikawa, Y. and Nakayama, S. (1995). An Acquisition of Operators Rules for Collision Avoidance Using Fuzzy Neural Networks. IEEE Transactions on Fuzzy Systems, Vol. 3, No. 3, pp280-287.

Ishii, K., Fujii, T. and Ura, T. (1993). An On-Line Adaption Method in a Neural Network Based Control System for AUVs. IEEE Journal of Oceanic Engineering. Vol. 20, No. 3, pp221-228.

Jang, J.S.R. (1993). ANFIS: Adaptive Network-based Fuzzy Inference System. IEEE Transactions on Systems, Man and Cybernetics. Vol. 23. pp665-685.

Jang, J.S.R. (1993b). Self Learning Fuzzy Controllers Based on Temporal Backpropagation. Department of Electrical Engineering and Computer Science. MIT, Cambridge, MA, USA.

Jang, J.S.R. and Sun, C.T. (1995). Neuro-Fuzzy Modelling and Control. Proceedings of the IEEE. Vol. 83, No. 3, pp378-406.

Jang, J.S.R. and Gulley, N. (1995). The Fuzzy Logic Toolbox for use with MATLAB. Natick, M.A., USA. The Mathworks Inc.

Johnson, C. (1995). Depth and Yaw Control of an Unmanned Underwater Vehicle using Neural Networks. MPhil Thesis. University of Plymouth.

Kirkpatrick, S., Gelatt, Jr., C. D. and Vecchi, M. P. (1983). Optimization by Simulated Annealing. Science, 220, pp671-680.

Kodogiannis, V. S., Lisboa, P. J. G. and Lucas, J. (1996). Neural Network Modelling and Control for Underwater Vehicles. Artificial Intelligence in Engineering, Vol. 1, pp203-212.

Kong, S.G. and Kosko, B. (1992). Adaptive Fuzzy Systems. IEEE Transactions on Neural Networks No.3, pp2-9.

Kosko, B. (1992). Neural Networks and Fuzzy Systems. Prentice Hall International.

Koshland, D.E. (1980). Bacterial Chemotaxis in Relation to Neurobiology, Annual Review of Neuroscience, Vol. 3, pp43-75.

Lloyd, A. R. J. M. (1989). Seakeeping: Ship Behaviour in Rough Weather. Ellis Harwood, U.K.

Mandic, N.J., Scharf, E.M. and Mamdani, E.H. (1985). Practical Application of a Heuristic Fuzzy Rule-Based Controller to the Dynamic Control of a Robot Arm. IEE Proceedings. Part D, Vol. 132, No. 4, pp190-203.

McCulloch, W.S. and Pitts, W. (1943). A Logical Calculus of the Ideas Imminent in Nervous Activity. Bulletin of Mathematical Biophysics. Vol. 5, pp115-133.

Nguyen, D.H. and Widrow, B. (1990). Neural Networks for Self Learning Control Systems. IEEE Control Systems Magazine. pp18-23.

Pacini, P.J. and Kosko, B. (1992). Adaptive Fuzzy Systems for Target Tracking. Intelligent Systems Engineering. pp3-21.

Pedrycz, W. (1993). Fuzzy Control and Fuzzy Systems. Second Edition. J. Wiley and Sons Inc., Taunton, U.K.

Polkinghorne, M. N., Roberts, G. N. and Burns, R. S. (1996). Consideration of Performance Assessment Criteria Required for a Self-Organising Fuzzy Logic Autopilot. Proceedings of the Eleventh Ship Systems Control Symposium, Southampton, U.K., pp152-159.

Procyk, T.J. and Mamdani, E.H. (1979). A Linguistic Self-Organising Process Controller. Automatica. Vol. 15. pp15-30.

Richter, R., Burns, R.S., Polkinghorne, M.N. and Nurse, P. (1996). A Predictive Ship Control using a Fuzzy-Neural Autopilot. Proceedings of the Eleventh Ship Systems Control Symposium, Southampton, U.K.

Rosenblatt, F. (1962). Principles of Neurodynamics. Sparten Books. New York.

Rumelhart, D.E. and McClelland, J.L. (1986). Parallel Distributed Processing: Explorations in the Microstructures of Cognition. Volume 1. MIT Press. Cambridge, Massachusetts, USA.

Sejnowski, T. and Rosenberg, C.R. (1987). Parallel Networks that Learn to Pronounce English Text. Complex Systems. pp145-168.

Shao, S. (1988). Fuzzy Self-Organising Controller and its Application for Dynamic Processes. Journal of Fuzzy Sets and Systems. No. 26, pp151-164.

Smith, S.M., Rae, G.J.S., Anderson, D.T. and Shein, A.M. (1993). Fuzzy Logic Control of an Autonomous Underwater Vehicle. Proceedings of the First International Workshop on Autonomous Underwater Vehicles. Southampton, U.K. pp318-323.

Sutton, R.S. (1988). Learning to Predict by the Methods of Temporal Differences. Machine Learning 3. pp9-44. Kluwer Academic Publishers.

Sutton, R. and Jess, I.M. (1991). A Design Study of a Self-Organising Fuzzy Autopilot for Ship Control. Proceedings of the IMechE. Part I, Vol. 205. pp35-47.

Sutton, R., Taylor, S.D.H. and Roberts, G.N. (1996). Neuro-Fuzzy Techniques Applied to a Ship Autoplilot Design. Journal of Navigation, Vol. 49, No. 3, pp410-430.

Takagi, T. and Sugeno, M. (1985). Fuzzy Identification of Systems and its Applications to Modelling and Control. IEEE Transactions on Systems, Man and Cybernetics, Vol. 15, pp116-132.

Tanscheit, R. and Scharf, E.M. (1988). Experiments with the use of a Rule-Based Self-Organising Controller for Robotics Applications. Fuzzy Sets and Systems. Vol. 26. pp195-214.

Tao, W. and Burkhardt, H. (1994). Application of Fuzzy Logic and Neural Network to the Control of a Flame Process. IEE Intelligent Systems Engineering. Publication No. 395.

Taylor, S.D.H. (1995). The Design and Tuning of Fuzzy Autopilots using Artificial Neural Networks. MPhil Thesis, University of Plymouth/RNEC.

Tomera, M. and Morawski, L. (1996). Neural-Network-Based Fuzzy Logic Marine Autopilot. Third International Symposium on Methods and Models in Automation and Robotics, pp1207-1212.

Unnikrishnan, K.P. and Venugopal, K.P. (1994). Alopex: A Correlation - Based Learning Algorithm for Feedforward and Recurrent Neural Networks, downloaded from the Joint Academic Network.

Venugopal, K.P., Sudhakar, R. and Pandya, A.S. (1992). On-Line Learning Control of Autonomous Underwater Vehicles using Feedforward Neural Networks. IEEE Journal of Oceanic Engineering. Vol. 17, No. 4, pp308-318.

Waldock, M.I., Roberts, G.N. and Sutton, R. (1995). Terrain Following Control of an Unmanned Underwater Vehicle using Artificial Neural Networks. IEE Colloquium on Control and Guidance of Remotely Operated Vehicles. London, pp4/1-4/8.

Yamakazi, T. (1982). An Improved Algorithm for a Self-Organising Controller. Ph.D Thesis, University of London.

Yuh, J. (1990). A Neural Network Controller for Underwater Robotic Vehicles. IEEE Journal of Oceanic Engineering. Vol. 15, No. 3. pp161-166.

Yuh, J. and Lakshmi, R. (1993). An Intelligent Control System for Remotely Operated Vehicles. IEEE Journal of Oceanic Engineering. Vol. 18, No. 1, pp55-62.

Chapter 3

Underwater Vehicle Modelling and System Performance Criteria

3.1 Introduction

This aim of this chapter is to provide a detailed description of the autonomous underwater vehicle (AUV) simulation model, and to discuss system performance criteria used within this thesis. Additionally, close attention will be given to the design of benchmark autopilots with which to compare the results of the following chapter.

3.2 Modelling the Vehicle Dynamics

The model employed throughout this study was purposely designed to provide a common design framework within the United Kingdom UUV research community in the navigation, guidance and control (NGC) fields. If required, the cable dynamics

pertaining to a remotely operated vehicle (ROV) can be included during simulation. The vehicle is considered herein as an AUV, but the following description of the dynamics is equally applicable to a ROV. Obviously, the package alleviates the requirement for an in-depth study of the UUV background modelling work, facilitating research that addresses the more important NGC issues.

3.2.1 Equations of Motion

To implement the vehicle equations of motion use is made of a MATLAB/Simulink simulation model termed *Release Version 1.0/UUVmod-GEN* supplied by the Defence Evaluation and Research Agency (DERA), Sea Systems Sector, Winfrith. This model has been validated against standard DERA hydrodynamic code using tank test data and an experimentally derived set of hydrodynamic coefficients from the Southampton Oceanography Centre's (SOC) AUTOSUB vehicle.

The inertial terms within the AUV equations of motion are given by Eqn.(3.1):

$$m\left[\dot{u} - vr + wq - x_G\left(q^2 + r^2\right) + y_G\left(pq - \dot{r}\right) + z_G\left(pr + \dot{q}\right)\right] = X$$

$$m\left[\dot{v} + ur - wp + x_G\left(pq + \dot{r}\right) - y_G\left(p^2 + r^2\right) + z_G\left(qr - \dot{p}\right)\right] = Y$$

$$m\left[\dot{w} - uq + vp + x_G\left(pr - \dot{q}\right) + y_G\left(qr + \dot{p}\right) - z_G\left(p^2 + q^2\right)\right] = Z \tag{3.1}$$

$$I_x \dot{p} - (I_y - I_z)qr + I_{xy}\left(pr - \dot{q}\right) - I_{yz}(q^2 - r^2) - I_{xz}\left(pq + \dot{r}\right)$$
$$+ m\left[y_G\left(\dot{w} - uq + vp\right) - z_G\left(\dot{v} + ur - wp\right)\right] = K$$

$$I_y \dot{q} + (I_x - I_z)pr - I_{xy}\left(qr + \dot{p}\right) + I_{yz}\left(pq - \dot{r}\right) + I_{xz}(p^2 - r^2)$$
$$- m\left[x_G\left(\dot{w} - uq + vp\right) - z_G\left(\dot{u} - vr + wq\right)\right] = M$$

$$I_z \dot{r} + (I_y - I_x)pq - I_{xy}(p^2 - q^2) - I_{yz}\left(pr + \dot{q}\right) + I_{xz}\left(qr - \dot{p}\right)$$
$$+ m\left[x_G\left(\dot{v} + ur - wp\right) - y_G\left(\dot{u} - vr + wq\right)\right] = N$$

The hydrodynamic force model provided by the DERA is as follows:

$$
\begin{aligned}
X &= \frac{1}{2}\rho l^2\left[X'_{uu}u^2 + X'_{vv}v^2 + X'_{ww}w^2\right] \\
&+ \frac{1}{2}\rho l^3\left[X'_{\dot{u}}\dot{u} + X'_{vr}vr + X'_{wq}wq\right] \\
&+ \frac{1}{2}\rho l^2u^2\left[X'_{uu\delta bp}\delta_{bp} + X'_{uu\delta bs}\delta_{bs} + X'_{uu\delta sp}\delta_{sp} + X'_{uu\delta ss}\delta_{ss}\right] \\
&+ \frac{1}{2}\rho l^2u^2\left[X'_{uu\delta bru}\delta_{bru} + X'_{uu\delta brl}\delta_{brl} + X'_{uu\delta sru}\delta_{sru} + X'_{uu\delta srl}\delta_{srl}\right] \\
&+ \frac{1}{2}\rho l^4\left[X'_{qq}q^2 + X'_{rr}r^2 + X'_{pr}pr\right] + X_{wave} + G_X
\end{aligned}
$$

$$
\begin{aligned}
Y &= \frac{1}{2}\rho l^2\left[Y'_{uu}u^2 + Y'_{uv}uv + Y'_{vw}vw + Y'_{v}vT_p + Y'_{v}vT_s\right] \\
&+ \frac{1}{2}\rho l^2u^2\left[Y'_{uu\delta bru}\delta_{bru} + Y'_{uu\delta brl}\delta_{brl} + Y'_{uu\delta sru}\delta_{sru} + Y'_{uu\delta srl}\delta_{srl}\right] \\
&+ \frac{1}{2}\rho l^3\left[Y'_{\dot{v}}\dot{v} + Y'_{up}up + Y'_{ur}ur + Y'_{vq}vq + Y'_{wp}wp + Y'_{wr}wr\right] \\
&+ \frac{1}{2}\rho l^3\left[Y'_{u|r|\delta bru}u|r|\delta_{bru} + Y'_{u|r|\delta brl}u|r|\delta_{brl} + Y'_{u|r|\delta sru}u|r|\delta_{sru} + Y'_{u|r|\delta srl}u|r|\delta_{srl}\right] \\
&+ \frac{1}{2}\rho l^4\left[Y'_{\dot{p}}\dot{p} + Y'_{\dot{r}}\dot{r} + Y'_{p|p|}p|p| + Y'_{pq}pq + Y'_{qr}qr\right] + Y_{wave} + G_Y
\end{aligned}
$$

$$
\begin{aligned}
Z &= \frac{1}{2}\rho l^2\left[Z'_{uu}u^2 + Z'_{uw}uw + Z'_{vv}v^2\right] \\
&+ \frac{1}{2}\rho l^2u^2\left[Z'_{uu\delta bp}\delta_{bp} + Z'_{uu\delta bs}\delta_{bs} + Z'_{uu\delta sp}\delta_{sp} + Z'_{uu\delta ss}\delta_{ss}\right] \\
&+ \frac{1}{2}\rho l^2\left[Z'_{u|w|}u|w| + Z'_{uv}uv + Z'_{w}wT_b + Z'_{w}wT_s\right] \\
&+ \frac{1}{2}\rho l^3\left[Z'_{\dot{w}}\dot{w} + Z'_{uq}uq + Z'_{vp}vp + Z'_{vr}vr\right] \\
&+ \frac{1}{2}\rho l^3\left[+ Z'_{u|q|\delta sp}u|q|\delta_{sp} + Z'_{u|q|\delta ss}u|q|\delta_{ss} + Z'_{w|q/w|}w|q/w|T_b + Z'_{w|q/w|}w|q/w|T_s\right] \\
&+ \frac{1}{2}\rho l^4\left[Z'_{\dot{q}}\dot{q} + Z'_{pp}p^2 + Z'_{rr}r^2 + Z'_{pr}pr\right] + Z_{wave} + G_Z
\end{aligned}
\tag{3.2}
$$

and the hydrodynamic moment model is:

$$
\begin{aligned}
K &= \frac{1}{2}\rho l^3\left[K'_{uu}u^2 + K'_{uv}uv + K'_{vw}vw\right] \\
&+ \frac{1}{2}\rho l^3u^2\left[K'_{uu\delta bru}\delta_{bru} + K'_{uu\delta brl}\delta_{brl} + K'_{uu\delta sru}\delta_{sru} + K'_{uu\delta srl}\delta_{srl}\right] \\
&+ \frac{1}{2}\rho l^4\left[K'_{\dot{v}}\dot{v} + K'_{up}up + K'_{ur}ur + K'_{vq}vq + K'_{wp}wp + K'_{wr}wr\right] \\
&+ \frac{1}{2}\rho l^5\left[K'_{\dot{p}}\dot{p} + K'_{\dot{r}}\dot{r} + K'_{qr}qr + K'_{pq}pq + K'_{p|p|}p|p|\right] + K_{wave} + G_K
\end{aligned}
$$

$$
\begin{aligned}
M &= \frac{1}{2}\rho l^3\left[M'_{uu}u^2 + M'_{vv}v^2 + M'_{uw}uw\right] \\
&+ \frac{1}{2}\rho l^3u^2\left[M'_{uu\delta bp}\delta_{bp} + M'_{uu\delta bs}\delta_{bs} + M'_{uu\delta sp}\delta_{sp} + M'_{uu\delta ss}\delta_{ss}\right] \\
&+ \frac{1}{2}\rho l^3\left[M'_{u|w|}u|w| + M'_{uv}uv + M'_{w}wT_b + M'_{w}wT_s\right] \\
&+ \frac{1}{2}\rho l^4\left[M'_{\dot{w}}\dot{w} + M'_{uq}uq + M'_{vr}vr + M'_{vp}vp\right] \\
&+ \frac{1}{2}\rho l^4\left[M'_{q}qT_b + M'_{q}qT_s + M'_{u|q|\delta sp}\delta_{sp}u|q| + M'_{u|q|\delta ss}\delta_{ss}u|q|\right] \\
&+ \frac{1}{2}\rho l^5\left[M'_{\dot{q}}\dot{q} + M'_{pp}p^2 + M'_{rr}r^2 + M'_{pr}pr + M'_{q|q|}q|q|\right] + M_{wave} + G_M
\end{aligned}
$$

$$
\begin{aligned}
N &= \frac{1}{2}\rho l^3\left[N'_{uu}u^2 + N'_{uv}uv + N'_{vw}vw +\right] \\
&+ \frac{1}{2}\rho l^3u^2\left[N'_{uu\delta bru}\delta_{bru} + N'_{uu\delta brl}\delta_{brl} + N'_{uu\delta sru}\delta_{sru} + N'_{uu\delta srl}\delta_{srl}\right] \\
&+ \frac{1}{2}\rho l^4\left[N'_{\dot{v}}\dot{v} + N'_{up}up + N'_{ur}ur + N'_{wp}wp + N'_{wr}wr + N'_{vq}vq\right] \\
&+ \frac{1}{2}\rho l^4\left[N'_{r}rT_b + N'_{r}rT_s\right] \\
&+ \frac{1}{2}\rho l^4\left[N'_{u|r|\delta bru}\delta_{bru}u|r| + N'_{u|r|\delta brl}\delta_{brl}u|r| + N'_{u|r|\delta sru}\delta_{sru}u|r| + N'_{u|r|\delta srl}\delta_{srl}u|r|\right] \\
&+ \frac{1}{2}\rho l^5\left[M'_{\dot{r}}\dot{r} + M'_{\dot{p}}\dot{p} + N'_{pq}pq + N'_{qr}qr + N'_{r|r|}r|r|\right] + N_{wave} + G_N
\end{aligned}
\tag{3.3}
$$

where the hydrostatic terms ($G_{X,\ldots,N}$) acting upon the vehicle are commonly referred to as restoring forces and moments.

Naturally, gravitational forces act down through the vehicle centre of gravity, whereas the force provided by the buoyancy of the vehicle acts through the vehicle centre of buoyancy. With respect to the underwater vehicle used within this study, the Euler angle representations of these restoring forces and moments are thus given by Eqn.(3.4):

$$\begin{aligned} G_X &= -(W-B)\sin(\theta) \\ G_Y &= (W-B)\cos(\theta)\sin(\phi) \\ G_Z &= (W-B)\cos(\theta)\cos(\phi) \\ G_K &= (y_G W - y_B B)\cos(\theta)\cos(\phi) - (z_G W - z_B B)\cos(\theta)\sin(\phi) \\ G_M &= -(x_G W - x_B B)\cos(\theta)\cos(\phi) - (z_G W - z_B B)\sin(\theta) \\ G_N &= (x_G W - x_B B)\cos(\theta)\sin(\phi) + (y_G W - y_B B)\sin(\theta) \end{aligned} \tag{3.4}$$

With respect to Eqn.(3.1), Eqn.(3.2), Eqn.(3.3) and Eqn.(3.4) the following parameters describe the AUV model used herein:

$W = 35316$ N	$B = 35316$ N	$l = 7.0$ m	$m = 3600$ kg
$\rho = 1025.2$ kgm^{-3}	$I_x = 320$ kgm^2	$I_y = 8304$ kgm^2	$I_z = 8304$ kgm^2
$I_{xy} = 0$ kgm^2	$I_{xz} = 0$ kgm^2	$I_{yz} = 0$ kgm^2	$g = 9.81$ ms^{-2}
$x_G = 0.34$ m	$y_G = 0$ m	$z_G = 0.02$ m	
$x_B = 0.34$ m	$y_B = 0$ m	$z_B = 0$ m	

It should be noted that whilst the full 6 degree of freedom equations of motion are reproduced here, the hydrodynamic coefficients of the model remain the property of the DERA.

3.2.2 Modelling Assumptions

As aforementioned, the effects of forces and moments due to the inclusion of a ROV umbilical are ignored within this thesis, the vehicle being considered of an autonomous form.

The calculation of the vehicle path relative to the earth fixed frame of reference is performed through an Euler co-ordinate transformation, as given in Fossen (1994). It should be noted however that other co-ordinate transformations can be employed (Fjellstad and Fossen (1994)).

All simulations assume that the AUV remains below 30 metres (approximately) as sea surface effects arising within the surface layer are not modelled within the simulation package.

3.2.3 Actuator Modelling

In addition, the model structure also takes into account the dynamic behaviour of the actuators by describing them as first order lags with appropriate slew rate limits. The movement of these actuators is defined such that a diving turn to port makes all control angles positive. Consequently, a positive deflection of the rear hydroplanes will cause a negative pitching effect (down direction) and a positive deflection of the rudders mounted at stern causes a negative yawing movement (to port), as shown in Figure 3.1:

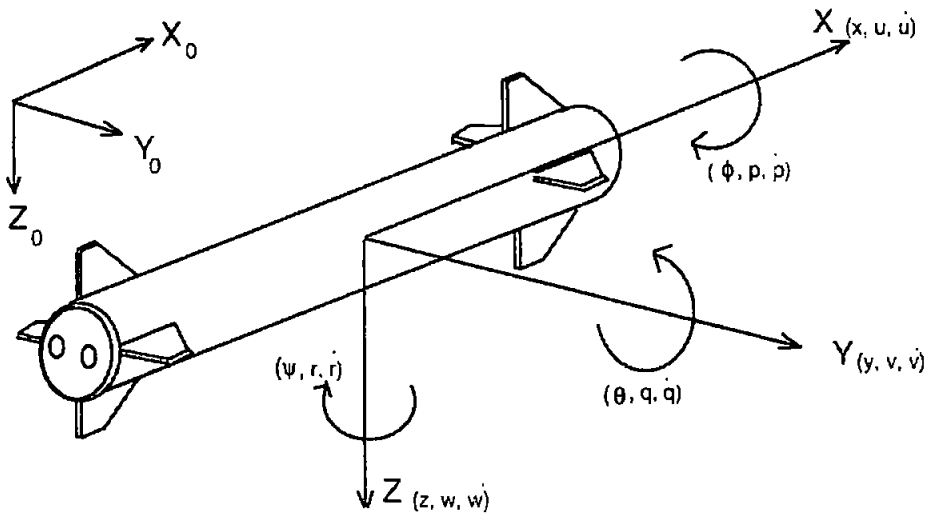


Figure 3.1: The body-fixed and earth-fixed reference frames.

Table 3.1 provides information concerning the angular and slew rate limitations imposed upon the vehicle actuators. Additionally, the maximal thrust available from each set of thrusters is given for completeness.

Thrusters	Limitation	Control surfaces	Limitation
Main longitudinal (X-axial)	± 450 N	Canard rudders (bow)	± 25.2 deg
		slew rate – canards	± 9.9 deg/s
Horizontal auxilliary (Y-axial)	± 120 N	Stern rudders	± 25.2 deg
		slew rate – sterns	± 9.9 deg/s
Vertical auxilliary (Z-axial)	± 120 N	Hydroplanes (bow)	± 25.2 deg
		slew rate – bow hydro	± 9.9 deg/s
		Hydroplanes (stern)	± 25.2 deg
		slew rate – stern hydro	± 9.9 deg/s

Table 3.1: Limitations imposed on the AUV actuators.

3.2.4 Sea Current Model

A disturbance model is included within the simulation package. This block describes the forces arising from sea currents within the hydrodynamic forces and moments (Eqn(3.2) and Eqn.(3.3)) as calculated along the translational axes as follows (Marshfield (1992)):

$$\begin{aligned}
X_{current} &= X_{velocity}[\cos(\psi)\cos(\theta)] + Y_{velocity}[-\sin(\psi)\cos(\theta)] \\
&+ Z_{velocity}[\sin(\theta)] \\
Y_{current} &= X_{velocity}[-\sin(\psi)\cos(\phi) + \cos(\psi)\sin(\theta)\sin(\phi)] \\
&+ Y_{velocity}[-\cos(\psi)\cos(\phi) - \sin(\psi)\sin(\theta)\sin(\phi)] + Z_{velocity}[-\cos(\theta)\sin(\phi)] \\
Z_{current} &= X_{velocity}[\sin(\psi)\sin(\phi) + \cos(\psi)\sin(\theta)\cos(\phi)] \\
&+ Y_{velocity}[\cos(\psi)\sin(\phi) - \sin(\psi)\sin(\theta)\cos(\phi)] + Z_{velocity}[-\cos(\theta)\cos(\phi)]
\end{aligned} \tag{3.5}$$

where the terms $X_{velocity}$, $Y_{velocity}$ and $Z_{velocity}$ represent the constant current component velocities of the total current velocity vector with respect to the X, Y and Z axes of the earth fixed frame of reference. The resulting current forces in each axis are obtained through a direction cosine matrix translation involving the yaw, pitch and roll angles.

3.3 System Performance Specifications and Autopilot Selection Criteria

The selection of a suitable autopilot model is clearly dependent on various criteria. In order that each autopilot is chosen based upon its merits, validation and verification tests must be designed which are unbiased and interpretable.

3.3.1 Suitability of Available Data

Within any modelling paradigm, the suitability of the resulting model is highly dependent on the amount and quality of the data used to generate it. It is one problem to create models which behave like the function from which the data has been gathered. However, to produce final models that generalize (interpolate) well to unseen inputs requires the data set used during training to suitably represent this function. Various techniques are available for assessing the performance of an autopilot including:

- **Cross validation** – traditionally the data set taken from the modelled function is split into two portions. One portion is employed during the learning phase, and the remaining portion is used to cross validate the resulting model.

- **Verification** – given that a model has been validated, it is then usually verified. This process concerns the testing of a model over a wide range of possible inputs to assess its overall suitability to the required task.

- **Hypothesis testing** – a set of models is produced which is designated as the model set. A cost function, which is often based on model size and model accuracy, is then formed and the model that displays the lowest cost is chosen as the most suitable.

- **Expert inspection** – can be exploited if the resulting model is in a simple enough form. A property of fuzzy models is that they typically exhibit good inspection properties, and are linguistically transparent. An expert can therefore verify the resulting autopilot by direct inspection. Examination of the resulting control surface (for a 2 input – 1 output system) can provide an insight into the smoothness of interpolation between rules within the autopilot rulebase.

- **Correlation testing** – if a more rigorous statistical approach to model selection is sought, the most commonly used technique is residual examination. A set of monomial functions is developed based on the product of input vector elements, output vector elements and errors. If any correlation exists between the residuals and the monomial functions the model is deemed unsuitable.

3.3.2 Model Validation and Verification

Throughout this work the autopilot models are based upon fuzzy linguistic rules which are subsequently tuned to improve their performance. The tuning methods employed within this thesis lend themselves naturally to the use of cross validation testing and verification.

To verify the performance of a given autopilot, certain AUV modelling performance criteria are also specified within this section. To quantify off-course error, yaw induced

roll motion, course-changing control effort and roll-minimizing control effort the following performance measures based upon the integral square of error over time (ITSE) were employed respectively (in their discrete form):

$$\psi_{\varepsilon} = \int_{t_1}^{t_2} (\psi_d - \psi_a)^2 dt \tag{3.6}$$

$$\phi_{\varepsilon} = \int_{t_1}^{t_2} (\phi_d - \phi_a)^2 dt \tag{3.7}$$

$$\delta_{can} = \int_{t_1}^{t_2} (\delta_{can-d} - \delta_{can-a})^2 dt \tag{3.8}$$

$$\delta_{hydro} = \int_{t_1}^{t_2} (\delta_{hydron-d} - \delta_{hydron-a})^2 dt \tag{3.9}$$

where ψ_d, ϕ_d, δ_{can-d} and δ_{hydro} represent desired yaw angle, roll angle, canard demand and stern hydroplane demand respectively; ψ_a, ϕ_a, δ_{can-a} and $\delta_{hydro-a}$ represent actual yaw angle, roll angle, canard angle and stern hydroplane angle respectively. To assess the course-changing response speed of the AUV model, figures pertaining to the rise time ($T_{R,\psi}$) were collected. Rise time is considered here as the time to reach 99 per cent of the course-change demand.

In addition to the quantitative performance measures of Eqn.(3.6) – Eqn.(3.9) each autopilot is assessed qualitatively to provide an overall measure of applicability to the required control task.

3.3.3 Robustness Experiments

Robustness (in this context) refers to the ability of the closed loop system, when employing an autopilot, to perform satisfactorily in light of unmeasured environmental

disturbances and vehicle configuration changes. Clearly, for an AUV it is required that the autopilot remains effective when operating at varying vehicle speeds, changing vehicle payloads and in the presence of environmental disturbances, such that all available sensor and on board systems remain functional. To simulate these effects, autopilot robustness to forward speed is tested at the nominal design speed of 7.5-knots and also at 5 and 10-knots. These tests are imperative because the equations of motion are non-linear with respect to the forward speed of the vehicle.

Additionally, variations within the AUV hydrodynamic coefficients are simulated. Robustness to mass and hydrodynamic coefficient variations is an important facet of any autopilots suitability to the given task.

Finally, if appropriate, the AUV model is simulated using a line of sight (LOS) guidance algorithm in the presence of sea current disturbances. This provides a method by which autonomous guidance may be implemented. This guidance law is determined by pre-specifying a number of target way-points $[X_k, Y_k]$; $k=1,2,\ldots,n$, which are stored in the vehicles mission planner prior to embarkation. Essentially, the guidance algorithm calculates the desired heading angle between the vehicles current position $[x_a(t), y_a(t)]$ and that of the target way-point using the following trigonometric rule:

$$\psi_d = \tan^{-1}\left(\frac{X_k - x_a(t)}{Y_k - y_a(t)}\right). \tag{3.10}$$

The next way-point is then selected given that the vehicle is within a radius of acceptance β of the current way-point, where β is calculated as follows:

$$[X_k - x_a(t)]^2 + [Y_k - y_a(t)]^2 \leq \beta^2. \tag{3.11}$$

An example of such a way-point is given in Figure 3.2. Autopilot robustness to sea current disturbances is simulated whilst employing this guidance law.

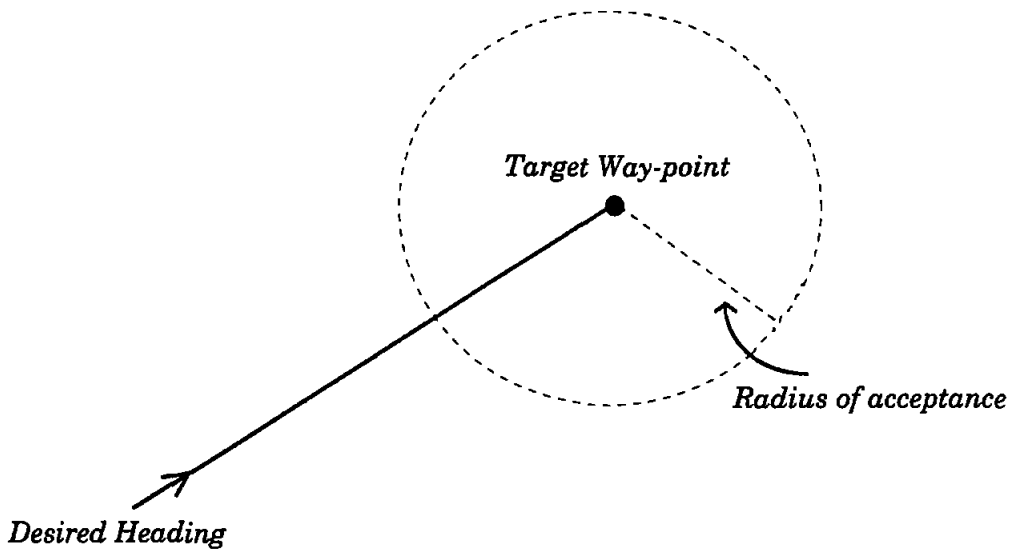


Figure 3.2: A typical radius of acceptance β.

3.4 Open Loop Simulations

Due to the high complexity and non-linearity of the AUV model, initial simulation experiments were conducted in the absence of a controller. This stage of the design process is necessary as the behaviour of the plant must be identified in some manner before it can actually be controlled; it is important to know approximately what is to be controlled before an autopilot system can be designed.

With respect to the AUV model this implies some form of initial experiments designed to ascertain how the AUV reacts to the use of individual and also combinations of its fourteen actuators. As the channels to be investigated within this work are those of yaw and roll, the basis of this study was to ascertain which actuators are most effective at instigating motion, or indeed suppressing cross-coupled motion, in these channels.

3.4.1 Initial Experiments

The nominal surge velocity during these open loop tests was set to 7.5-knots (3.859 metres per second). Typically, when an underwater vehicle is travelling at speeds in excess of roughly 1-knot (0.5 metres per second) control surfaces are employed in

preference to thrusters. The hydrodynamic forces acting upon the rudders and hydroplanes of the vehicle at such speeds provides much greater manoeuvring potential than the use of auxiliary thruster mechanisms, and are thus a more efficient means of controlling the vehicle motion (Cowling (1996)). Additionally, low speed control using rudders and hydroplanes is subject to reversal effects.

The positioning of the thrusters and control surfaces on the AUV hull provides some insight into the most effective actuators for a particular manoeuvre. For example, the bow and stern rudders are almost always associated with lateral plane motions such as yawing and sway changing, as one would expect. Also the hydroplanes (mounted on the side of the hull section) are typically employed in longitudinal motion control as perhaps roll stabilizers or heave actuators.

Examination of the AUV dynamics provides an important insight into the use of actuator strategies for particular control scenarios. For example, the roll moment equation (K equation within Eqn.(3.3)) details the use of port and starboard stern hydroplanes as the actuators which are influential in roll manoeuvring. Indeed, yaw control could be effected by manipulation of the main axial port and starboard stern thrusters. However, Cowling and Corfield (1995) examined this approach and found it to be unsatisfactory due to induced roll effects, which produced a cyclic oscillation in the heading angle. This was considered to be a function of thruster switching and was subsequently considered an inappropriate control strategy, the response time also being relatively slow for this type of vehicle. A more appropriate means for achieving yaw control is via the use of locked upper and lower canard rudders. Employing these actuators in this manner leads to a cancellation of the rolling moment normally produced by the use of an individual rudder or differential main X-axial thruster strategy.

A simple open loop test was devised to compare the use of single and locked rudder strategies for course-changing. The open loop AUV model is subjected to a 25.2° step input on:

(i) the low canard rudder, and

(ii) the locked upper and lower canard rudders.

Results from experiment (i) are illustrated in Figures 3.3 and 3.4 whilst those pertaining to experiment (ii) are shown in Figures 3.5 and 3.6. (NB: The upper canard rudder was not employed individually as this lead to the AUV heaving in the negative direction and leaving the water!).

The AUV follows a clearly defined circle when employing locked canard rudders (Figure 3.5) whereas the use of the lower canard (Figure 3.3) leads to instability due to increased cross-coupling in the other AUV degrees of freedom. From the plots of Figure 3.4, it is clear that the pitch angle (θ) has reached -90° and thus a singularity in the Euler transformations between earth-fixed and body-fixed reference frames has occurred. Consequently, the vehicle behaviour becomes unstable beyond this point in the simulation.

Conversely, the open loop simulations employing locked canard rudders (Figure 3.6) display no evidence of such behaviour. This effect was borne out during extended simulation periods and varying degrees of rudder angle steps. Consequently, the upper and lower canards were used in locked formation throughout the remainder of the course-changing simulation studies.

Figure 3.7 shows the complete control authority of the AUV model. The torpedo-like appearance of the vehicle suggests its use as a mine-hunting device. However, various mission scenarios are envisaged ranging from under ice data collection to covert surveillance and ordnance disposal.

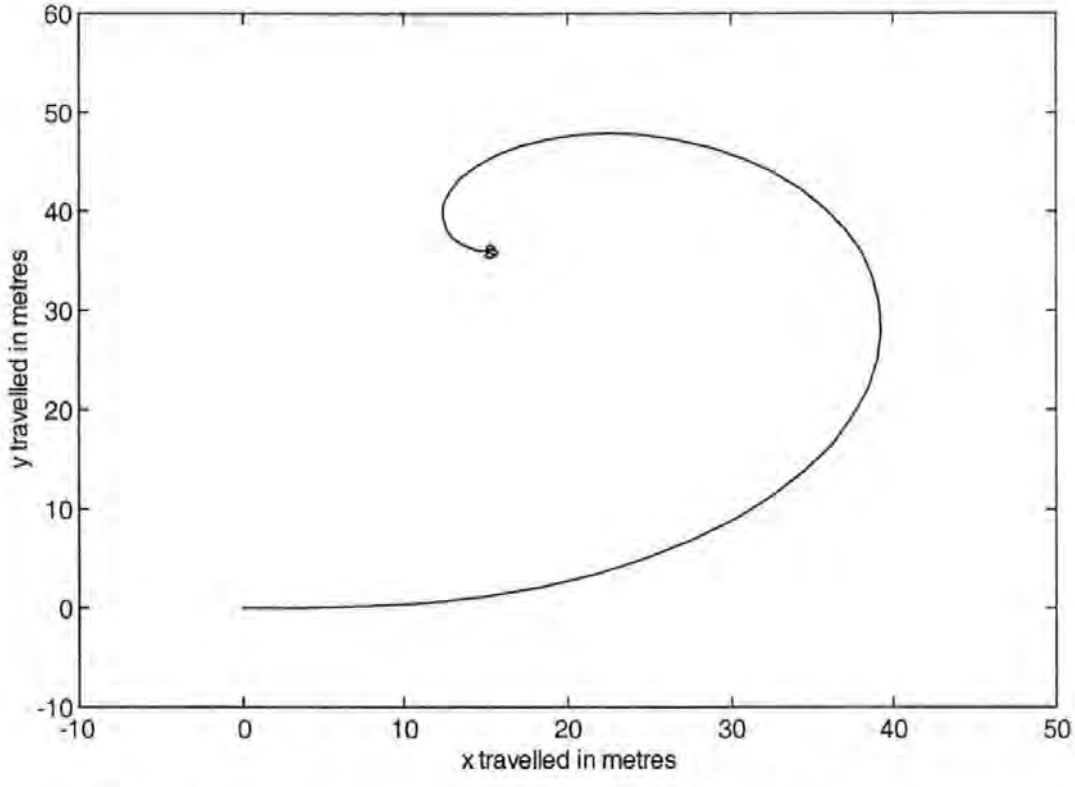


Figure 3.3: The x-y trajectory of the AUV when subjected to a saturated step input on the low canard rudder in the open loop.

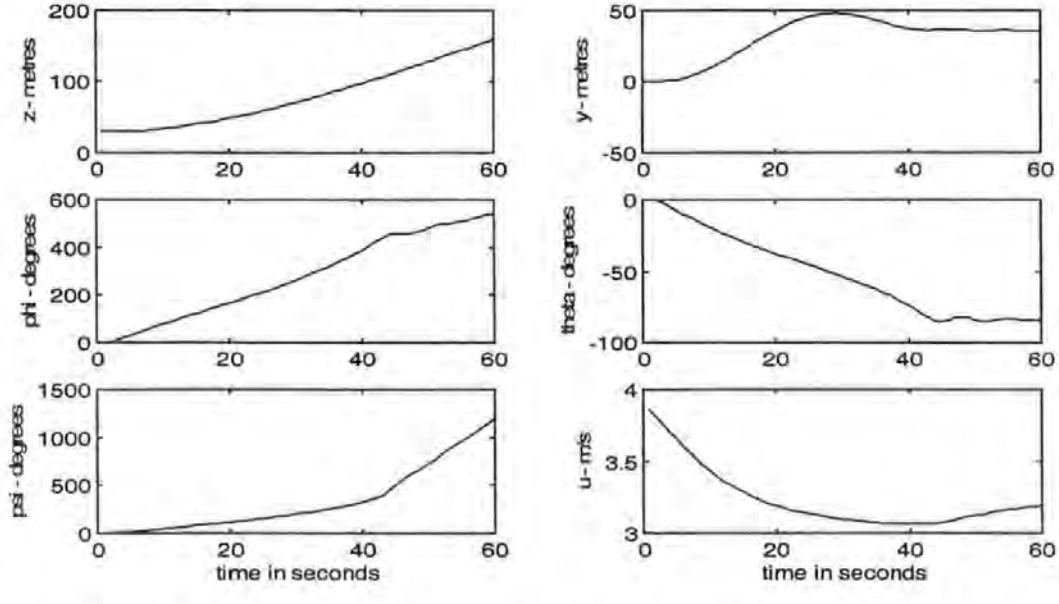


Figure 3.4: The cross-coupled motion of the AUV when subjected to a saturated step input on the low canard rudder in the open loop.

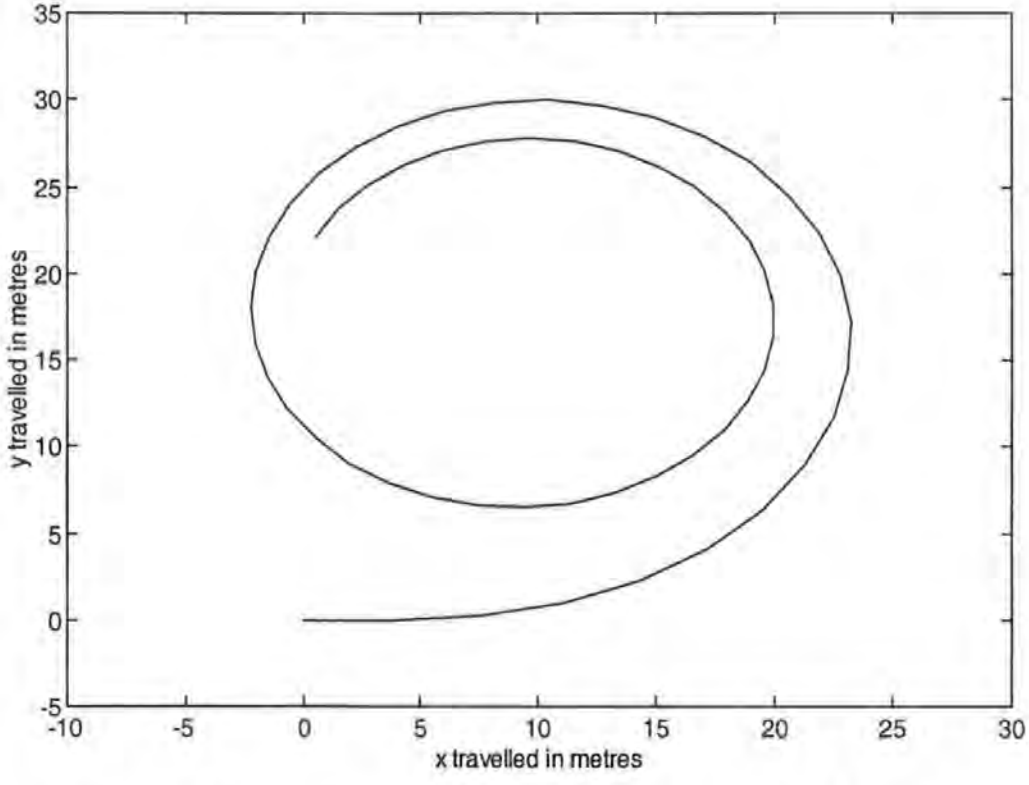


Figure 3.5: The x-y trajectory of the AUV when subjected to a saturated step input on the locked upper and lower canard rudders in the open loop.

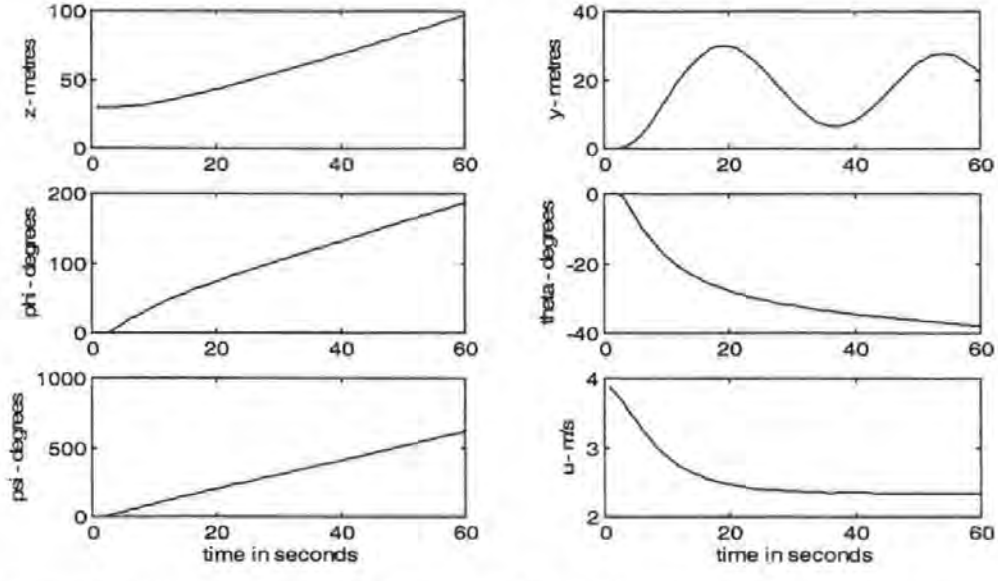


Figure 3.6: The cross-coupled motion of the AUV when subjected to a saturated step input on the locked upper and lower canard rudders in the open loop.

Due to the small metacentric height of the vehicle (2 centimetres) the propensity for the AUV to exhibit roll cross-coupling motion when performing course-changing is quite pronounced (as shown in Figures 3.4 and 3.6). Indeed, when applying a step input to differential port and starboard stern hydroplanes, the open loop response of the AUV in the roll channel exhibits a ramp type response. This behaviour can be accredited to the existence of a free integrator within the open loop roll dynamics, which can subsequently be approximated using:

$$\frac{\phi(s)}{\delta_{ster\_hyd}(s)} = \frac{20}{s(s+1)} \tag{3.12}$$

Chapter 5 discusses the implementation of various control strategies for roll regulation, one of which is based upon this transfer function approximation (Eqn.(3.12)) and was designed using a Nichols plot approach.

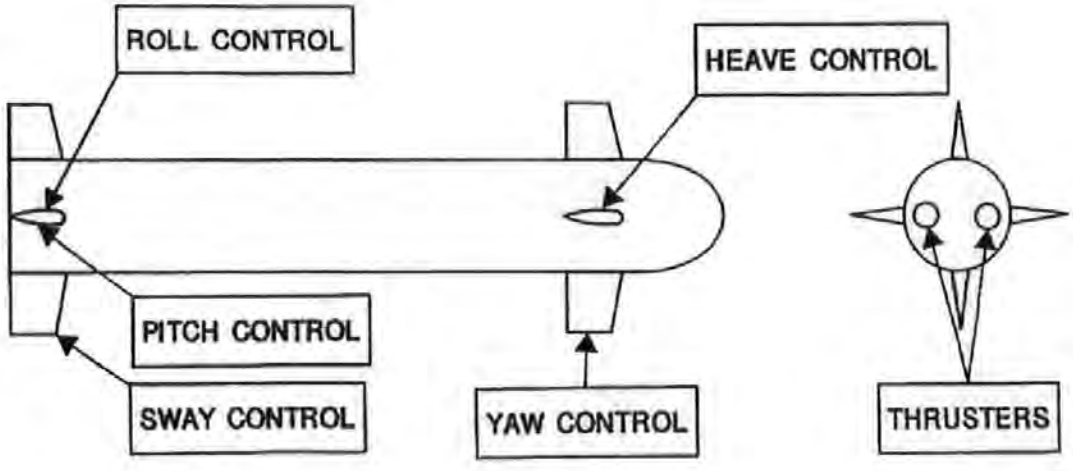


Figure 3.7: The complete control authority of the AUV.

3.5 Benchmark Autopilot Designs

The design of a classical autopilot for course-changing control of the AUV requires explicit knowledge of the open loop yaw dynamics. The following sections pursue the modelling and design of course-changing autopilots based on the following approximation to this dynamic response.

Yaw manoeuvring was found to be most effective by considering locked actuation of high and low canard rudders. As aforementioned, the use of locked rudders also eliminates some undesirable cross-coupling effects. The open loop yaw response with respect to a 25.2° step on differential upper and lower canard rudders is depicted in Figure 3.6. Approximating this response using the typical Nomoto first order transfer function form (Fossen (1994)) yields the open loop yaw to canard rudder transfer function:

$$\frac{\psi(s)}{\delta_{can}(s)} = \frac{33.518}{s(1.45s+1)} \tag{3.13}$$

3.5.1 Traditional Autopilot Design

From this transfer function it was possible to produce a well designed linear autopilot. Examination of the uncompensated system via a Nichols plot confirmed that a phase lead network autopilot would introduce suitable phase advance yet retain a reasonable bandwidth. The corresponding compensator was taken as:

$$G_{c,\psi}(s) = \frac{1+1.204s}{1+0.708s}. \tag{3.14}$$

3.5.2 Fuzzy Logic Autopilot Design

A detailed discussion of fuzzy logic control systems is omitted here so as not to impair the readability of this thesis. For the interested reader some excellent references are available [Sutton (1987)], [Lee (1990)], [Harris et *al.* (1993)], and [Jang et *al.* (1997)].

This chapter continues with the development of four Takagi-Sugeno-Kang (TSK) (Takagi and Sugeno, (1985)) style fuzzy autopilots for course-changing control of the AUV. The most suitable autopilot structure is highlighted and proposed for use within subsequent course-changing autopilot designs.

Fuzzy logic controllers were developed for course-changing control of the AUV using traditional derivative based methods (i.e. error and rate of change of error as inputs). The consequent functions were chosen as first order polynomial functions based upon the TSK style of fuzzy inference. However, for simplicity and to provide a direct comparison amongst each autopilot the consequent functions were initially taken as fuzzy singleton spikes, equally spaced over the output universe of discourse.

Partitioning of the input space was performed using the grid partitioning approach for simplicity. Fuzzy systems which involve a large number of input variables, often suffer from the curse of dimensionality, which states that:

> "*as the number of inputs increases, the number of fuzzy rules required also increases, but exponentially*"

which means that as the number of system inputs grows, the fuzzy model size often becomes prohibitively large. An alternative method of input space partitioning is to employ scatter partitioning. This approach places rules within the areas that contain the most rich data sources. Figure 3.8 illustrates these methods of partitioning where a) represents grid partitioning and b) scatter partitioning. Although scatter methods of rule positioning are efficient, they are heavily reliant upon the data set used to initialize the fuzzy rule base and were not considered at this juncture.

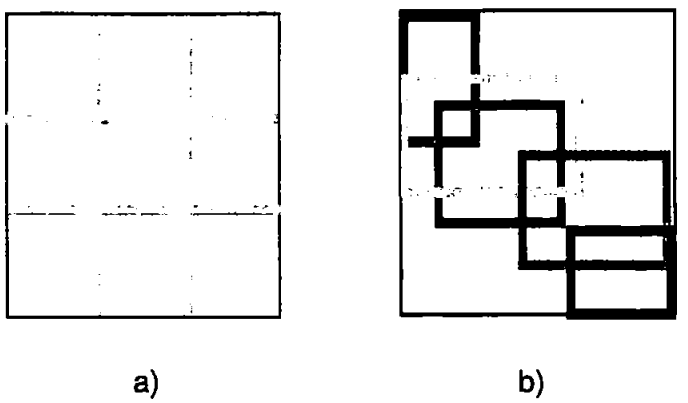


Figure 3.8: Input space partitioning methods.

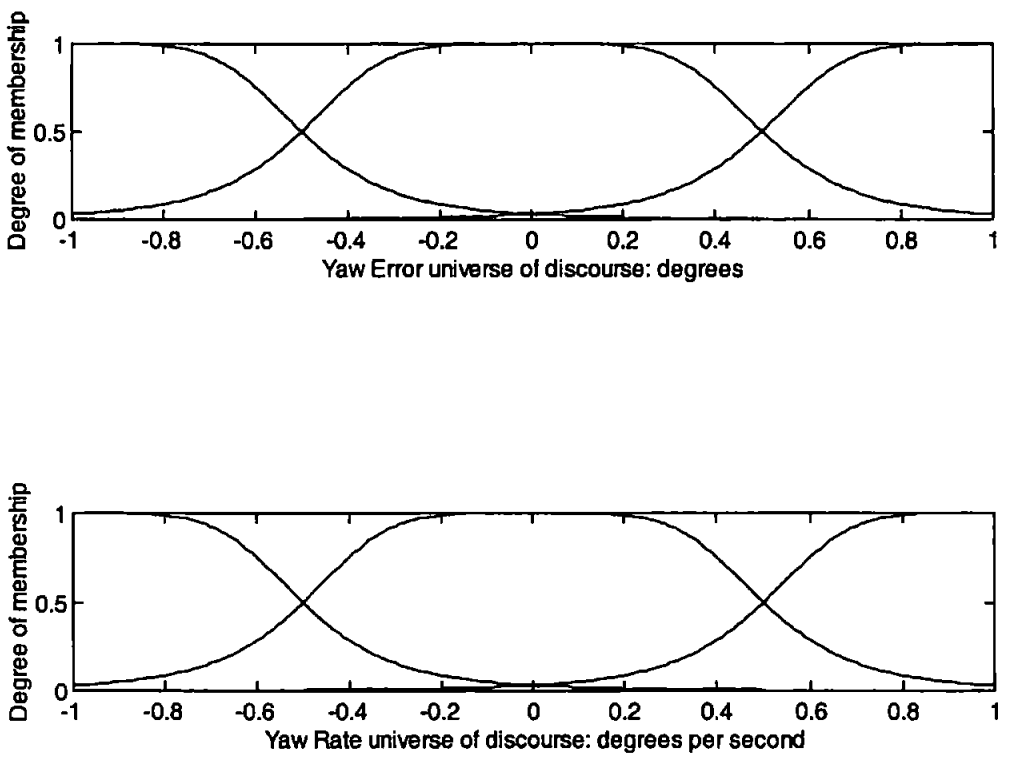


Figure 3.9: The input fuzzy sets for the 9 rule autopilot.

Varying the number of premise membership functions provided some initial insight into how the number of fuzzy rules in each autopilot affects the course-changing

control action of the AUV. The number of membership functions considered on each input universe of discourse were 2, 3, 4 and 5 leading to rulebases of 4, 9, 16 and 25 rules under a grid partitioning approach. The resulting input fuzzy sets for the 9 rule autopilot are depicted in Figure 3.9. The input fuzzy sets for the 4, 16 and 25 rule autopilots are not reproduced here to avoid turgidity.

The developed fuzzy autopilots were applied to the non-linear AUV simulation model using locked actuation of upper and lower canard rudders for course-changing control. The AUV responses to a 40° course-changing demand, at the nominal forward speed of 7.5-knots, are depicted in Figure 3.10.

To ascertain the robustness of each fuzzy autopilot to forward speed, the AUV model was also required to simulate the 40° course change at forward speeds of 5 and 10-knots. The results for the three course-changing simulations are given in Table 3.2, where ψ_ε and δ_ε are defined in Eqn.(3.6) and Eqn.(3.8) respectively.

AUV model	4 rule fuzzy autopilot					9 rule fuzzy autopilot				
	ψ_ε $(^o)^2$	δ_ε $(^o)^2$	T_R sec	$M_p(t)$ %	sse %	ψ_ε $(^o)^2$	δ_ε $(^o)^2$	T_R sec	$M_p(t)$ %	sse %
5 knots	4313.6	2884.0	8.17	0.58	0.00	6898.5	1030.3	23.44	0.26	0.00
7.5 knots	3050.5	1702.0	8.65	0.03	0.00	5039.6	620.95	18.27	0.02	0.00
10 knots	2373.6	2466.4	4.48	-0.25	0.00	4152.7	432.31	16.49	0.00	0.00

AUV model	16 rule fuzzy autopilot					25 rule fuzzy autopilot				
	ψ_ε $(^o)^2$	δ_ε $(^o)^2$	T_R sec	$M_p(t)$ %	sse %	ψ_ε $(^o)^2$	δ_ε $(^o)^2$	T_R sec	$M_p(t)$ %	sse %
5 knots	4347.6	4927.3	6.35	13.18	0.00	5682.4	1399.7	16.14	1.15	0.20
7.5 knots	3045.1	4008.0	4.45	16.22	0.00	4077.8	872.68	12.24	0.69	0.11
10 knots	2580.4	4029.3	3.46	27.50	0.00	3279.9	605.19	10.61	0.50	0.11

Table 3.2: AUV responses to a course change of 40° at 5, 7.5 and 10-knots.

The 4 rule autopilot performs the desired course change quickly with no evidence of steady-state error, at each vehicle speed. However, at 10-knots the result pertaining to

peak overshoot (Table 3.2) displays a value of -0.25%, which illustrates that the AUV failed to reach the desired 40° course-change. Examination of the low canard response pertaining to this course-change illustrates a bang-bang control effect which suggests that either the rule base contains too few rules or that the lack of fuzzy sets centred around zero (on their input universes of discourse) causes oscillatory behaviour. Indeed, examination of the control surface for the four fuzzy autopilots (Figure 3.11) highlights the steep areas of transition between end rules on the 4 rule autopilot surface, which has consequently lead to this bang-bang control effort effect. That is, there are insufficient rules to interpolate the range of possible control effort demands.

The 9 rule autopilot performs the course-change demand effectively with no evidence of steady-state errors and the smallest peak overshoots of all the autopilot results, at all three surge velocities. The low canard rudder response supports the smooth course-changing behaviour, with no saturation (unlike the 4 rule autopilot case), and a maximum incurred angle of approximately14°.

The results pertaining to the rise time are much in excess of those for the 4 rule autopilot but overall performance is superior over this course-change. Examination of the control surface for this autopilot illustrates the smooth interpolation between rules. The speed of response is notably slower arising from the ample coverage of the output domain by the fuzzy rules. This leads to a flatter control surface and thus a less pronounced or slower transition between rules at opposing ends of the rule base.

As in the case of the 4 rule autopilot, the 16 rule results show no evidence of steady-state errors at any speed, although large overshoots are experienced. The low canard rudder response pertaining to the 7.5-knot simulation illustrates the bang-bang control effect as experienced when employing the 4 rule autopilot. As the number of rules has quadrupled with respect to the 4 rule autopilot, and the 9 rule autopilot can be seen to control the AUV successfully, this effect can be purely accredited to the lack of a central fuzzy set on the input universes of discourse.

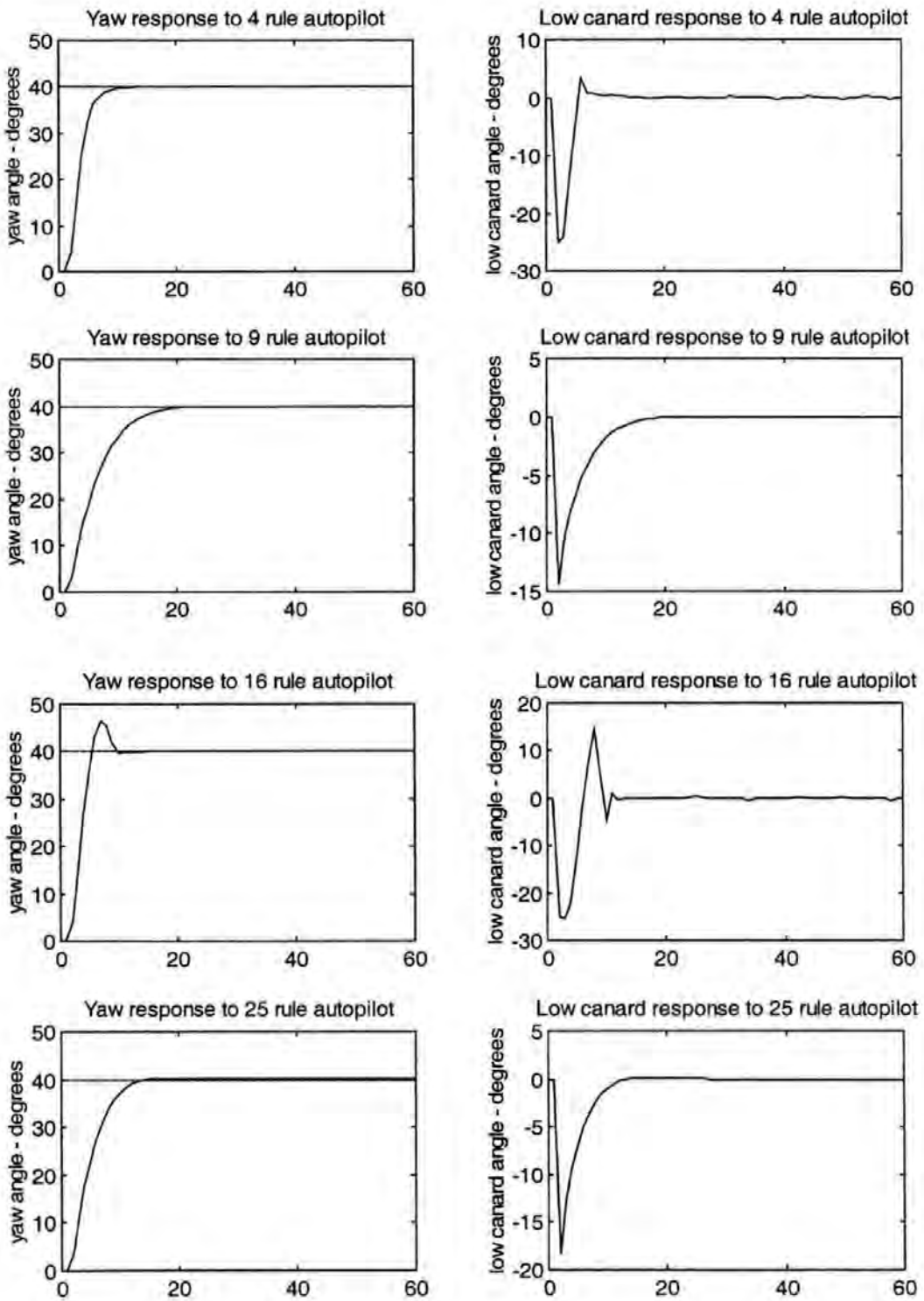


Figure 3.10: AUV yaw and low canard responses for each fuzzy autopilot over a 40 degree course-changing demand at 7.5-knots.

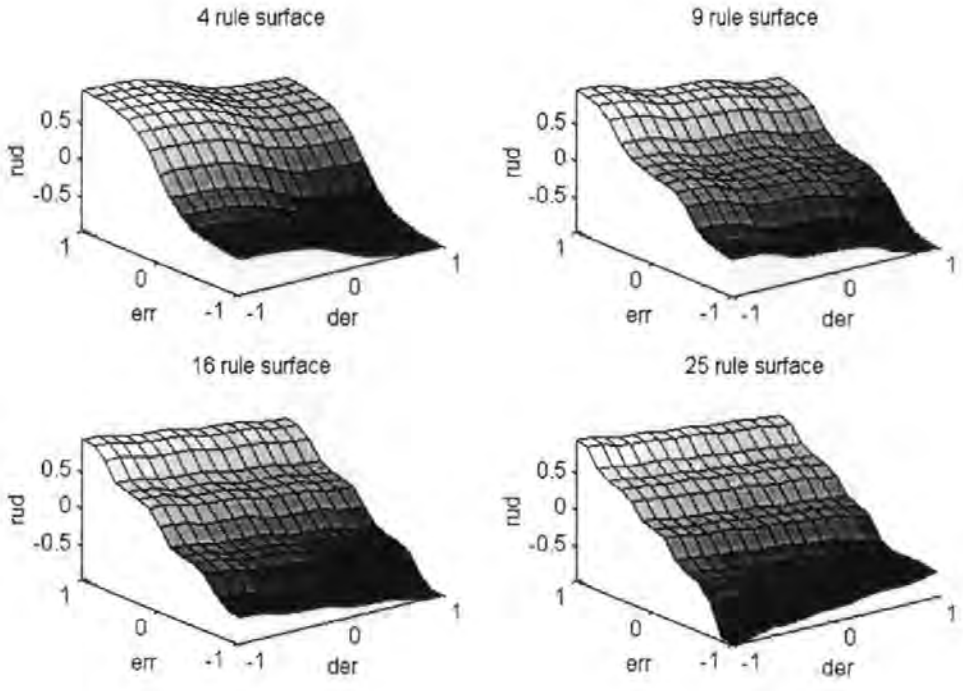


Figure 3.11: Control surfaces for the untuned fuzzy autopilots.

Finally, the 25 rule autopilot shows smooth course-changing characteristics with moderate response times and relatively low results for peak overshoot. However, this autopilot was the only autopilot with evidence of steady-state errors. The reason for these errors was thought to be due to the large number of rules which provide a greater degree of inter-rule interpolation and consequently a diluted amount of canard action in response to off-course errors. Effectively, the coverage of the output domain is too great and each rule does not possess a significant degree of control effort to correct minor steady-state errors. Additionally, the maximum incurred canard demand was greater than that of the 9 rule autopilot at almost 19°. Indeed, the canard effort employed was far greater at all three forward speeds.

To further verify the course-changing ability of each fuzzy autopilot, a course consisting of a number of positive and negative course-changing steps of varying

magnitude was developed. This course includes a 100° course-changing demand in order to examine the properties of each autopilot with respect to yaw induced cross-coupled motion, and also whether the autopilots can control the AUV in the presence of sustained periods of canard saturation. The response of the AUV when employing the 4, 9, 16 and 25 rule TSK fuzzy autopilots over the verification course is depicted in Figure 3.12 for the 7.5-knot nominal AUV surge velocity.

From these initial simulations it can be seen that the 4 and 16 rule autopilots provide poor overall AUV course-changing control displaying some initial oscillatory motion which subsequently leads to unstable behaviour. This inability to follow the verification course accurately is considered due to the lack of a fuzzy set centred on zero on each input universe of discourse. Consequently, a bang-bang control effort effect is produced when the crisp yaw error and rate of change of yaw error input values to these autopilots is close to zero.

Additionally, the 25 rule autopilot was rejected on the basis that the canard control effort was not considerably less than that of the 9 rule autopilot, even though the rule base contained a much greater number of rules. Indeed, the 25 rule fuzzy autopilot also exhibited some steady-state error for the 40° course change.

Hence, based on the results from these preliminary closed loop simulations, the 9 rule TSK style fuzzy autopilot was chosen as the most effective fuzzy autopilot for AUV course-changing control.

The performance of this autopilot was consequently compared to that of the classical phase lead network autopilot design of Eqn.(3.14) under the conditions introduced thus far. The responses of the linear PD style autopilot in comparison to the untuned 9 rule TSK autopilot are reproduced in Figure 3.13 and 3.14 with respect to a 40° course change and the verification track.

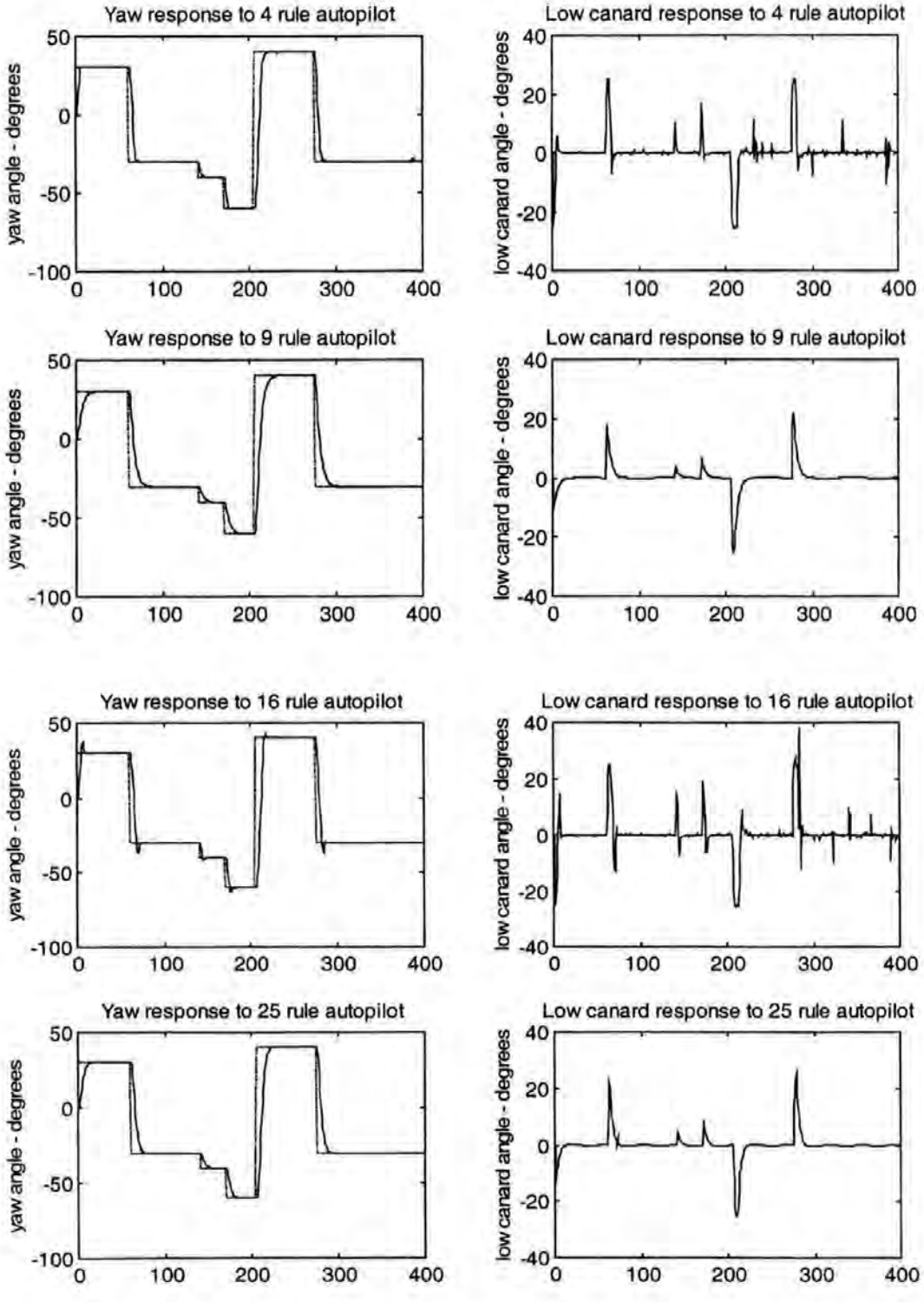


Figure 3.12: AUV yaw and low canard responses for the 4, 9, 16 and 25 rule fuzzy autopilots over the verification course at 7.5-knots.

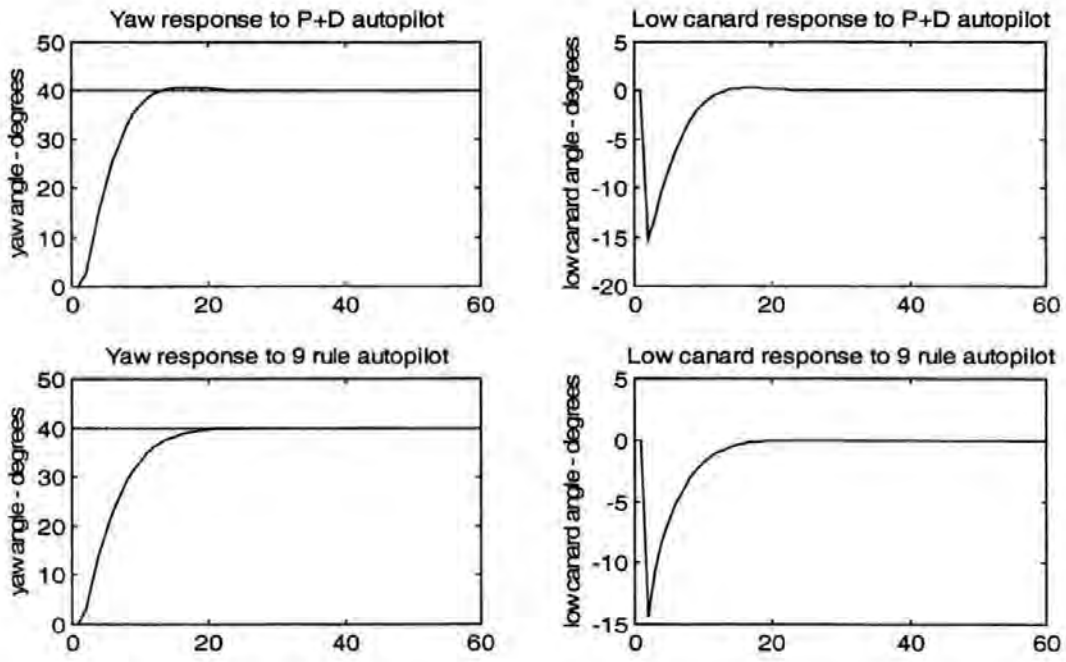


Figure 3.13: AUV yaw and low canard responses for the PD and 9 rule fuzzy autopilots over a 40 degree course-changing demand at 7.5-knots.

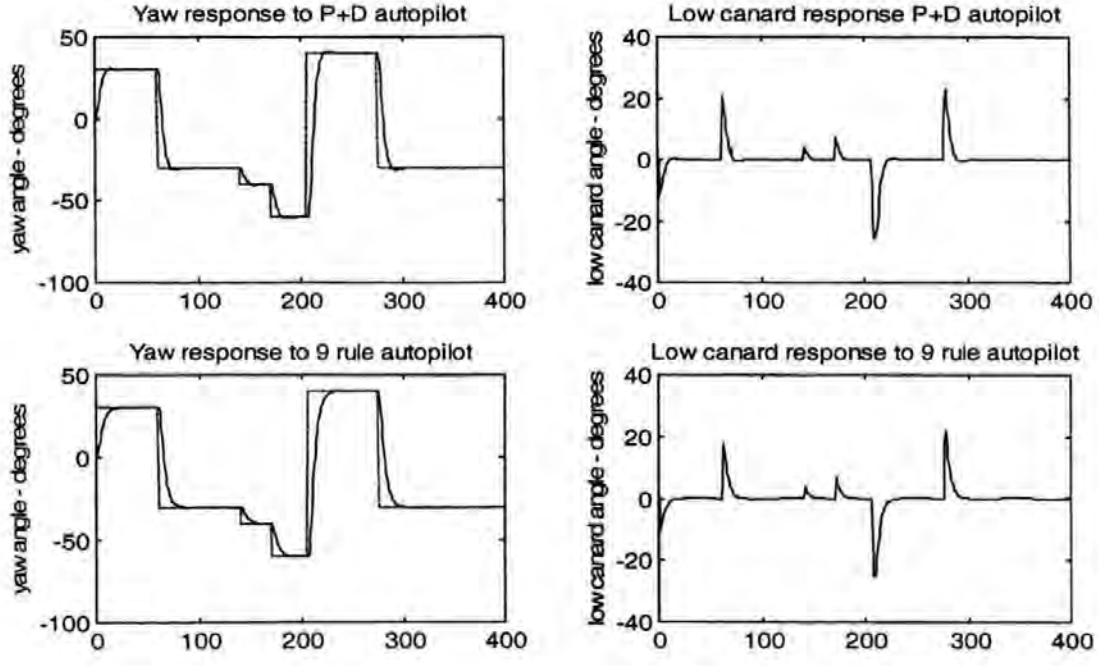


Figure 3.14: AUV yaw and low canard responses for the PD and 9 rule fuzzy autopilots over the verification course at 7.5-knots.

Clearly, the linear compensator performs particularly well in comparison to the untuned fuzzy autopilot under the given conditions.

3.6 Concluding Remarks

Benchmark autopilots have been designed and tested for course-changing control of the AUV model of section 3.2. The classically designed autopilot displays excellent performance. Additionally, the 9 rule TSK fuzzy autopilot proved very effective with respect to the 4, 16 and 25 rule TSK autopilots. However, this autopilot is not tuned at present. Consequently, the course-changing autopilot designs of the following chapter illustrate the application of various tuning regimes to the 9 rule TSK fuzzy autopilot. The resulting autopilots will thus be compared with the traditional and 9 rule TSK fuzzy autopilots.

References

Cowling, D. (1996). Full Range Autopilot Design for an Autonomous Underwater Vehicle. 13th IFAC Triennial World Congress, San Francisco, U.S.A., Vol. Q, pp339-344.

Cowling, D. and Corfield, S. J. (1995). Control Functions for Autonomous Underwater Vehicle On-Board Command and Control Systems. IEE Computing and Control Division Colloquium on Control and Guidance of Remotely Operated Vehicles, Digest No. 1995/124, pp3/1-3/8.

Fjellstad, O.-E. and Fossen, T. I. (1994). Position and Attitude Tracking of AUV's: A Quaternion Feedback Approach. IEEE Journal of Oceanic Engineering, Vol.19, No. 4, pp512-518.

Fossen, T. I. (1994). Guidance and Control of Ocean Vehicles. John Wiley and Sons, Chichester, U.K.

Jang, J.-S. R. and Gulley, N. (1995). Fuzzy Logic Toolbox for use with MATLAB. The Mathworks, Natrick, Massachusetts, U.S.A.

Jang, J.-S. R., Sun C.-T. and Mizutani, E. (1997). Neuro-Fuzzy and Soft Computing. Prentice Hall, New Jersey, U.S.A.

Harris, C. J., Moore, C. G. and Brown, M. (1993). Intelligent Control – Aspects of Fuzzy Logic and Neural Nets. World Scientific Press.

Lee, C. C. (1990). Fuzzy Logic in Control Systems: Fuzzy Logic Controller – Part I. IEEE Transactions on Systems, Man and Cybernetics, Vol. 20, No. 2, pp404-418.

Marshfield, W.B. (1992). Submarine Data Set for use in Autopilot Research. Technical Memorandum, DERA/MAR TM (MTH) 92314, DERA Haslar.

Sutton, R. (1987). Fuzzy Set Models of the Helmsman Steering a Ship in Course-Keeping and Course-Changing Modes. Ph.D Thesis, University of Wales.

Takagi, T. and Sugeno, M. (1985). Fuzzy Identification of Systems and its Applications to Modelling and Control. IEEE Transactions on Systems, Man and Cybernetics, Vol. 15, pp116-132.

Chapter 4

A Neural Network Approach to AUV Fuzzy Autopilot Design

4.1 Introduction

Chapter 2 discussed the inherent ability of fuzzy logic and neural network systems for controlling plant of diverse characteristics. It was noted that the fusion of these techniques yields various methods by which a fuzzy inference system (FIS) can be taught to follow a desired trajectory, based on the choice of suitable training data and learning paradigms. This chapter focuses on the implementation of FISs as single-input single-output (SISO) course-changing autopilots for the AUV model described in Chapter 3. These autopilots are subsequently tuned to improve their performance by adapting the parameters of their structure. The algorithms used to perform this adaptation, namely the hybrid learning rule, the chemotaxis algorithm and the simulated annealing algorithm, are discussed in detail within this chapter prior to an examination

of the results achieved. The work concerning the use of the hybrid learning rule for the tuning of course-changing AUV autopilots is to the best of the present authors knowledge a novel application in the field. Similar studies have been conducted into ship autopilot design (Taylor (1995), Tomera and Morawski (1996)).

4.2 Tuning the 9 Rule Autopilot

By encoding a Takagi-Sugeno-Kang (TSK) (Takagi and Sugeno (1985)) FIS as an adaptive network the parameters of that structure can be tuned to vary the performance of the FIS. This architectural approach to fuzzy membership function adaptation, based on the work of Jang (1992), allows the fuzzy system to learn, and consequently improve its performance when tuned with suitable training data. By comparing the actual output of this network structure with the desired output, an error measure (or cost function) can be formulated which illustrates the learning history of the FIS.

Three algorithms are employed within this chapter to tune the 9 rule TSK fuzzy autopilot of the preceding chapter. These algorithms are discussed in detail in the ensuing sections. The method by which the fuzzy autopilot is encoded as a network architecture is also discussed in depth. The resulting autopilot designs are compared to the pre-tuned 9 rule TSK fuzzy autopilot, and a traditional proportional plus derivative (PD) autopilot, also from the previous chapter, for completeness. Additionally, all autopilot designs are verified by assessing their generalization capabilities. The most suitable autopilots are then tested for their robustness to hydrodynamic coefficient and mass variations as well as forward speed changes.

4.2.1 The Neuro-Fuzzy Autopilot Structure

If it is assumed that the fuzzy inference system under consideration has multiple inputs (x_i) and one functional output (f) then the fuzzy rule-based algorithm may be represented in the first order TSK form as shown below:

Rule 1 : If x_1 is A_1 and x_2 is B_1 and ... and x_n is P_1
then $f_1 = p_1 x_1 + q_1 x_2 + ... + v_1$

Rule 2 : If x_1 is A_2 and x_2 is B_1 and ... and x_n is P_1
then $f_2 = p_2 x_1 + q_2 x_2 + ... + v_2$

. . . . . (4.1)

Rule n : If x_1 is A_n and x_2 is B_n and ... and x_n is P_n
then $f_n = p_n x_1 + q_n x_2 + ... + v_n$

where A_i, B_i,...,P_i are membership functions and p_n, q_n,...,v_n are constants within the consequent functions. The corresponding autopilot architecture is shown in Figure 4.1.

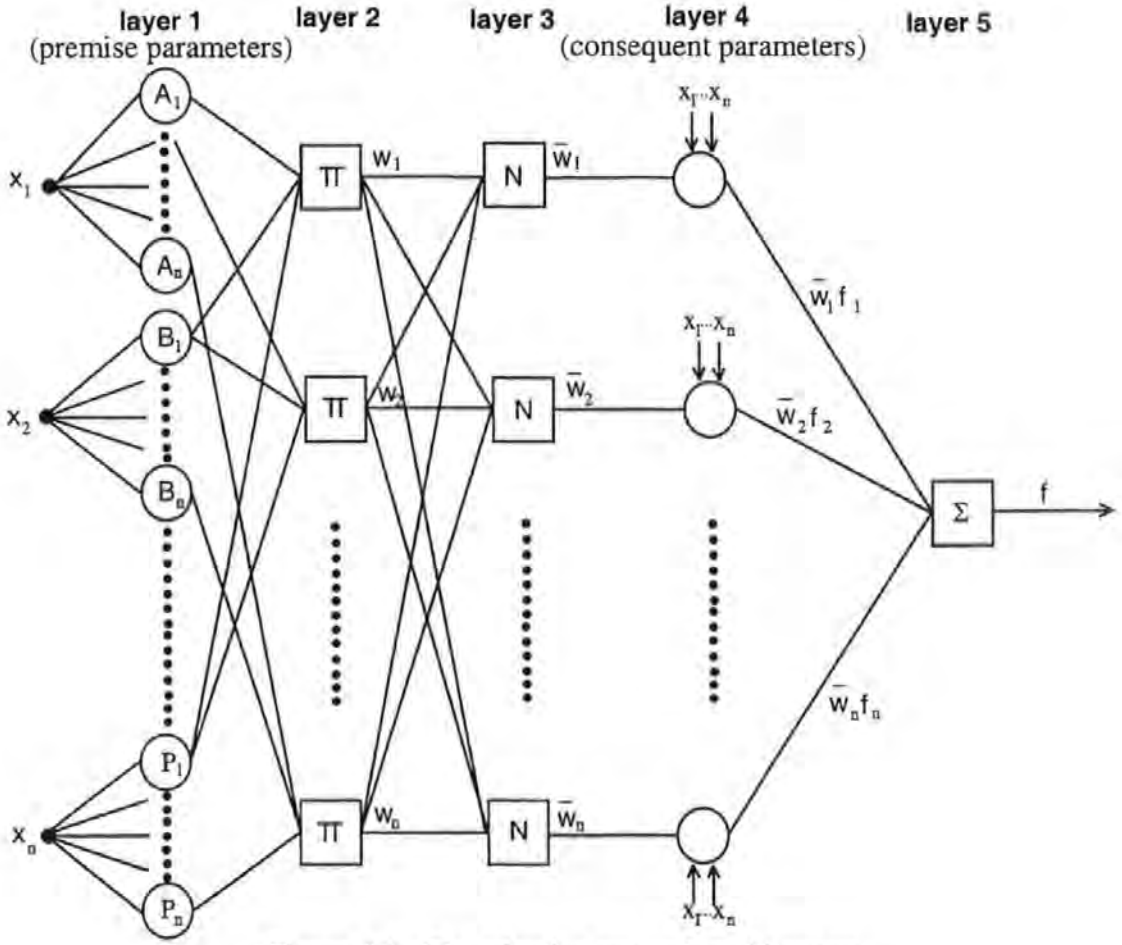


Figure 4.1: The adaptive network architecture.

Node functions within each individual layer of the architecture belong to the same family as follows:

Layer 1: Every node in this layer is an adaptive node with a node output defined by

$$O_{1,i} = \mu_{\gamma_i}(x_i) \tag{4.2}$$

where x_i is the input to the *ith* node and γ_i is the fuzzy set associated with the nodes pertaining to input x_i, that is γ_i can take on the values of the premise membership functions given by $A_i, B_i, \ldots, P_i$. These membership functions can be any appropriate parameterized shapes. Here $A_i, B_i, \ldots, P_i$ are characterized by the generalized bell function

$$\mu_{\gamma_i}(x) = \frac{1}{1+\left[\left(\frac{x - c_i}{a_i}\right)^2\right]^{b_i}} \tag{4.3}$$

where $\{a_i, b_i, c_i\}$ is the parameter set pertaining to that particular node. Parameters in this layer are referred to as *premise parameters*.

Layer 2: Every node in this layer is a fixed node labelled Π, which multiplies the incoming signals and outputs the product or T-norm operator result, e.g.

$$O_{2,i} = w_i = \mu_{A_i}(x_1) \times \mu_{B_i}(x_2) \times \ldots\ldots \times \mu_{P_i}(x_n), \qquad i = 1,2,\ldots,n. \tag{4.4}$$

Each node output represents the *firing strength* of a rule.

Layer 3: Every node in this layer is a fixed node labelled *N*. The *ith* node calculates the ratio of the *ith* rules' firing strength to the sum of all rules' firing strengths

$$O_{3,i} = \bar{w}_i = \frac{w_i}{\sum_i w_i}, \quad i = 1,2,...,n. \tag{4.5}$$

that is, the outputs of nodes within this layer are *normalized firing strengths.*

Layer 4: Each node in this layer is an adaptive node with a node function

$$O_{4,i} = \bar{w}_i f_i = \bar{w}_i (p_i x_1 + q_i x_2 + ... + v_i) \tag{4.6}$$

and $\bar{w}_i$ is the output of layer 3 and $\{p_i, q_i, ..., v_i\}$ is the parameter set pertaining to the *ith* node, v_i being the constant term within the generic linear consequent functions. Parameters in this layer are hereby referred to as *consequent parameters.*

Layer 5: The single node in this layer is labelled $\sum$, which computes the *overall output* as the summation of incoming signals

$$O_{5,i} = \textit{overall output} = \sum_i \bar{w}_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \tag{4.7}$$

Thus, an adaptive network architecture that performs exactly the same function as a single output TSK FIS may be constructed.

4.2.2 Tuning Algorithms

In order to examine the influence of the premise parameters on the approximation power of the developed FIS, two strategies were adopted for parameter tuning. Firstly,

the premise parameters were adapted whilst the consequents remained fixed. Alternatively, tuning of both the premise and consequent parameters took place simultaneously. Both experiments employed the hybrid learning rule, the chemotaxis and simulated annealing algorithms.

4.2.2.1 The Hybrid Learning Rule

This section discusses the hybrid learning rule of Jang (1992) which fuses two algorithms to systematically adapt the membership functions of the premise and the consequent portions simultaneously. With regard to the tuning architecture of Figure 4.1, in the forward pass data is presented to the network structure and is propagated forward until layer 4 whereby the sequential least squares algorithm is employed to assign the parameters of the consequent terms. Next, the data is fed through to the output of the network structure and the cost function is evaluated based on the newly assigned consequent function parameters. Finally, the rate of change of the cost function with respect to each nodes output is back-propagated through the network in order to tune the premise membership functions with gradient descent.

When tuning the premise parameters, the backpropagation element of the hybrid learning rule can be solely employed. Conversely, by employing the complete hybrid rule, the cost function obtained after the first forward pass (0.5 epochs) is somewhat reduced due to the ability of this technique to generate an initial estimate of the consequent portions of the fuzzy rules.

Suppose that a given adaptive network has L layers and the k_{th} layer has $\#(k)$ nodes. The node in the i_{th} position of the k_{th} layer can thus be denoted by (k,i), and its node function (or node output) by O_i^k . Since a node output depends upon its incoming signals and its parameter set

$$O_i^k = O_i^k\left(O_1^{k-1},\ldots,O_{\#(k-1)}^{k-1},a,b,c,\ldots\right) \tag{4.8}$$

where $a,b,c,...$etc are the parameters pertaining to that node. Assuming the given training data has P entries, the error measure (or cost function) for the p_{th} $(1 \le p \le P)$entry of training data can be defined as the sum of squared error

$$E_p = \sum_{m=1}^{\#(L)} \left(T_{m,p} - O_{m,p}^{L} \right)^2 \tag{4.9}$$

where $T_{m,p}$ is the m_{th} component of the p_{th} target output vector, and $O_{m,p}^{L}$ is the m_{th} component of the adaptive networks output vector, produced by the presentation of the p_{th} input vector. Hence the overall error measure (for all P training vectors) is

$$E = \sum_{p=1}^{P} E_p = \sum_{p=1}^{P} \sum_{m=1}^{\#(L)} \left(T_{m,p} - O_{m,p}^{L} \right)^2 . \tag{4.10}$$

In order to develop a learning procedure that implements gradient descent in E over the parameter space, first the error rate $\frac{\partial E_p}{\partial O}$ must be calculated for the p_{th} training data and for each node output O. The error rate for the output node at (L,i) where $(1 \le i \le \#(L))$ can be calculated readily from Eqn.(4.9)

$$\frac{\partial E_p}{\partial O_{i,p}^{L}} = -2 \sum_{p=1}^{P} \left(T_{m,p} - O_{m,p}^{L} \right) \tag{4.11}$$

For the internal node at (k,i), the error rate can be derived by the chain rule:

$$\frac{\partial E_p}{\partial O_{i,p}^{k}} = \sum_{m=1}^{\#(k+1)} \frac{\partial E_p}{\partial O_{m,p}^{k+1}} \cdot \frac{\partial O_{m,p}^{k+1}}{\partial O_{i,p}^{k}} \tag{4.12}$$

where $(1 \le k \le L-1)$. That is, the error rate of an internal node can be expressed as a linear combination of the error rates of the nodes in the following layer. Therefore for all $(1 \le k \le L)$ and $(1 \le i \le \#(k))$, $\frac{\partial E_p}{\partial O_{i,p}^k}$ can be found using Eqn.(4.11) and Eqn.(4.12).

Now if α is a parameter of the given adaptive network

$$\frac{\partial E_p}{\partial \alpha} = \sum_{O^* \in S} \frac{\partial E_p}{\partial O^*} \cdot \frac{\partial O^*}{\partial \alpha} \tag{4.13}$$

where S is the set of nodes whose outputs depend on α. Then the derivative of the overall error measure E with respect to α is

$$\frac{\partial E}{\partial \alpha} = \sum_{p=1}^{P} \frac{\partial E_p}{\partial \alpha}. \tag{4.14}$$

Accordingly, the update formula for the generic parameter α is

$$\Delta \alpha = -\eta \frac{\partial E}{\partial \alpha} \tag{4.15}$$

in which η is a learning rate, expressed as

$$\eta = \frac{\kappa}{\sqrt{\sum_{\alpha} \left(\frac{\partial E}{\partial \alpha} \right)^2}} \tag{4.16}$$

and κ is the step size, the length of each gradient transition in the parameter space.

Writing the premise membership functions as generalized bells:

$$\mu_{ij}(x) = \frac{1}{1+\left[\left(\frac{x-c_{ij}}{a_{ij}}\right)^2\right]^{b_{ij}}} \tag{4.17}$$

then Eqn.(4.17) represents the j_{th} membership function on the i_{th} input universe of discourse, where a_{ij} governs the width of the bell function, b_{ij} governs the flatness of the bell function and c_{ij} is the centre of the bell function. Thus the change in the general parameter α can be re-written more specifically as

$$\Delta\alpha_{ij} = -\eta.\sum_{p=1}^{P}\frac{\partial E_p}{\partial O^1_{m,p}}.\frac{\partial O^1_{m,p}}{\partial \alpha_{ij}}$$

$$= -\eta.\sum_{p=1}^{P}\frac{\partial E_p}{\partial O^2_{m,p}}.\frac{\partial O^2_{m,p}}{\partial O^1_{m,p}}.\frac{\partial O^1_{m,p}}{\partial \alpha_{ij}}. \tag{4.18}$$

If the function $O^1 = f(\alpha_{ij})$ is differentiable then $\frac{\partial O^1_{m,p}}{\partial \alpha_{ij}}$ is a straightforward calculation. Hence the motivation for choosing the set functions described by Eqn.(4.17). Considering the AUV model as the final layer in the adaptive network yields

$$\frac{\partial E_p}{\partial O^5_{m,p}} = \frac{\partial}{O^5_{m,p}}\left[\sum_{p=1}^{P}\left(T_{m,p} - O^5_{m,p}\right)^2\right] \tag{4.19}$$

$$= \sum_{p=1}^{P} -2\left(T_{m,p} - O^5_{m,p}\right) \tag{4.20}$$

There are no adaptable parameters in the AUV model layer. The next layer, layer 4, is the one that produces the defuzzified output. The computation of $\frac{\partial E_p}{\partial O^4_{m,p}}$ uses a backpropagation of $\frac{\partial E_p}{\partial O^5_{m,p}}$:

$$\frac{\partial E_p}{\partial O^4_{m,p}} = \sum_{m=1}^{\#(5)} \frac{\partial E_p}{\partial O^5_{m,p}} \cdot \frac{\partial O^5_{m,p}}{\partial O^4_{m,p}} \tag{4.21}$$

Now $\frac{\partial O^5_{m,p}}{\partial O^4_{m,p}}$ may be written as

$$\frac{\partial O^5_{m,p}}{\partial O^4_{m,p}} = \frac{\partial y_{plant}}{\partial \delta_{control}} \tag{4.22}$$

where the function relating each plant output y_{plant} to each control input $\delta_{control}$ is non-linear and the derivative function (*Jacobian matrix*) may be approximated by

$$\frac{\partial O^5_{m,p}}{\partial O^4_{m,p}} = \frac{O^5(n) - O^5(n-1)}{O^4(n) - O^4(n-1)}. \tag{4.23}$$

where n is a chosen level of discretization. The only layer to be adapted using the backpropagation method is the first layer. Hence, continuing the above process for each layer leads to the following learning rules for each individual parameter within layer 1:

$$\Delta a_{ij} = -\eta \sum_{p=1}^{P} \frac{\partial E_p}{\partial O^2_{m,p}} \cdot \frac{\partial O^2_{m,p}}{\partial O^1_{m,p}} \cdot \left[\frac{2b_{ij} \cdot a_{ij}^{(2b_{ij}-1)} \cdot \left(x - c_{ij}\right)^{2b_{ij}} \cdot sign\left(a_{ij}\right)^{2b_{ij}} \cdot sign\left(x - c_{ij}\right)^{2b_{ij}}}{\left\{ a_{ij}^{2b_{ij}} \cdot sign\left(x - c_{ij}\right)^{2b_{ij}} + \left(x - c_{ij}\right)^{2b_{ij}} \cdot sign\left(a_{ij}\right)^{2b_{ij}} \right\}^2} \right] \tag{4.24}$$

$$\Delta b_{ij} = -\eta \sum_{p=1}^{P} \frac{\partial E_p}{\partial O^2_{m,p}} \cdot \frac{\partial O^2_{m,p}}{\partial O^1_{m,p}} \cdot \left[\frac{2a_{ij}^{\,2b_{ij}} \cdot \left(x - c_{ij}\right)^{2b_{ij}} \cdot sign\left(a_{ij}\right)^{2b_{ij}} \cdot sign\left(x - c_{ij}\right)^{2b_{ij}} \cdot \ln\left[\left|\frac{a_{ij}}{x - c_{ij}}\right|\right]}{\left\{a_{ij}^{\,2b_{ij}} \cdot sign\left(x - c_{ij}\right)^{2b_{ij}} + \left(x - c_{ij}\right)^{2b_{ij}} \cdot sign\left(a_{ij}\right)^{2b_{ij}}\right\}^2} \right] \quad (4.25)$$

$$\Delta c_{ij} = -\eta \sum_{p=1}^{P} \frac{\partial E_p}{\partial O^2_{m,p}} \cdot \frac{\partial O^2_{m,p}}{\partial O^1_{m,p}} \cdot \left[\frac{-2b_{ij} \cdot a_{ij}^{\,2b_{ij}} \cdot \left(c_{ij} - x\right)^{(2b_{ij}-1)} \cdot sign\left(a_{ij}\right)^{2b_{ij}} \cdot \left(-sign\left(x - c_{ij}\right)\right)^{2b_{ij}}}{\left\{a_{ij}^{\,2b_{ij}} \cdot \left(-sign\left(x - c_{ij}\right)\right)^{2b_{ij}} + \left(c_{ij} - x\right)^{2b_{ij}} \cdot sign\left(a_{ij}\right)^{2b_{ij}}\right\}^2} \right] \quad (4.26)$$

In order to complete the derivation of the hybrid learning algorithm all that remains is to discuss the tuning paradigm for the consequents. Generalizing such that the adaptive network under consideration has multiple outputs in layer L, consider

$$\underline{output} = F\left(\vec{I}, S\right) \quad (4.27)$$

where $\vec{I}$ is the set of input variables, S is the full set of adaptive network parameters and $\underline{output}$ is a column vector of network outputs. If there exists a function H such that the composite function $H \circ F$ is linear in some of the elements of S, then these elements can be identified by the least squares method. More formally, if the parameter set can be decomposed into two sets

$$S = S_1 \oplus S_2 \quad (4.28)$$

(where $\oplus$ represents direct sum) such that $H \circ F$ is linear in the elements of S_2, then applying H to Eqn.(4.27) yields

$$H(\underline{output}) = H \circ F\left(\vec{\underline{I}}, S\right) \tag{4.29}$$

which is linear in the elements of S_2. Now given values for the parameters S_1, P training data can be plugged into Eqn.(4.29) to obtain a matrix equation

$$A\underline{x} = \underline{b} \tag{4.30}$$

where $\underline{x}$ is an unknown vector whose elements are parameters in S_2. Let $|S_2| = M$, then the dimensions of A, $\underline{x}$ and $\underline{b}$ are $P \times M$, $M \times 1$ *and* $P \times 1$ respectively. Since P is usually greater than M (number of linear parameters), this is an over-determined problem and generally there is no exact solution to Eqn.(4.30). Instead, a recursive least squares estimate (RLSE) of $\underline{x}$, $\underline{x}^*$ is sought to minimize the squared error $\|A\underline{x} - b\|^2$, (or $\|\underline{e}\|^2$ where $\underline{e} = \underline{b} - A\underline{x}$). The most well known formula for $\underline{x}^*$ is calculated as follows. To minimize $\|\underline{e}\|^2$ consider the following:

$$\nabla_{x^*}\left(\|\underline{e}\|^2\right) = 0$$

$$\nabla_{x^*}\left(\left\|\underline{b} - A\underline{x}^*\right\|^2\right) = 0$$

$$0 - A^T\underline{b} - A^T\underline{b} + A^TA\underline{x}^* + A^TA\underline{x}^* = 0$$

$$(A^TA)^{-1}A^T\underline{b} = (A^TA)^{-1}A^TA\underline{x}^*$$

$$\underline{x}^* = \left(A^TA\right)^{-1}A^T\underline{b} \tag{4.31}$$

where $(A^T A)^{-1} A^T$ is the pseudo-inverse of A *if* $A^T A$ is non-singular. While Eqn.(4.31) is concise in notation it is expensive in computation when dealing with the matrix inverse and, moreover it becomes ill defined if $A^T A$ is singular. As a result the sequential least squares formulae are employed to compute the LSE of $\underline{x}^*$. This method is more efficient, especially when the number of linear parameters M is small.

Specifically, let the i_{th} row vector of matrix A defined in Eqn.(4.31) be a_i^T and the i_{th} row vector of matrix $\underline{b}$ defined in Eqn.(4.31) be b_i^T, then $\underline{x}^*$ can be calculated iteratively using the sequential least squares formulae, where S_i is the covariance matrix:

$$\left.\begin{aligned} x_{i+1} &= x_i + S_{i+1} a_{i+1} \left(b_{i+1}^T - a_{i+1}^T x_i\right) \\ S_{i+1} &= S_i - \frac{S_i a_{i+1} a_{i+1}^T S_i}{1 + a_{i+1}^T S_i a_{i+1}}, i = 0,1,...k-1. \end{aligned}\right\} \tag{4.32}$$

The sequential least squares estimator for time-invariant systems of equations with multiple outputs can be obtained almost identically as:

$$\begin{cases} S_{k+1} = S_k - \dfrac{S_k \underline{a}_{k+1} \underline{a}_{k+1}^T S_k}{1 + \underline{a}_{k+1}^T S_k \underline{a}_{k+1}} \\ x_{k+1} = x_k + S_{k+1} \underline{a}_{k+1} \left(\underline{b}_{k+1}^T - \underline{a}_{k+1}^T x_k\right) \end{cases} \tag{4.33}$$

Thus the hybrid learning rule is formulated. Training data for this approach were collected by simulating the dynamic model in the open loop. Essentially, a pseudo random binary sequence (PRBS) signal was presented to the open loop dynamics to ensure that training data were taken from a wide range of the AUV yaw dynamic

operational envelope. Yaw error, yaw rate and canard demands vector pairs were collected at every 0.1 seconds of the resulting trajectories, and pre-processed prior to tuning with each algorithm.

An example of the suitability of this approach for the modelling of a non-linear function is provided by Jang (1992), for the interested reader. Also various other experiments with this method are discussed in Jang et *al.* (1997).

4.2.2.2 The Chemotaxis Algorithm

The main disadvantage of gradient descent methods, such as the backpropagation algorithm, is the tendency for the search to fall into a local minimum on the error hyper-surface. The more complex the network the more likely this is to happen as this hyper-surface is increasingly multi-dimensional and therefore irregular, with more local minima in which the partially trained network can become trapped. An alternative is to use less guided methods to search the parameter space. Such random methods are virtually guaranteed to find a global solution but training times are typically much longer than gradient methods as there is little direction in the search.

The chemotaxis algorithm [Koshland (1980)] was inspired by observations of the movement of bacteria in a chemical environment, hence 'chemo' - chemical, and 'taxis' - movement. In the presence of an irritant, bacteria would move randomly away in any direction in order to reduce the irritation, until this direction took them into an area where the irritation would again increase. A new random direction would then be chosen and if this again led to less irritation the bacteria would head in this new direction, otherwise another random direction would be tried. In time the bacteria would be located at the global minima, that is, furthest from the source of irritation. This behaviour may be transformed into a general search algorithm for an optimum set of neural network weights or parameters. The increase or decrease in irritation may be characterized by an increase or decrease in a suitable cost function during the

optimization. By converting this better/worse information into a reinforcement signal according to:

$$\begin{aligned} r(t) &= 1, \text{ better} \\ r(t) &= 0, \text{ worse} \end{aligned} \tag{4.34}$$

the chemotaxis search algorithm is obtained, which may be classed as a reinforcement learning technique. The algorithm is summarized in Table 4.1.

1.	Simulate the system with an initial set of parameters.
2.	Generate some small random changes in the parameters and re-simulate the system.
3.	If the system's performance has improved with the new set of parameters, retain the changes and re-apply. This assumes that the local cost function gradient will continue to be negative in the local area.
4.	If the system performance has worsened, return to step 2.
5.	Continue until the system has reached an acceptable level.

Table 4.1: The chemotaxis algorithm.

Given sufficient training time, the algorithm should converge to a global minimum of the cost function, although given the random nature of the search an extended training period is often necessary. During the search process, input data are used to generate an error function. The chemotaxis algorithm is then applied to search for a set of optimal autopilot parameters.

4.2.2.3 The Simulated Annealing Algorithm

Another popular technique which is often used to overcome the shortcomings of gradient descent based approaches is that of simulated annealing which is based upon an analogy of a certain physical system and was first employed by Kirkpatrick et *al.*

(1983). Annealing is the process whereby a substance is initially melted at an extremely high temperature and then gradually cooled to a desired form, such as a crystal structure of a solid.

Kirkpatrick et *al.* (1983) adapted an algorithm taken from the statistical mechanics field for converging to one of many possible cooled or low energy states. Energies of this algorithm are described by a Boltzman probability distribution as shown in Figure 4.2. Clearly, the probability of any given energy E, is an exponentially decreasing function of E.

Thus if a new matrix of parameters θ, (which have been perturbed from an initially assumed solution by a randomly generated amount), leads to an improved performance of the system under consideration, then they are accepted. The process is then repeated. However, if this new matrix leads to a worsened performance of the system the new parameters may occasionally be accepted with probability $P(\theta)$ such that:

$$P(\theta)=\exp\left[\frac{-E(\theta)}{kT}\right] \tag{4.35}$$

where $E(\theta)$ is the energy associated with the state θ, k is a constant and T is a temperature parameter which decays training according to:

$$T=\frac{T_0}{1+an}. \tag{4.36}$$

T_0 is the initial temperature, a is a constant which governs the rate of decay and n is the training epoch. By including this probability function (Eqn.(4.35)) the system is allowed to escape the local minima of the error hyper-surface.

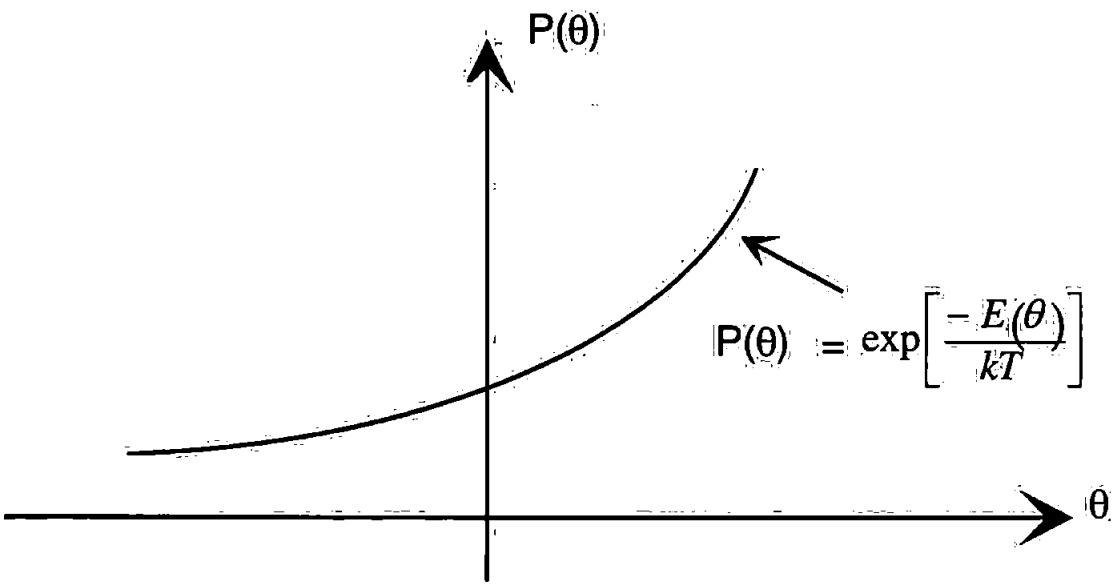


Figure 4.2: The Boltzman probability distribution.

Simulated annealing consists of three distinct steps: a random search step, a minimization step and a stopping rule. The random search step is given by the iterative generation of random matrices in a domain $S(\theta_k)$, constituted by neighbouring matrices associated to the current matrix θ_k, given by:

$$\theta_k = \begin{bmatrix} \theta_k^{11} & \cdots & \theta_k^{1n} \\ \vdots & \ddots & \vdots \\ \theta_k^{m1} & \cdots & \theta_k^{mn} \end{bmatrix}, \qquad \theta_k \in \Re^n \tag{4.37}$$

The intermediary step consists of applying a local minimization routine to the sampled matrices in order to assess their suitability as system parameter sets matrices. Convergence towards the global minimum of the error hypersurface is possible given suitable training conditions.

Finally, the stopping rule is used to stop the algorithm if there is sufficient evidence that the global minimum has been detected within the limits of a specified accuracy or when some pre-specified iteration number is reached. The simulated annealing algorithm is summarized in Table 4.2.

1. Simulate the dynamic system with an initial set of parameters.
2. Generate some small random changes in the parameters and re-simulate the system.
3. If the system's performance has improved with the new set of parameters, retain the changes and re-apply.
4. If the system performance has degraded, compute the probability of accepting the poorer parameters.
5. Generate random number in the range 0-1 and compare to probability computed at 4. If random number is less than probability, then accept poorer parameters; otherwise reject.
6. Re-simulate and return to 3 until convergence.

Table 4.2: The simulated annealing algorithm.

4.3 Results and Discussion

Prior to application of the tuning algorithms, each universe of discourse for the 9 rule TSK fuzzy autopilot was normalized to the range [-1, 1]. Clearly, these ranges needed to be re-scaled upon completion of autopilot tuning as appropriate. This method was considered effective in ensuring that the magnitude of parameter perturbations be equally effectual over each range. The initial input fuzzy sets are reproduced in Figure 4.3. The output TSK functions were again chosen as fuzzy singletons (as in Chapter 3) equally distributed over the normalized output universe of discourse. Consequently, the initial fuzzy rulebase was taken as Eqn.(4.38):

$$
\begin{aligned}
&\text{If } \psi_\varepsilon \text{ is N and } \dot{\psi} \text{ is N then } \delta = +1.00\\
&\text{If } \psi_\varepsilon \text{ is N and } \dot{\psi} \text{ is Z then } \delta = +0.75\\
&\text{If } \psi_\varepsilon \text{ is N and } \dot{\psi} \text{ is P then } \delta \ +0.50\\
&\text{If } \psi_\varepsilon \text{ is Z and } \dot{\psi} \text{ is N then } \delta = +0.25\\
&\text{If } \psi_\varepsilon \text{ is Z and } \dot{\psi} \text{ is Z then } \delta = 0.000\\
&\text{If } \psi_\varepsilon \text{ is Z and } \dot{\psi} \text{ is P then } \delta = -0.25\\
&\text{If } \psi_\varepsilon \text{ is P and } \dot{\psi} \text{ is N then } \delta = -0.50\\
&\text{If } \psi_\varepsilon \text{ is P and } \dot{\psi} \text{ is Z then } \delta = -0.75\\
&\text{If } \psi_\varepsilon \text{ is P and } \dot{\psi} \text{ is P then } \delta = -1.00
\end{aligned}
\qquad (4.38)
$$

Clearly, no a$^{/}$ *priori* knowledge is encompassed within the initial FIS. Thus direct comparisons between autopilots can be made within the ensuing sections.

4.3.1 Initial Tuning Experiments

These initial experiments were designed to provide a measure of the effects of the learning rate on the input fuzzy set adaptation, prior to the addition of consequent

parameter tuning. Additionally, the approximation power of the resulting fuzzy model with respect to the non-linear premise parameters could be examined.

The backpropagation, chemotaxis and simulated annealing algorithms were applied to the task of tuning the parameters of the premise membership functions of the 9 rule TSK autopilot. The nominal surge velocity of the AUV during each learning regime was 7.5-knots and the period of tuning was initially designated as 300 epochs. Forward propagation of the autopilot parameters is required to formulate the first cost function value leading to an extra half an epoch of training. Backpropagation tuning of these parameters took place using a set of suitable training data, which was considered varied enough to provide good autopilot generalization results after tuning. Because the backpropagation algorithm employs the gradient in the local space, the search is a local one. Thus the aim of tuning is to capture the local or regional information inherent within the training data. Conversely, the random search techniques explore a wider area of the parameter space (unless the learning rate is sufficiently small) and thus elicit the global optimum parameter set to be found given sufficient training time.

The backpropagation rule elicits use of a cross validation data set whilst adapting the fuzzy sets off-line. Thus, error training history curves for the backpropagation tuned autopilot include the cost function obtained by also applying the cross validation data set. This history curve gives a measure of the generalization of the tuned FIS. Figure 4.4 shows the training and checking error histories during the initial backpropagation training of 300.5 epochs. The cost functions obtained when employing the chemotaxis and simulated annealing algorithms are shown in Figures 4.5 and 4.6.

The input fuzzy sets resulting from the tuning regime (for each autopilot) were taken as shown in Figure 4.7 whilst the consequent functions remained as equally spaced fuzzy singletons. The non-symmetrical nature of the backpropagation tuned input fuzzy sets over the original fuzzy sets of Figure 4.3 was considered partly due to computer truncation errors arising during the training process.

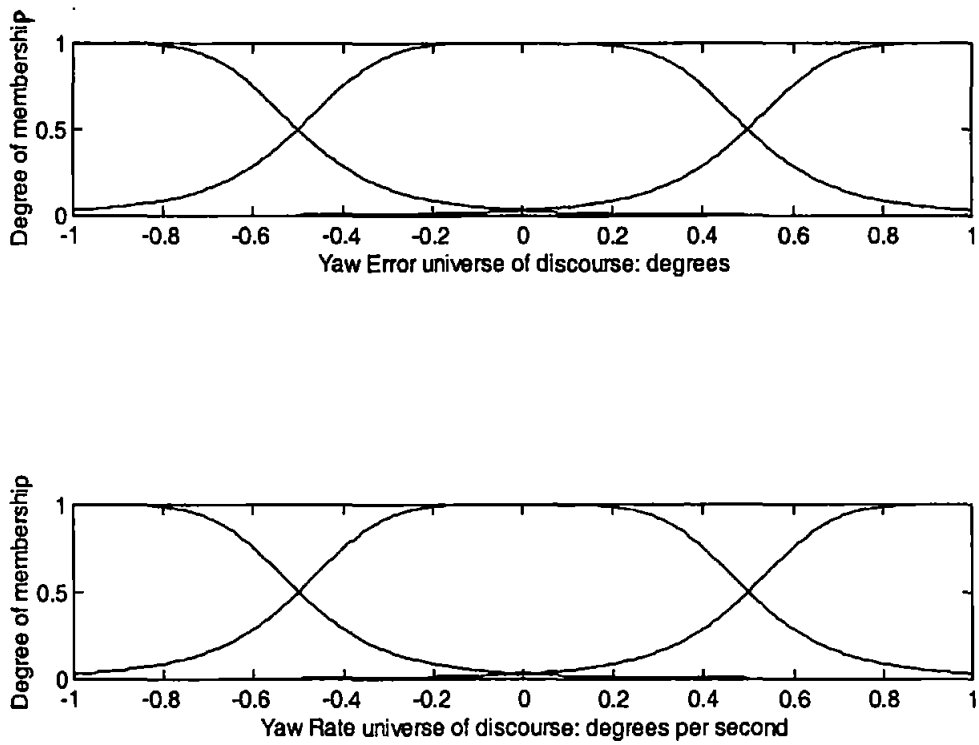


Figure 4.3: The input fuzzy sets defined on the interval [-1,1] for the pre-tuned 9 rule TSK style autopilot.

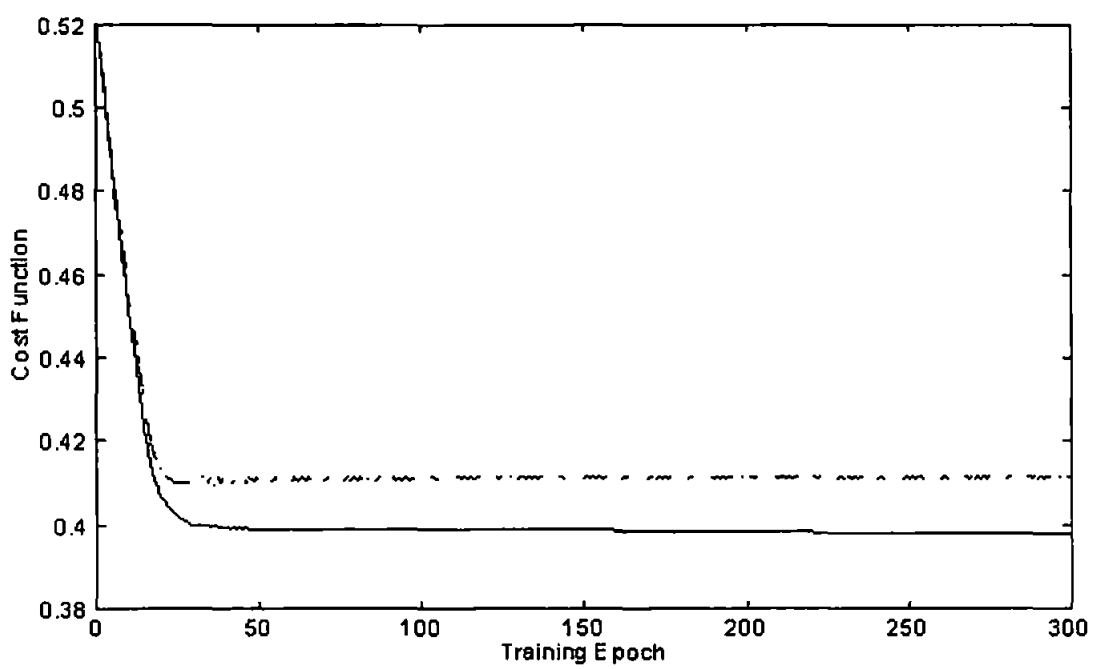


Figure 4.4: Training and checking error histories for the backpropagation algorithm. Solid and dashed-dot lines represent training and checking error curves respectively.

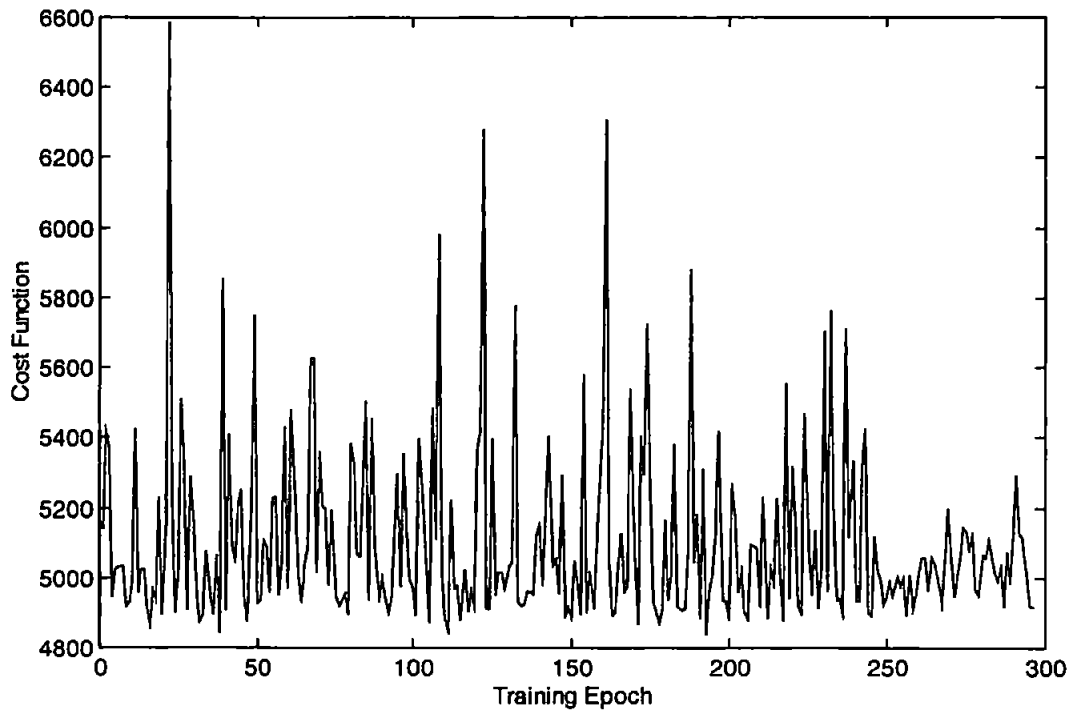


Figure 4.5: The training error history for the chemotaxis search algorithm.

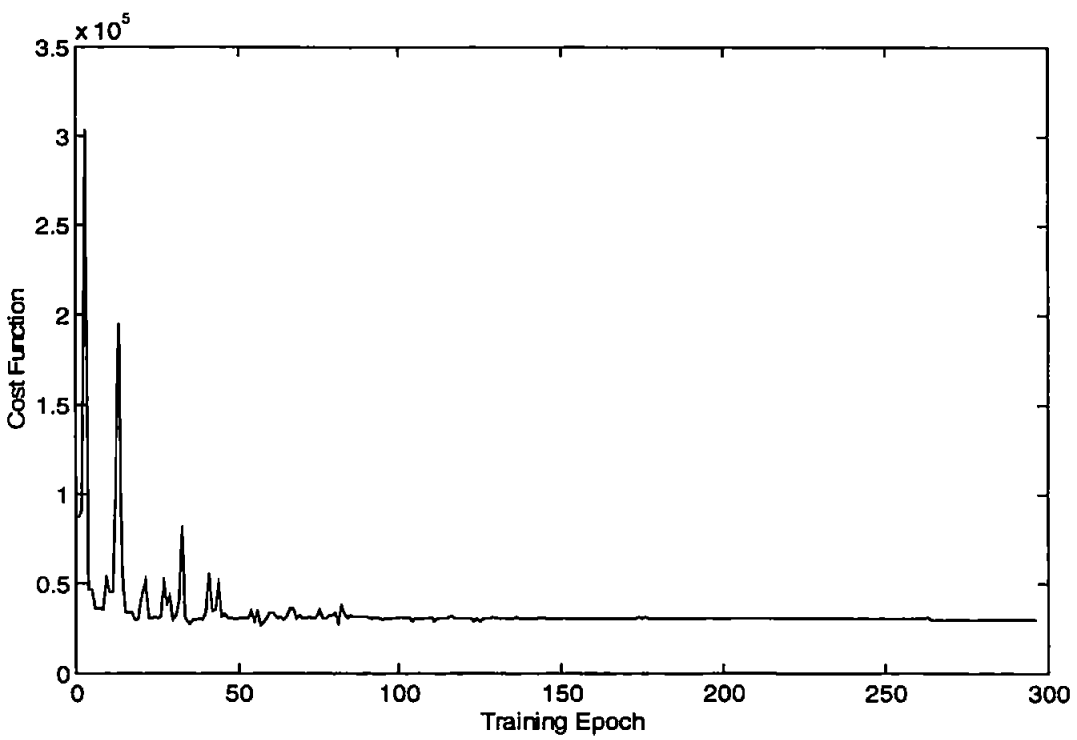


Figure 4.6: The cost function history during simulated annealing learning.

The AUV was applied to a 40° course-changing task at 5, 7.5 and 10-knots whilst employing each autopilot. This provided an assessment of the performance of the resulting autopilots in terms of course-changing ability and robustness to variation in forward speed. The resulting yaw and low canard rudder responses for the nominal 7.5-knot simulations are reproduced in Figure 4.8. The results pertaining to all three forward speeds are given in Table 4.3 below.

AUV model	Backpropagation Tuning					Chemotaxis Tuning				
	ψ_s $(\circ)^2$	δ_e $(\circ)^2$	T_R sec	$M_p(t)$ %	sse %	ψ_s $(\circ)^2$	δ_e $(\circ)^2$	T_R sec	$M_p(t)$ %	sse %
5 knots	4727.8	4606.2	7.14	24.37	2.88	6908.3	1028.0	23.61	0.24	0.01
7.5 knots	4166.3	6182.1	5.21	39.19	2.88	5060.3	617.5	19.55	0.03	0.01
10 knots	5383.6	1030.3	4.33	50.01	3.03	4170.3	426.8	17.72	0.01	0.01

AUV model	Simulated Annealing Tuning					Untuned Sugeno fuzzy autopilot				
	ψ_s $(\circ)^2$	δ_e $(\circ)^2$	T_R sec	$M_p(t)$ %	sse %	ψ_s $(\circ)^2$	δ_e $(\circ)^2$	T_R sec	$M_p(t)$ %	sse %
5 knots	6852.0	1081.8	-	-	3.91	6940.7	1020.6	23.75	0.23	0
7.5 knots	5082.0	656.1	-	-	3.91	5083.5	613.1	19.67	0.17	0
10 knots	4212.2	451.4	-	-	3.91	4191.6	424.3	17.91	0	0

Table 4.3: Autopilot robustness to forward speed variations - course-change of 40°.

Clearly, the backpropagation and simulated annealing autopilots failed to significantly improve upon the original 9 rule fuzzy autopilot course-changing performance. The backpropagation tuned autopilot showed excessive overshoot values at all surge velocities. Additionally, this autopilot resulted in unsatisfactory values for steady-state error, as did the simulated annealing tuned autopilot.

The most successful tuning regime was seen to be that of the chemotaxis approach which produced an AUV course-changing response improvement of 0.46% but at the expense of 0.72% increase in control effort per rudder (and hence a 1.44% total increase in canard rudder effort). As the cost function employed during each tuning procedure did not explicitly account for control effort minimization, these increases in

control effort can be expected. Moreover, the rise time of this autopilot was 0.86% smaller and the peak percentage overshoot 82.35% smaller than the pre-tuned 9 rule autopilot at the tuning velocity of 7.5-knots.

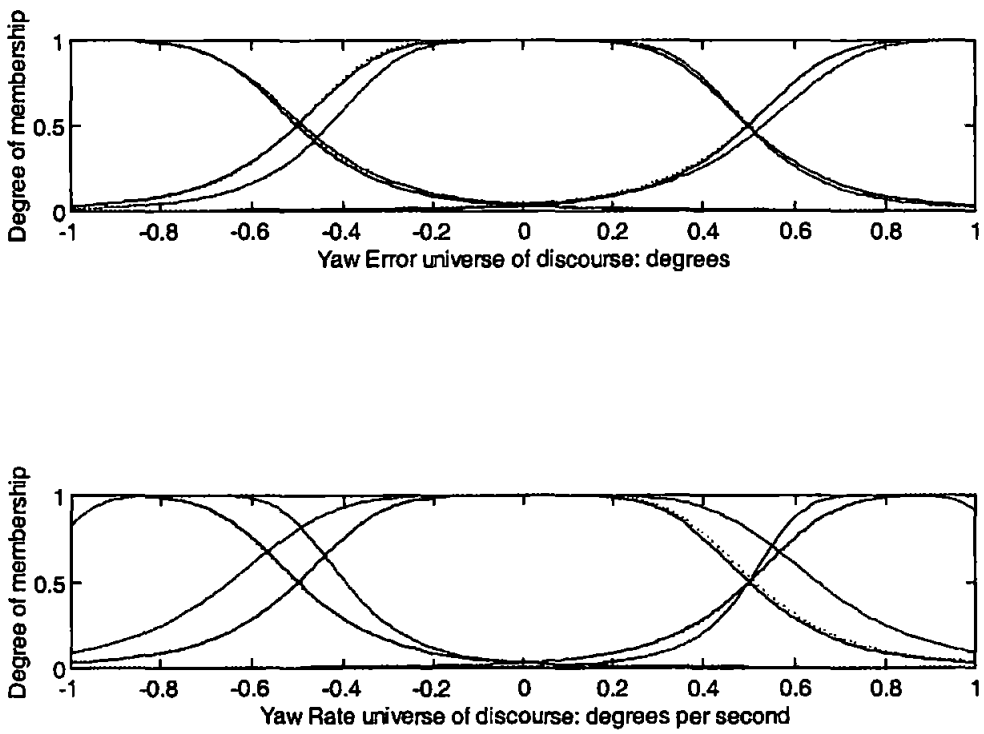


Figure 4.7: The tuned input fuzzy sets. The solid, dashed-dot and dotted lines represent the sets for the backpropagation, chemotaxis, and simulated annealing autopilots respectively.

Examination of the control surfaces for these autopilots (Figure 4.9) provides some insight into the results obtained during these simulations. These surfaces highlight that the tuning regimes have failed to significantly adjust the input-output relationship of the original fuzzy autopilot.

A verification course was designed in order that each autopilot be verified over a wider range of course-changing manoeuvres. Figure 4.10 illustrates the yaw and low canard rudder responses of the AUV over this particular course when employing each autopilot at 7.5-knots.

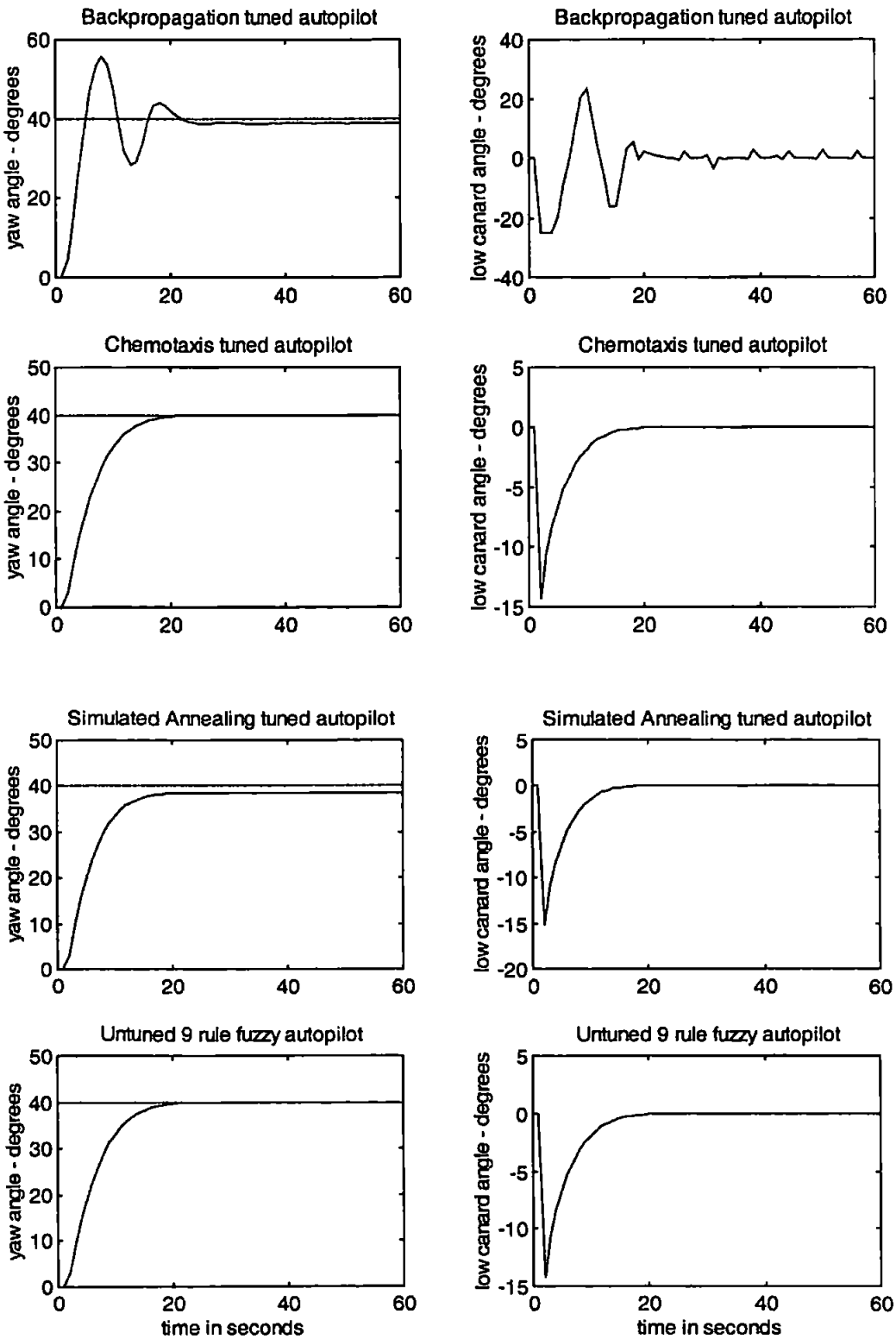


Figure 4.8: AUV responses to a course-changing manoeuvre of 40° at 7.5-knots.

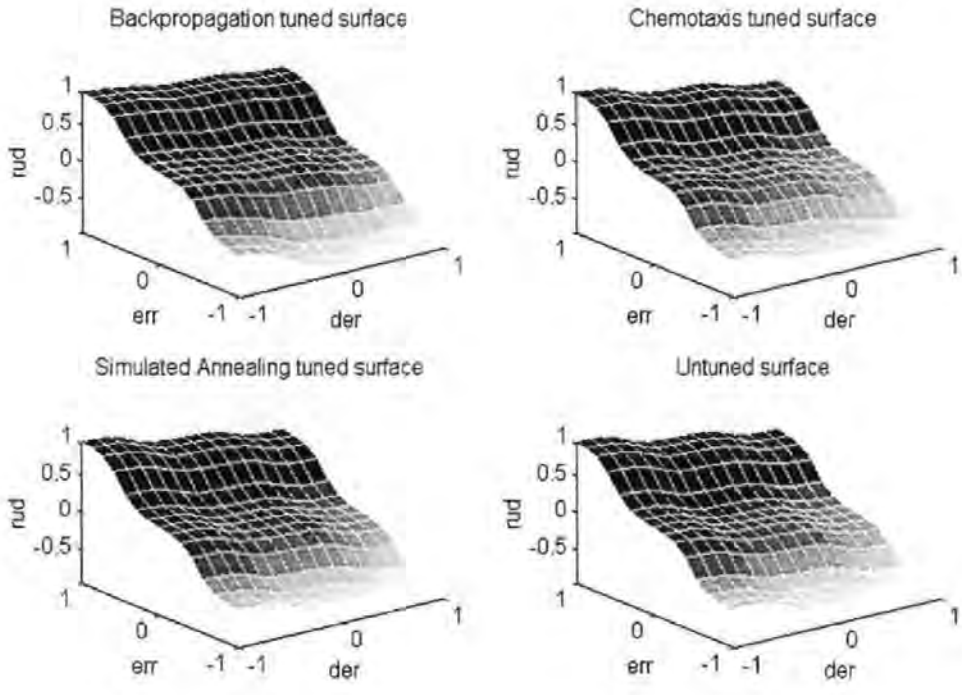


Figure 4.9: The control surfaces for the tuned 9 rule TSK fuzzy autopilots.

From these simulations it is evident that the backpropagation and simulated annealing tuning regimes have resulted in completely unsuitable course-changing autopilots which exhibit oscillatory behaviour and steady state error respectively about the designated set-points. The chemotaxis tuned autopilot produces the superior course-changing behaviour of all the fuzzy autopilots, but does not provide a significant improvement over the pre-tuned autopilot to warrant the extensive tuning periods involved. The MATLAB FIS files pertaining to the premise tuning results can be viewed in Appendix C.

Indeed, extended training periods were administered to attempt to improve the performance of these premise simulation results. However, the resulting autopilots still exhibited undesirable properties in light of this. Indeed, the backpropagation tuned autopilot illustrated even poorer performance with respect to the verification course.

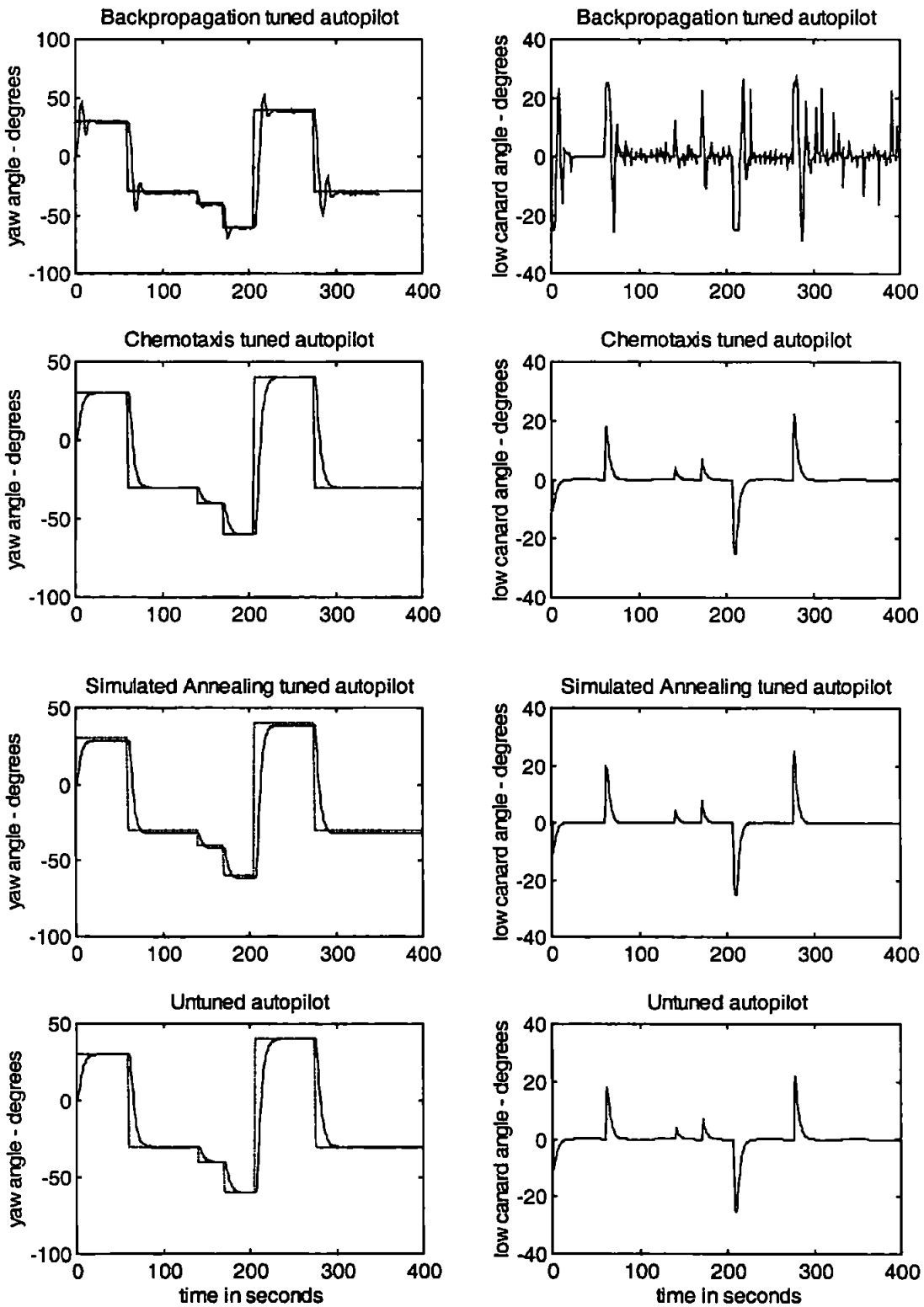


Figure 4.10: AUV responses to a verification course at 7.5-knots using the tuned and untuned 9 rule fuzzy autopilots.

This behaviour was captured during the tuning phase within the cost function histories of training and checking errors, the checking error having surpassed the lowest point on the training error curve history as shown in Figure 4.11.

The ability of the resulting fuzzy models to approximate the non-linear function representing the autopilots input-output behaviour is clearly restricted by not considering the adaptation of the consequent parameters. Preliminary tuning of the full autopilot parameter sets yielded results that were considered far superior to these initial results. This is the topic of the following section.

4.3.2 Simultaneous Premise and Consequent Parameter Tuning

The hybrid learning rule and random search algorithms were subsequently applied to the task of tuning both the premise and consequent parameters of the 9 rule TSK fuzzy autopilot. Tuning of the network parameters again took place at 7.5-knots over 300 epochs. The input-output data set used for the backpropagation tuning of the previous section was employed during the hybrid rule tuning regime to elicit comparisons.

The rulebase of the hybrid learning algorithm tuned autopilot was taken as Eqn.(4.39):

$$
\begin{aligned}
&\text{If } \psi_\varepsilon \text{ is N and } \dot{\psi} \text{ is N then } \delta = -0.487\,\psi_\varepsilon - 0.879\,\dot{\psi} - 0.029\\
&\text{If } \psi_\varepsilon \text{ is N and } \dot{\psi} \text{ is Z then } \delta = -0.489\,\psi_\varepsilon - 0.902\,\dot{\psi} + 0.001\\
&\text{If } \psi_\varepsilon \text{ is N and } \dot{\psi} \text{ is P then } \delta = -0.486\,\psi_\varepsilon - 0.896\,\dot{\psi} + 0.003\\
&\text{If } \psi_\varepsilon \text{ is Z and } \dot{\psi} \text{ is N then } \delta = -0.299\,\psi_\varepsilon - 0.703\,\dot{\psi} - 0.123\\
&\text{If } \psi_\varepsilon \text{ is Z and } \dot{\psi} \text{ is Z then } \delta = -0.488\,\psi_\varepsilon - 0.891\,\dot{\psi} + 0.004\\
&\text{If } \psi_\varepsilon \text{ is Z and } \dot{\psi} \text{ is P then } \delta = -0.305\,\psi_\varepsilon - 0.306\,\dot{\psi} - 0.037\\
&\text{If } \psi_\varepsilon \text{ is P and } \dot{\psi} \text{ is N then } \delta = -0.590\,\psi_\varepsilon - 0.839\,\dot{\psi} - 0.117\\
&\text{If } \psi_\varepsilon \text{ is P and } \dot{\psi} \text{ is Z then } \delta = -0.481\,\psi_\varepsilon - 1.081\,\dot{\psi} - 0.061\\
&\text{If } \psi_\varepsilon \text{ is P and } \dot{\psi} \text{ is P then } \delta = -0.659\,\psi_\varepsilon - 1.311\,\dot{\psi} + 0.781
\end{aligned}
\qquad (4.39)
$$

whereas the rulebase of the chemotaxis tuned autopilot was taken as Eqn.(4.40):

$$
\begin{aligned}
&\text{If } \psi_\varepsilon \text{ is N and } \dot{\psi} \text{ is N then } \delta = 0.129\,\psi_\varepsilon - 0.058\,\dot{\psi} + 0.143\\
&\text{If } \psi_\varepsilon \text{ is N and } \dot{\psi} \text{ is Z then } \delta = -0.101\,\psi_\varepsilon - 0.029\,\dot{\psi} + 0.148\\
&\text{If } \psi_\varepsilon \text{ is N and } \dot{\psi} \text{ is P then } \delta = -0.025\,\psi_\varepsilon - 0.061\,\dot{\psi} - 0.044\\
&\text{If } \psi_\varepsilon \text{ is Z and } \dot{\psi} \text{ is N then } \delta = -0.028\,\psi_\varepsilon - 0.109\,\dot{\psi} + 0.146\\
&\text{If } \psi_\varepsilon \text{ is Z and } \dot{\psi} \text{ is Z then } \delta = -0.202\,\psi_\varepsilon + 0.085\,\dot{\psi} - 0.005\\
&\text{If } \psi_\varepsilon \text{ is Z and } \dot{\psi} \text{ is P then } \delta = 0.149\,\psi_\varepsilon + 0.145\,\dot{\psi} - 0.067\\
&\text{If } \psi_\varepsilon \text{ is P and } \dot{\psi} \text{ is N then } \delta = -0.124\,\psi_\varepsilon + 0.067\,\dot{\psi} - 0.041\\
&\text{If } \psi_\varepsilon \text{ is P and } \dot{\psi} \text{ is Z then } \delta = -0.084\,\psi_\varepsilon + 0.170\,\dot{\psi} + 0.160\\
&\text{If } \psi_\varepsilon \text{ is P and } \dot{\psi} \text{ is P then } \delta = 0.009\,\psi_\varepsilon + 0.035\,\dot{\psi} + 0.059
\end{aligned}
\qquad (4.40)
$$

and, the rulebase of the simulated annealing tuned autopilot was taken as Eqn.(4.41):

$$
\begin{aligned}
&\text{If } \psi_\varepsilon \text{ is N and } \dot{\psi} \text{ is N then } \delta = 0.122\,\psi_\varepsilon - 0.215\,\dot{\psi} + 0.268\\
&\text{If } \psi_\varepsilon \text{ is N and } \dot{\psi} \text{ is Z then } \delta = -0.042\,\psi_\varepsilon - 0.011\,\dot{\psi} + 0.314\\
&\text{If } \psi_\varepsilon \text{ is N and } \dot{\psi} \text{ is P then } \delta = 0.404\,\psi_\varepsilon - 0.326\,\dot{\psi} + 0.102\\
&\text{If } \psi_\varepsilon \text{ is Z and } \dot{\psi} \text{ is N then } \delta = 0.115\,\psi_\varepsilon - 0.215\,\dot{\psi} + 0.351\\
&\text{If } \psi_\varepsilon \text{ is Z and } \dot{\psi} \text{ is Z then } \delta = -0.472\,\psi_\varepsilon + 0.492\,\dot{\psi} - 0.004\\
&\text{If } \psi_\varepsilon \text{ is Z and } \dot{\psi} \text{ is P then } \delta = 0.434\,\psi_\varepsilon + 0.291\,\dot{\psi} - 0.254\\
&\text{If } \psi_\varepsilon \text{ is P and } \dot{\psi} \text{ is N then } \delta = -0.262\,\psi_\varepsilon + 0.281\,\dot{\psi} - 0.190\\
&\text{If } \psi_\varepsilon \text{ is P and } \dot{\psi} \text{ is Z then } \delta = -0.053\,\psi_\varepsilon + 0.110\,\dot{\psi} + 0.100\\
&\text{If } \psi_\varepsilon \text{ is P and } \dot{\psi} \text{ is P then } \delta = -0.083\,\psi_\varepsilon + 0.163\,\dot{\psi} + 0.059
\end{aligned}
\qquad (4.41)
$$

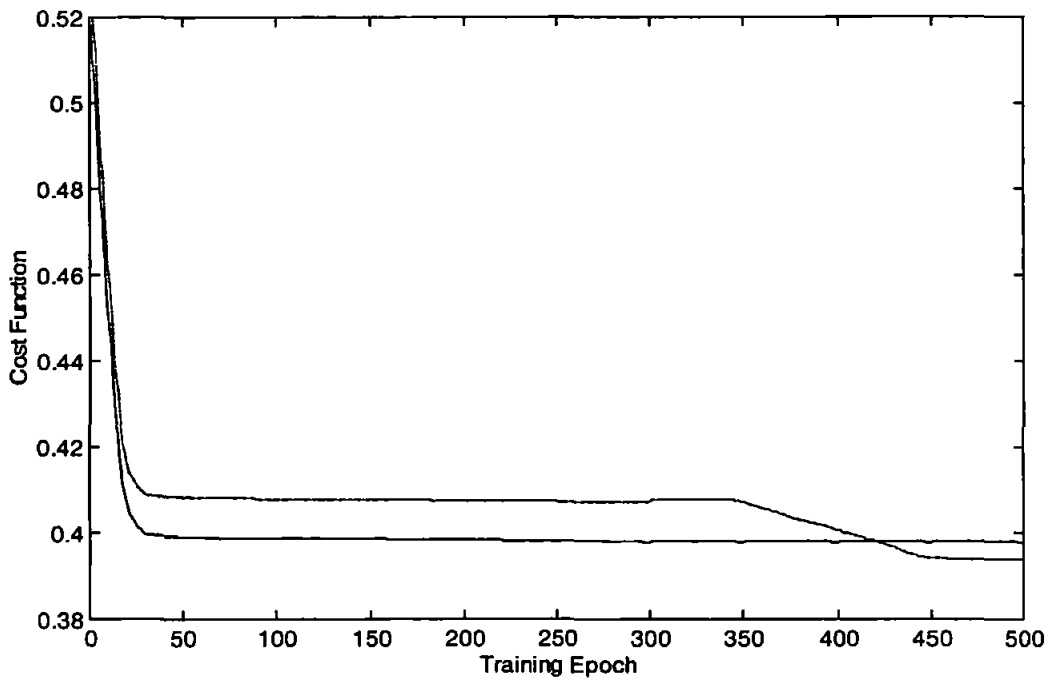


Figure 4.11: The training (solid line) and checking error (dashed-dot line) curves for the extended training period.

The resulting input fuzzy sets for the tuned autopilots were taken as shown in Figure 4.12, the full parameter detail being provided in Appendix D. Again note the non-symmetrical nature of the tuned input fuzzy sets over the original fuzzy sets of Figure 4.3. With regard to the hybrid rule tuned sets, these asymmetries are possibly accredited to the initial conditions required to bootstrap the recursive least-squares algorithm as well as computer truncation errors.

Again, the resulting autopilots were employed within the AUV model for the 40° course-changing task at 5, 7.5 and 10-knots to examine their generalization and robustness to AUV speed variations. Figure 4.13 illustrates the AUV responses for the 7.5-knot simulations.

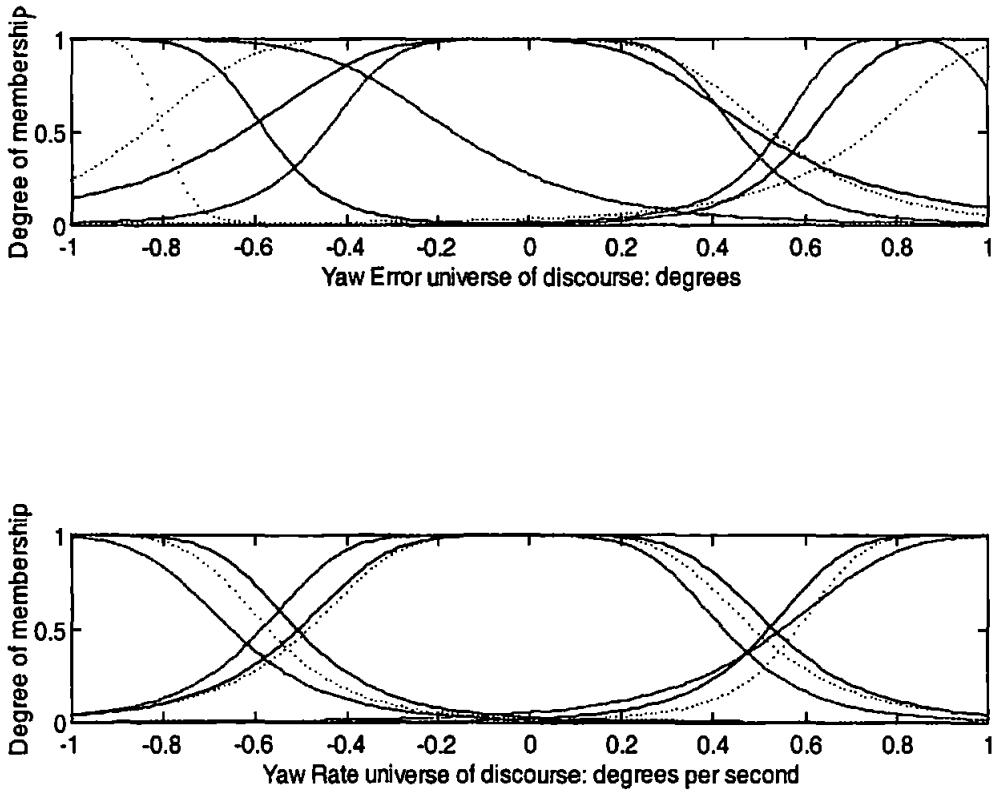


Figure 4.12: The input fuzzy sets for the tuned autopilots. The solid, dashed-dot and dotted lines represent the sets for the hybrid rule, chemotaxis, and simulated annealing autopilots respectively.

The results pertaining to all three surge velocities are reproduced in Table 4.4. The hybrid learning rule produced the most accurate course-changing autopilot for premise and consequent parameter tuning. The course-changing results being 15.00%, 12.00% and 39.86% more accurate than the chemotaxis tuned, simulated annealing tuned and pre-tuned autopilots respectively at 7.5-knots. The simulated annealing tuned autopilot is neglected in the discussions from this point forward due to the unacceptable levels of overshoot at all three surge velocities.

The speed of response for the hybrid rule tuned autopilot was 10.21% and 61.11% faster than those of the chemotaxis tuned and pre-tuned fuzzy autopilots at 7.5-knots. However, the peak overshoot exhibited by this autopilot was 175 times greater and

100% greater than the chemotaxis and pre-tuned autopilots respectively. However, at all three speeds, the overshoot of the hybrid rule tuned autopilot was quite acceptable, never exceeding 1.12 degrees. Again the control effort demands for the hybrid tuned autopilot were greater than those of the pre-tuned autopilot as the cost function made no provision for control effort minimization.

AUV model	Hybrid Algorithm Tuning					Chemotaxis Tuning				
	ψ_ε $(^\circ)^2$	δ_ε $(^\circ)^2$	T_R sec	$M_p(t)$ %	sse %	ψ_ε $(^\circ)^2$	δ_ε $(^\circ)^2$	T_R sec	$M_p(t)$ %	sse %
5-knots	4401.2	2450.9	9.85	2.79	0	5295.2	1647.1	12.58	3.33	1.25
7.5-knots	3057.4	1451.7	7.65	1.75	0	3597.0	1154.5	8.52	0.01	0.16
10-knots	2410.7	987.1	6.53	1.35	0	2785.6	913.7	6.74	5.86	0.16

AUV model	Simulated Annealing Tuning					Untuned Sugeno fuzzy autopilot				
	ψ_ε $(^\circ)^2$	δ_ε $(^\circ)^2$	T_R sec	$M_p(t)$ %	sse %	ψ_ε $(^\circ)^2$	δ_ε $(^\circ)^2$	T_R sec	$M_p(t)$ %	sse %
5-knots	4420.9	3421.5	7.38	14.83	0.38	6940.7	1020.6	23.75	0.23	0
7.5-knots	3474.4	3217.6	5.25	32.95	0.35	5083.5	613.1	19.67	0.17	0
10-knots	3581.8	4125.6	4.34	21.96	0.35	4191.6	424.3	17.91	0	0

Table 4.4: AUV responses to a course-change of 40° at 5, 7.5 and 10-knots.

The hybrid rule tuned autopilot illustrated no evidence of steady-state error whereas the chemotaxis algorithm tuned autopilot showed a steady-state error in excess of 0.16% at all three tested surge velocities. Indeed, the chemotaxis algorithm tuned autopilot would seem more suitable than the hybrid rule tuned autopilot with respect to overshoot and control effort. However the response of the each autopilot when simulated at an AUV surge velocity of 5-knots illustrates the superior robustness of the hybrid rule tuned autopilot, as depicted in Figure 4.14.

In summary, the robustness to forward speed of the hybrid rule tuned autopilot was superior to that of both the chemotaxis tuned and pre-tuned fuzzy autopilots, highlighting the effectiveness of this tuning regime in this instance.

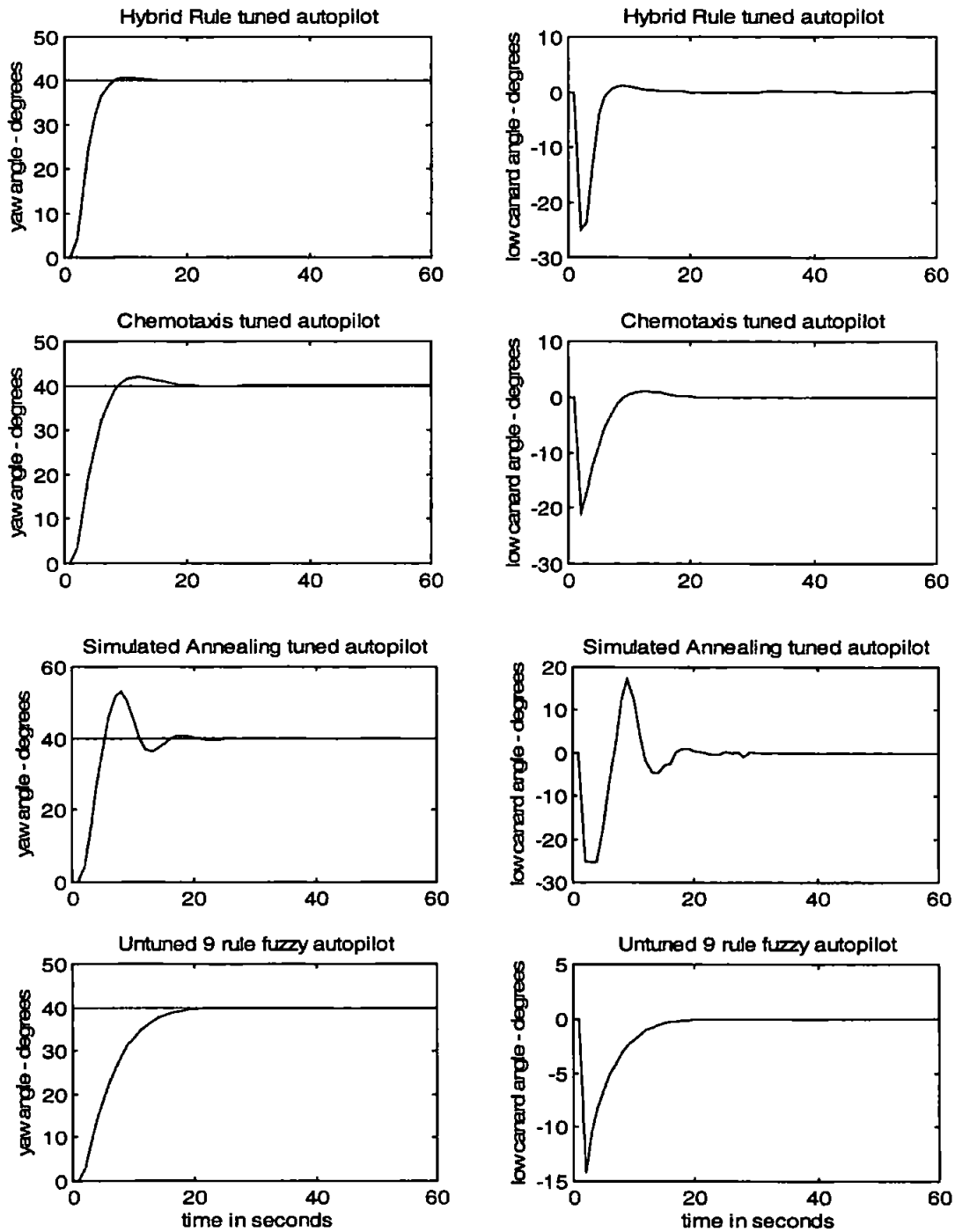


Figure 4.13: AUV responses to a 40° course-change at 7.5-knots when employing the tuned fuzzy autopilots.

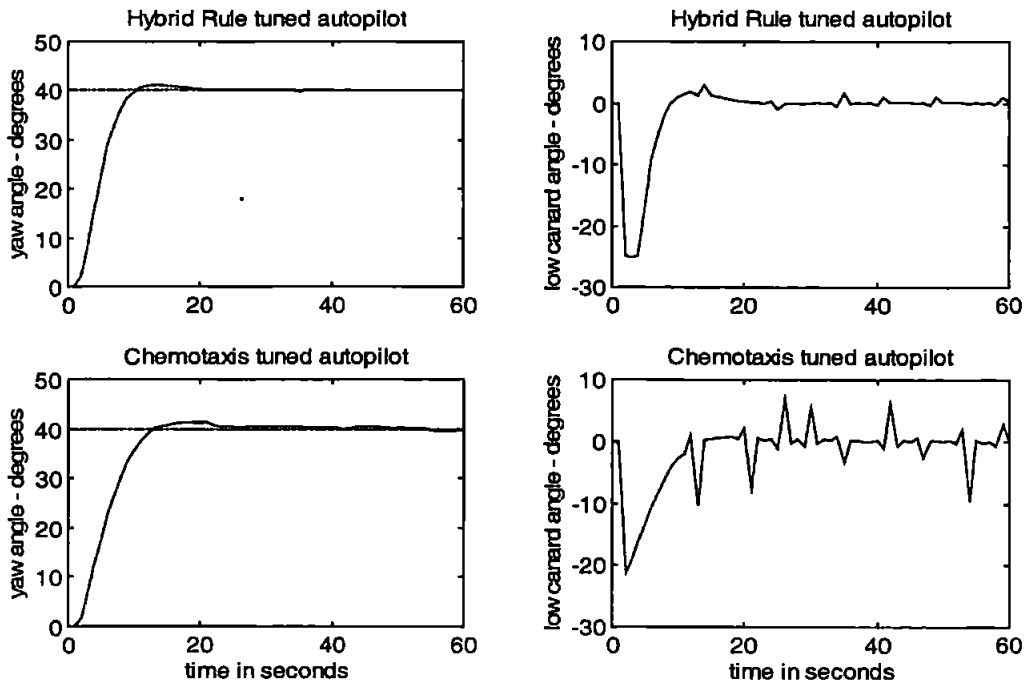


Figure 4.14: A comparison of the hybrid rule tuned and chemotaxis tuned autopilots robustness at 5-knots.

In order to assess the generalization capabilities of each tuned autopilot the verification course of Figure 4.10 was employed. Figure 4.15 illustrates the AUV yaw and low canard rudder responses when employing the 4 fuzzy autopilots over the verification course at 7.5-knots. The hybrid rule and chemotaxis tuned fuzzy autopilots illustrate good performance when presented with course changes for which they have not been explicitly tuned. The simulated annealing tuned autopilot provides poor course-changing control of the AUV with highly oscillatory behaviour. Figure 4.16 depicts the control surface for each tuned autopilot.

The control surface of the hybrid tuned autopilot exhibits a particularly flat profile, providing smooth interpolation between individual fuzzy rules. This is in contrast to the control surfaces of the random search techniques, which display sections with steep gradients. Clearly, such surfaces will lead to oscillatory actuator movement when the input vector pertains to a control output in these regions.

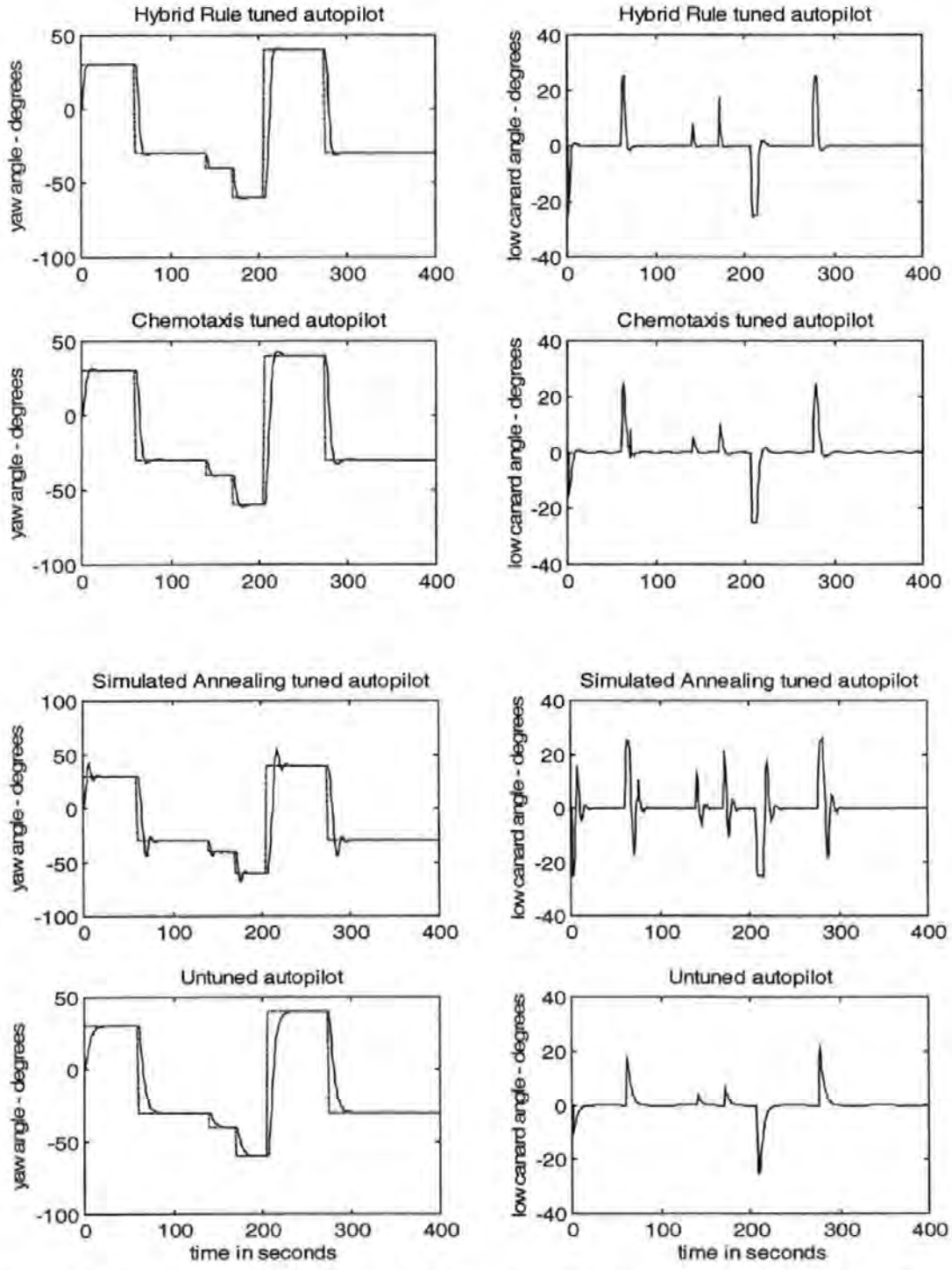


Figure 4.15: AUV responses to a verification course at 7.5-knots using the tuned and untuned 9 rule fuzzy autopilots.

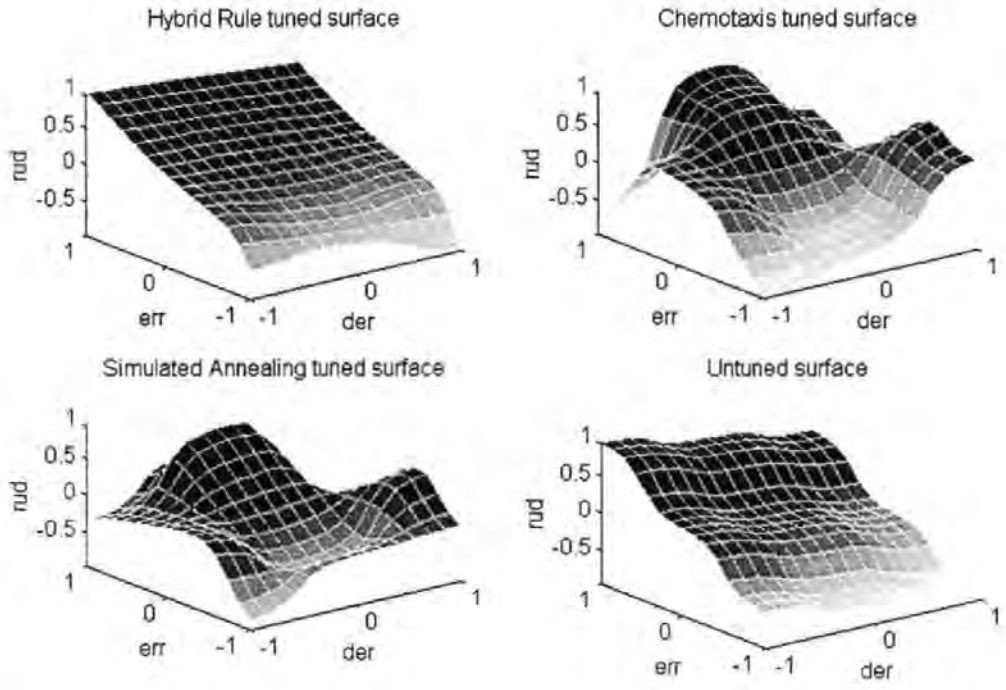


Figure 4.16: The control surfaces for the tuned 9 rule TSK fuzzy autopilots.

4.4 Autopilot Robustness

The preceding sections have discussed the relative merits of various tuning regimes in both quantitative and qualitative form. Subsequently, the hybrid tuned fuzzy autopilot was proposed as the most suitable for course-changing. This section assesses the performance of this autopilot in the presence of coefficient uncertanties. Numerous coefficients were varied including the vehicle mass (to simulate deployment of payloads) and the hydrodynamic coefficients pertaining to the course-changing dynamics; these experiments are discussed in section 4.4.1. Additionally, a line of sight guidance algorithm, as detailed in Chapter 3 section 4, was employed to simulate the presence of sea current disturbances. These simulation results are given in section 4.4.2.

4.4.1 Vehicle Coefficient Variations

Figure 4.17 depicts the yaw and low canard rudder responses of the AUV when the mass of the vehicle is increased by 75%. The nominal mass of the vehicle is

approximately 3,600 kilograms but will clearly increase with payload. For example, if the vehicle is employed for covert mine disposal, the mass of the vehicle may increase by anything from 275 kilograms to 2,750 kilograms (approximately 7.5% to 75%).

Evidently, the hybrid rule tuned autopilot performs well in light of the expected increased AUV payload. Evidence of increased overshoot is apparent as the overall system is now under-damped. This illustrates the sophistication of the AUV model, which compensates for reduced yaw damping in light of a mass increase. Additionally, the following perturbations in the hydrodynamic coefficients are considered:

- ± 20% variation in Y_{uv}, the sway damping coefficient
- ± 20% variation in Y_{ur}, the yaw into sway coefficient
- ± 20% variation in N_{uv}, the sway into yaw coefficient
- ± 20% variation in N_{ur}, the yaw damping coefficient

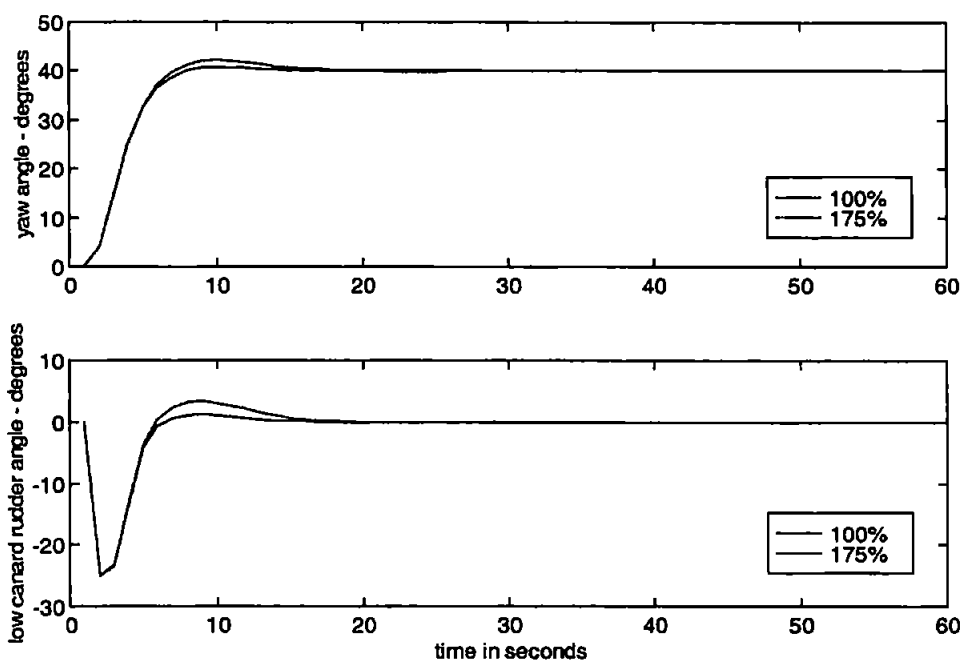


Figure 4.17: Mass variation during a 40° course-change when employing the hybrid tuned autopilot.

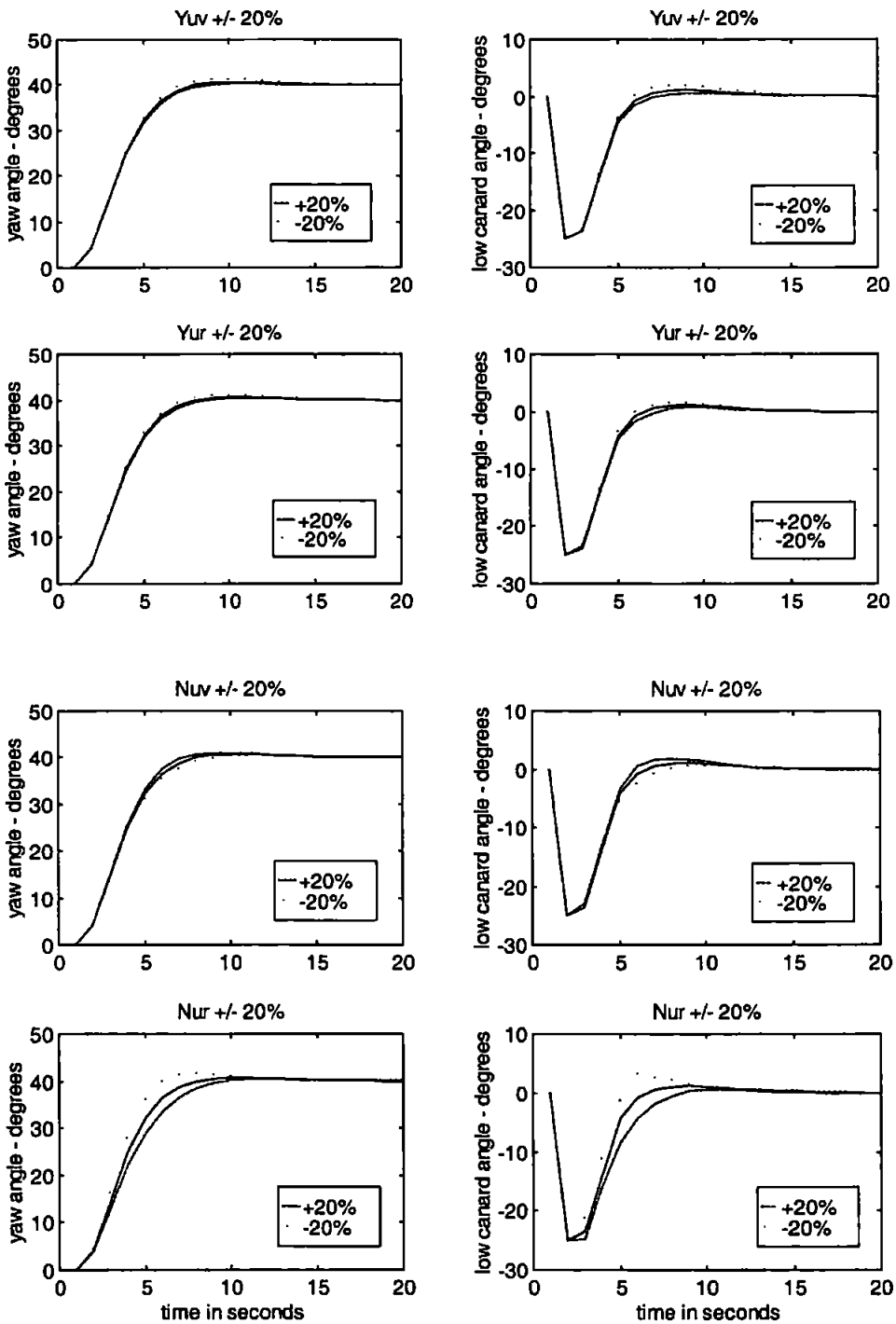


Figure 4.18: Hydrodynamic coefficient variations during a 40° course-changing manoeuvre when employing the hybrid tuned autopilot.

As the hydrodynamic coefficients are a function of the total velocity squared (Eqn.(4.42)), they will typically vary with velocity over a mission. It is therefore important that the autopilot is able to accommodate such variations effectively.

$$Y_{UV}, Y_{UR}, ..., N_{UV}, N_{UR} = f\left(\underline{u}^2 + \underline{v}^2 + \underline{w}^2\right) \tag{4.42}$$

The responses of the AUV to a 40° course-changing manoeuvre when employing the hybrid rule tuned autopilot under these coefficient variations are illustrated in Figure 4.18. These simulations highlight the robust performance of the hybrid rule tuned autopilot in the presence of these hydrodynamic coefficient variations. Clearly, the increase in sway, yaw and yaw into sway hydrodynamic damping coefficients produces a slightly over-damped course-changing response. Conversely, the increase in the sway into yaw coefficient leads to an under-damped response as one would intuitively expect, the vehicle having increased inertia in this axis.

4.4.2 Line of Sight Guidance

Autonomous guidance of the vehicle is achieved by employing the line of sight (LOS) guidance algorithm. The heading command to the vehicles canard rudders is provided by calculating the yaw angle between the vehicles current position and the target way-point with trigonometry (Eqn.(3.10)). Healey and Lienard (1993) employed this approach to assess the effectiveness of their multivariable sliding mode diving and steering autopilot. Certain observations concerning LOS guidance can be inferred from this study:

- way-points in close proximity to one another produce overshoots in vehicle heading
- a radius of acceptance β of approximately 2 vehicle lengths is adequate for accurate course-changing but can lead to overshoots in vehicle heading

- vehicle forward speed is controlled independently but can be incorporated into the LOS algorithm to schedule the acceptance region β

Based upon these comments, a verification track was devised to appraise the robustness properties of the hybrid rule tuned autopilot against a more PD autopilot (Eqn.(3.14)) and the untuned 9 fuzzy rule TSK autopilot. The radius of acceptance β was initially fixed at 15 metres (approximately 2 vehicle lengths) and the nominal vehicle speed at 7.5-knots.

***x co-ordinate* (metres)**	500	750	1000	1000	1000	500	0
***y co-ordinate* (metres)**	500	500	500	450	200	0	0

Table 4.5: Co-ordinates of the way-points within the mission management system.

Initial simulations were conducted in the absence of sea currents. Figure 4.19 is a reproduction of the responses for the hybrid rule tuned, untuned and PD autopilots with respect to the aforementioned verification track. Table 4.5 provides the exact co-ordinates of each way-point. These points are shown in Figure 4.19 as circles. However, these circles do not represent the 15-metre radius of each individual way-point.

The vehicle selects the next way-point co-ordinates in a clockwise manner beginning and ending at the origin. Note further the inclusion of the way-points positioned at [1000,500] and [1000,450] respectively. These way-points are deliberately located in close proximity to one another to assess the damping of each autopilot in terms of course overshoot. Each of the autopilots performs the course-changing demands successfully in the absence of sea current.

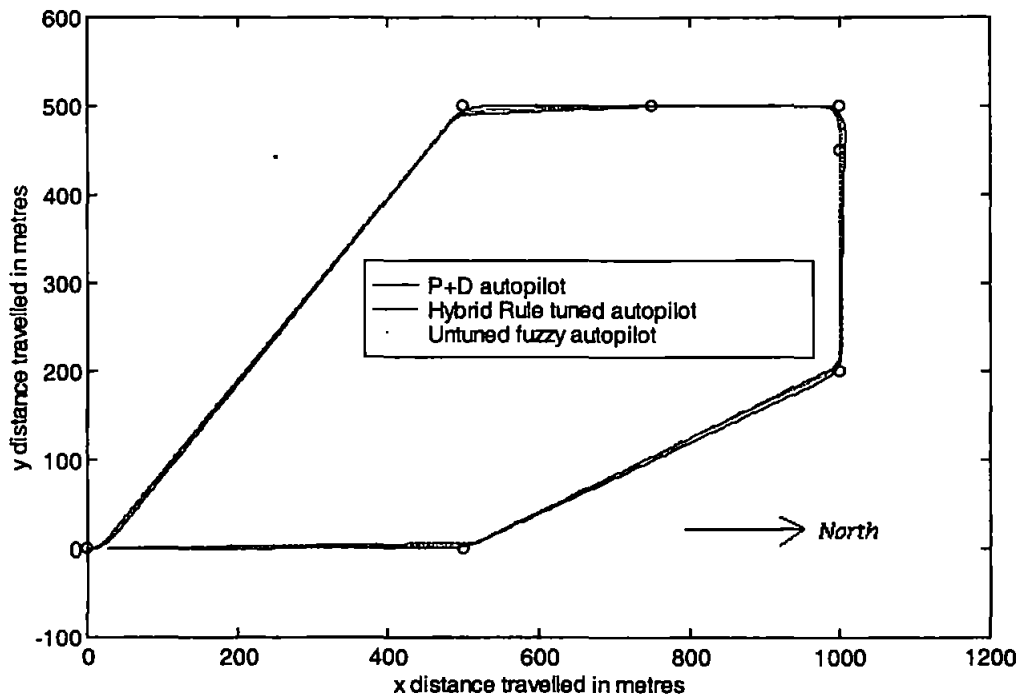


Figure 4.19: Line of sight responses over the verification track in the absence of current disturbances.

Obviously the effect of a given current upon the AUV will vary from favourable to adverse depending on the vehicle orientation (when the current remains constant in direction and magnitude). Initially, a constant current of 2 ms^{-1} along the +ve x-axis (Northerly) was invoked. The results pertaining to this simulation are depicted in Figure 4.20. The ineffectiveness of the untuned fuzzy autopilot is apparent in this simulation; the AUV fails to complete the designated course. Alternatively, the linear PD autopilot performed particularly well, as did the hybrid rule tuned fuzzy autopilot. To further compare these two controllers, the current disturbance was increased to 2.5 ms^{-1} along the +ve x-axis. These results are shown in Figure 4.21.

It is now apparent that the hybrid rule tuned autopilot provides a superior course-changing response to that of the PD autopilot in the presence of such current conditions. Indeed, the PD autopilot fails to select the final way-point correctly and the AUV circles around the penultimate way-point before heading off to the origin. This behaviour can be explained more clearly by consideration of the vehicle yaw angle response (Figure 4.22). When entering the penultimate way-point's circle of acceptance, the autopilots current heading is -150°.

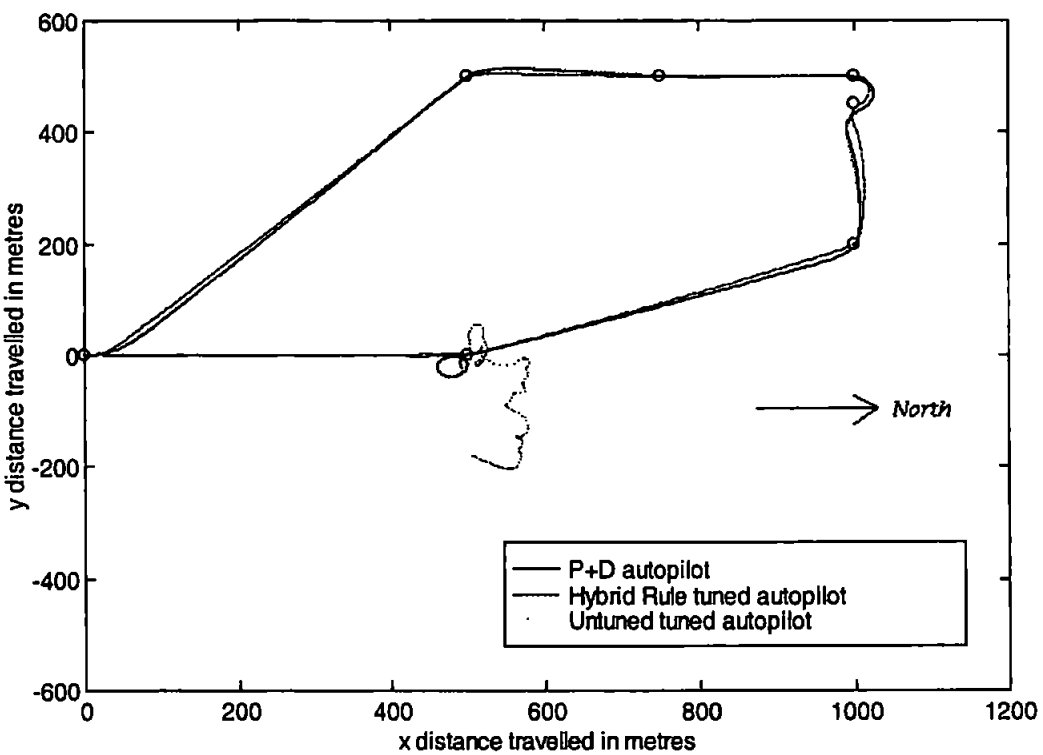


Figure 4.20: Line of sight responses over the verification track in the presence of a current disturbance of 2 ms^{-1} along the Northerly axis.

The autopilot must then select a course angle of -180° to guide the AUV to the final way-point. However, a heading angle of +180° also produces the desired effect, guiding the AUV to the final way-point. Obviously, it is preferred that the autopilot selects the course angle of -180° as this involves a change in heading of approximately 30°. If a heading angle of 180° is selected the heading demand changes through a full 330°. When this situation is encountered, the AUV circles the current way-point prior to selecting the target way-point.

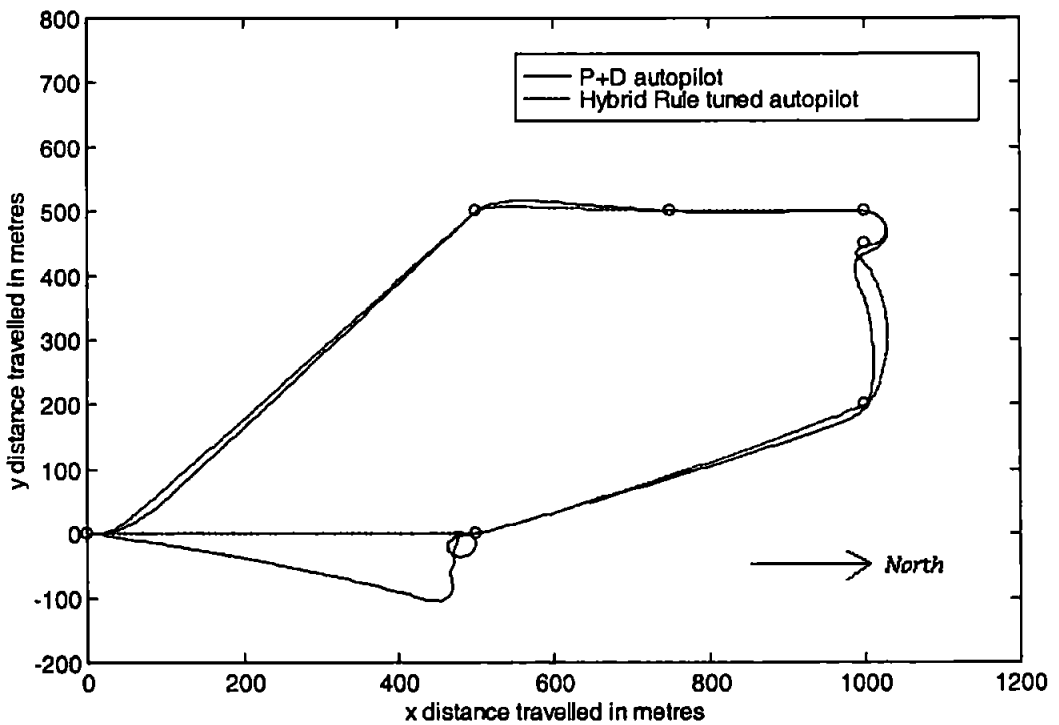


Figure 4.21: Line of sight responses over the verification track in the presence of a current disturbance of 2.5 ms^{-1} along the Northerly axis.

Further robustness experiments were carried out whilst employing a constant current of magnitude 2 ms^{-1} in the Westerly direction (in the +ve y axis). Figure 4.23 illustrates

the results of this test. Again, the hybrid rule tuned autopilot yields the superior result, visiting all 7 autopilots sequentially with no evidence of circling. Indeed, Figure 4.24 indicates the ability of this autopilot to effect control over the AUV even in the presence of a 3 ms^{-1} Westerly current.

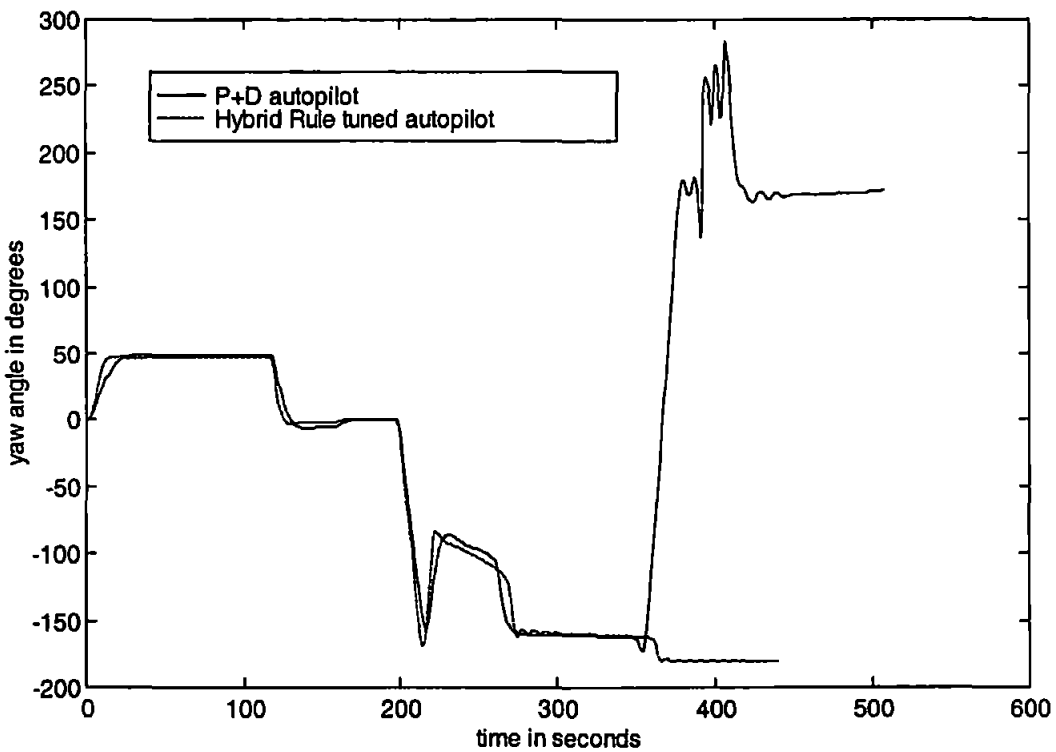


Figure 4.22: Yaw responses over the verification track in the presence of a current disturbance of 2.5 ms^{-1} along the Northerly axis.

As a final robustness experiment, a current of magnitude 2.83 ms^{-1} at an angle of 315° was applied. The effectiveness of the hybrid rule tuned fuzzy autopilot over the two other autopilots is further established by these results (Figure 4.25).

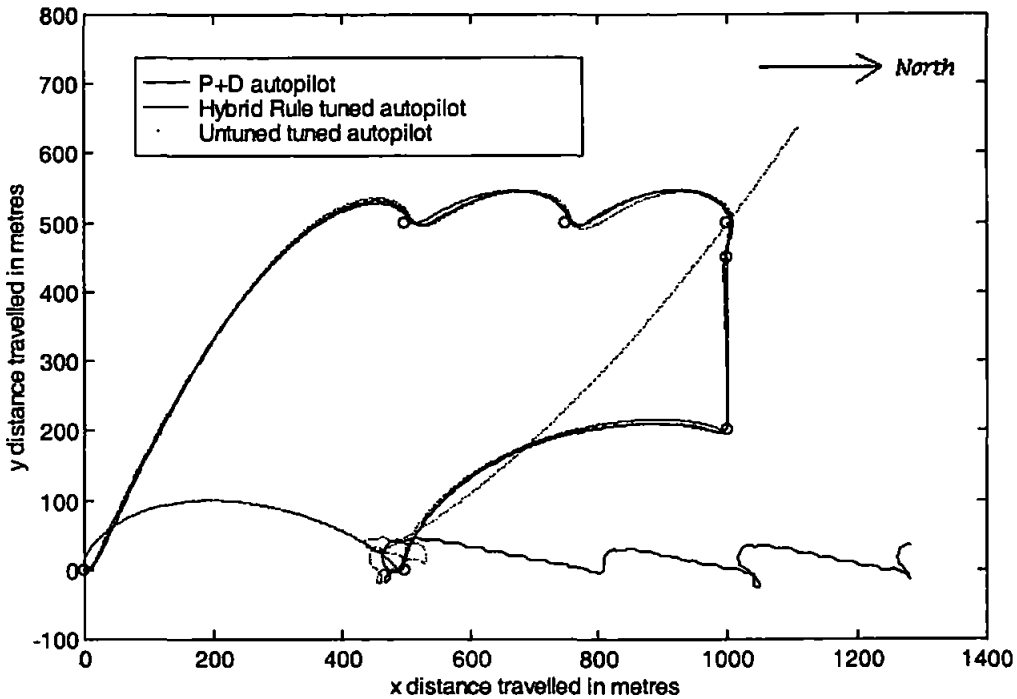


Figure 4.23: Line of sight responses over the verification track in the presence of a current disturbance of 2 ms^{-1} along the Westerly axis.

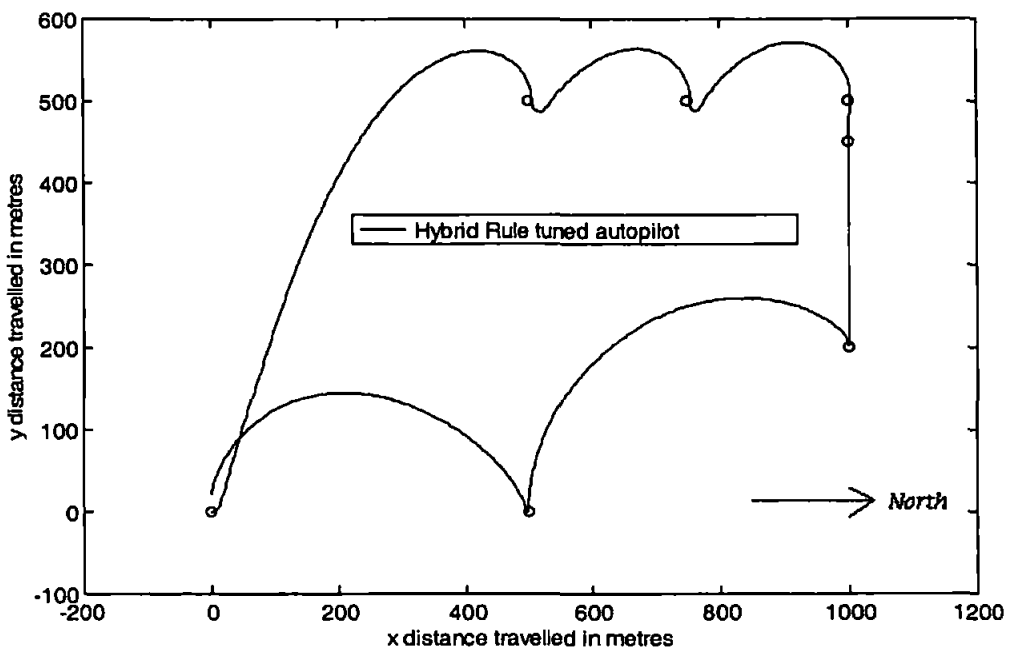


Figure 4.24: Line of sight responses over the verification track in the presence of a current disturbance of 3 ms^{-1} along the Westerly axis.

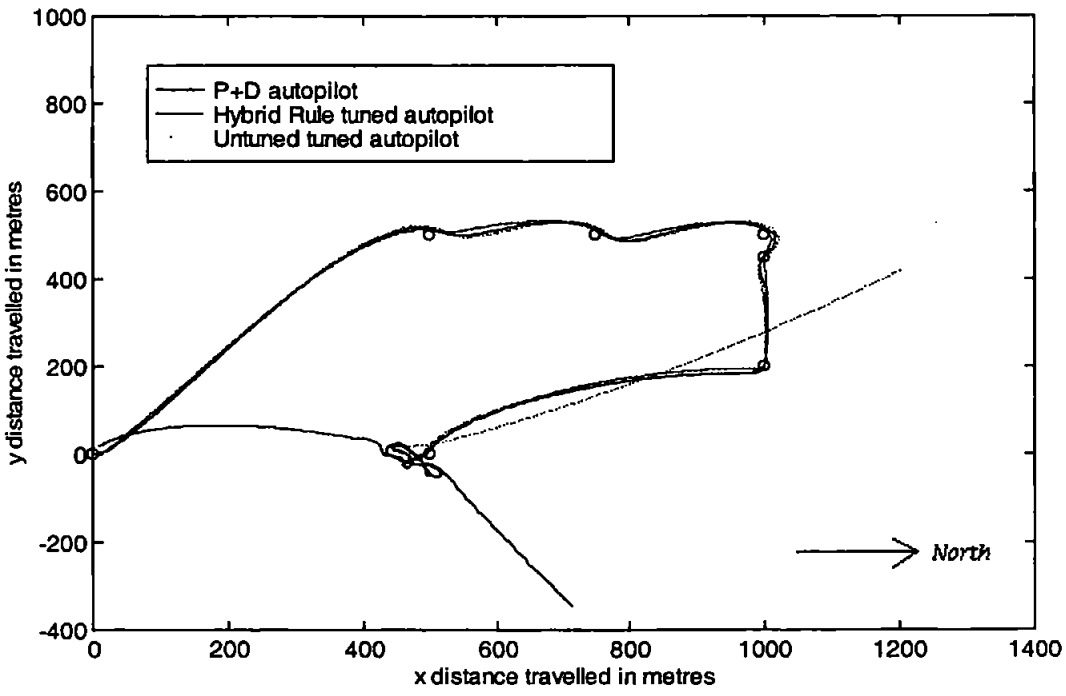


Figure 4.25: Line of sight responses over the verification track in the presence of a current disturbance of 2.83 ms^{-1} in the North Westerly direction.

4.5 Concluding Remarks

Course-changing autopilots for the AUV have been developed based on variants of the TSK 9 rule fuzzy model. Premise parameter tuning strategies resulted in unsatisfactory results and prompted the use of full parameter tuning regimes.

Whilst the random search techniques of chemotaxis and simulated annealing should produce autopilots with globally optimal parameter sets, the extensive tuning periods required limit their applicability to AUV autopilot designs which must be flexible to accommodate new vehicle structures and payloads. However, use of the hybrid learning rule produced an excellent autopilot for course-changing control, which required a minimal tuning period as compared to the random search techniques. Robustness investigations further confirmed the superiority of the hybrid rule tuned fuzzy autopilot

for course-changing control of the AUV and prompted the research of the following chapter.

Whilst the autopilot tuning strategies are neuro-fuzzy in nature, it should be noted that the resulting tuned autopilots remain purely fuzzy.

References

Healey, A. and Lienard, D. (1993). Multivariable Sliding Mode Control for Autonomous Diving and Steering of Unmanned Underwater Vehicles. IEEE Journal of Oceanic Engineering, Vol. 18, No. 3, pp327-339.

Jang, J.-S. R. (1992). Neuro-Fuzzy Modelling: Architecture, Analyses and Applications, Ph.D Thesis, Department of Electrical Engineering, University of California, Berkeley, CA 94720, United States of America.

Jang, J.-S. R., Sun, C.-T. and Mizutani, E. (1997). Neuro-Fuzzy and Soft Computing. Prentice-Hall, New Jersey.

Kirkpatrick, S., Gelatt, C. and Vecchi, M. (1983). Optimization by Simulated Annealing. Science, 220, pp671-680.

Koshland, D. E. (1980). Bacterial Chemotaxis in Relation to Neurobiology, Annual Review of Neuroscience, Vol. 3, pp45-75.

Takagi, T. and Sugeno, M. (1985). Fuzzy Identification of Systems and its Applications to Modelling and Control. IEEE Transactions on Systems, Man and Cybernetics, Vol. 15, pp116-132.

Taylor, S. D. H. (1995). The Design and Tuning of Fuzzy Autopilots using Artificial Neural Networks. M.Phil Thesis, University of Plymouth/RNEC.

Tomera, M. and Morawski, L. (1996). Neural-Network-Based Fuzzy Logic Marine Autopilot. Third International Symposium on Methods and Models in Automation and Robotics, pp1207-1212.

Chapter 5

Multivariable ANFIS Autopilot Design Approach

5.1 Introduction

The results of the preceding chapter highlight the effectiveness of the adaptive network-based fuzzy inference system (ANFIS) approach for tuning the parameter set of a Takagi Sugeno Kang (TSK) (Takagi and Sugeno (1985)) fuzzy autopilot. However, the technique as described is only suitable for tuning autopilots to control one degree of freedom at any one time. Owing to the highly non-linear coupled autonomous underwater vehicle (AUV) equations of motion, stimulating a dynamic response in any particular channel will invariably lead to motion in other degrees of freedom. Thus, to ensure that the AUV acts as a steady platform and makes effective use of available sonar packages, for example, it is required that such cross-coupling effects be regulated or compensated for by the vehicle autopilot system.

Traditional control methodologies often fail to compensate for the inherent coupling between plant degrees of freedom and thus some interest in multivariable approaches to UUV autopilot design has been shown in recent years. This chapter discusses the development of a novel multivariable autopilot for simultaneous control of multiple degrees of freedom and highlights the requirement for the compensation of cross-coupling effects between AUV channels. To illustrate the effectiveness of the technique, the new structure is applied to the task of course-changing control whilst regulating the roll dynamic response simultaneously.

5.2 A Brief Review of Multivariable UUV Autopilot Designs

Single input-single output (SISO) approaches to UUV control system design do not generally take account of cross-coupling effects which occur between interacting vehicle degrees of freedom, as typically shown in Figure 5.1. This often leads to a degradation in control system performance and robustness, especially if these cross-coupling disturbances become more pronounced with varying UUV dynamics.

In order that the behaviour of the vehicle matches the operational specifications for on-board sonar and sensor packages, it is often required that vehicle angular motion be decoupled from linear motion. This is the subject of the paper by Cowling and Corfield (1995) which discusses effective combinations of vehicle control surfaces for multivariable control strategies. Indeed this study was instrumental in the choices of actuator combinations used within this chapter, as the vehicle model employed herein is a direct descendent of the one used throughout their work. It is stated that for full decoupling of yaw, sway and roll motions an effective strategy is to employ locked canard rudders for yaw control, locked stern rudders for sway control and differential stern hydroplanes for roll control. For example, the use of differential port and starboard stern hydroplanes for roll decoupling ensures that a righting moment is produced to counter act roll disturbances, yet a pitching moment is not introduced.

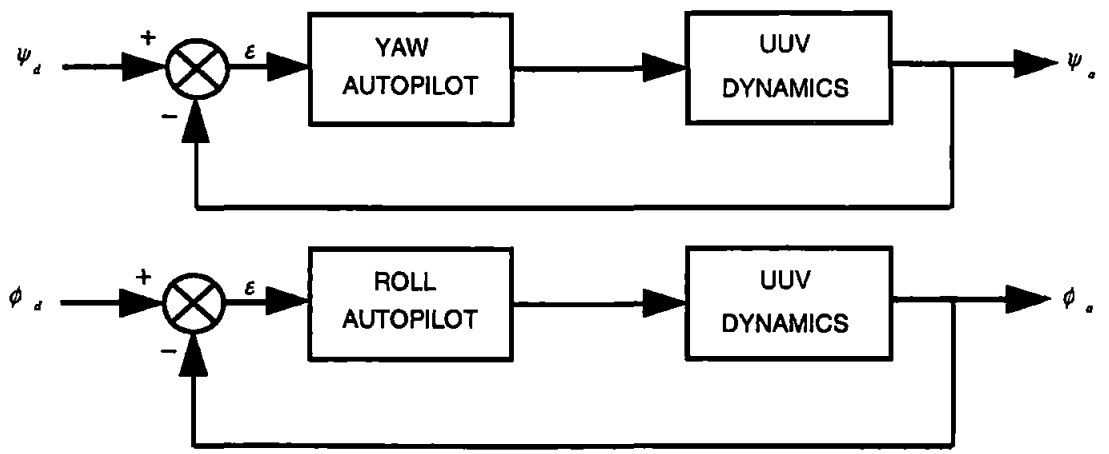


Figure 5.1: SISO approach to control of multiple UUV degrees of freedom.

Multivariable control system design techniques are well established in the control theory literature (see Maciejowski (1989)), and are employed when two or more degrees of freedom must be controlled simultaneously or when there is a need to compensate for the interaction between degrees of freedom in a plant. Indeed, some advanced multivariable control techniques have been developed and applied directly to the problem of UUV autopilot design.

Goheen and Jeffreys (1990) applied two self-tuning multivariable techniques to the control of an remotely operated vehicle (ROV), the first being based upon an implicit adaptive linear quadratic Gaussian method and the second upon a robust multi input-multi output (MIMO) de-coupling control law with an on-line recursive identification method. These approaches proved to be effective in manipulating the vehicle movement in a desired manner, but required large amounts of computational power when employed for on-line identification of the estimator matrix in the hardware system.

Healey and Lienard (1993) approached the problem in a different manner by applying a multivariable sliding mode method in which speed, steering and diving modes were de-coupled for slow speed manoeuvring. One problem associated with this study was the

limitation of available actuators on the six-degree of freedom model used. Consequently, this lead to a solution which considered a separate autopilot for each subsystem of the AUV modes of speed, steering, diving and roll, roll motion being considered passive. Subsequent results illustrate the performance of the three autopilots, but highlight their collective use as individual autopilots as opposed to a truly multivariable configuration.

Trebi-Ollennu et *al.* (1995) proposed four robust multivariable control system designs including input-output linearization, with and without sliding mode control and using adaptive fuzzy control, and H_∞ loop shaping control, all applied to the depth channel of an ROV. Although the input-output linearization technique was effective under the nominal design conditions, it was not particularly robust to perturbations in forward speed as it was designed on the basis of exact cancellation of the non-linearities in order to achieve the input-output relationship. By combining this technique with sliding mode control the resulting depth autopilot was extremely robust but the non-trivial task of raw estimates of the uncertainty bounds made this technique somewhat difficult to implement. The most effective method was seen to be the combination of input-output linearization with adaptive fuzzy control which produced a robust autopilot whereas the H_∞ technique was not considered the most effective as it was designed using a simplified model of the plant.

Choi and Hwang (1997) presented a new hybrid learning algorithm based on radial basis function networks (RBFNs). Using a multivariable self-organizing and self-learning technique a fuzzy autopilot for a submersible vehicle, based on the work of Nie and Linkens (1993), was developed. The performance of the RBFN autopilot proved effective over a straightforward fuzzy logic autopilot, but the results presented were by no means comprehensive and greater consideration needs to be given to its generalisation and robustness to parameter variations.

5.3 CANFIS – A Multivariable Design Technique

A novel approach for the tuning of multivariable fuzzy autopilots for AUV control, based upon an extension to the ANFIS technique of Jang (1992) is derived here. Similar structures have been employed by Mizutani and Jang (1995), and Jang et *al.* (1997), such as the multiple adaptive network based fuzzy inference system (MANFIS) and the co-active adaptive network-based fuzzy inference system (CANFIS) architectures. However, to the best of the present author's knowledge, there are no applications of a similar nature to the multivariable control of an AUV. Additionally, no public domain software is available to implement the CANFIS approach at the present time, this software being written solely by the present author.

The MANFIS structure is equivalent to the ANFIS in Jang et *al.* (1997) whereby Multiple-ANFIS models are considered side by side to produce a multi output configuration during the application of the hybrid learning algorithm. A significant drawback of this scheme is that the resulting independent fuzzy rule bases almost never examine all possible correlations amongst outputs, as each output is activated via a unique combination of premise parameters.

The proposed approach is functionally equivalent to that of the CANFIS architecture of Mizutani and Jang (1995) in which the premise membership functions are shared by the outputs of the network architecture. Within the rulebase each linguistic statement considers multiple outputs for a given combination of input values leading to a Co-active ANFIS design approach.

If it is assumed that the fuzzy inference system under consideration is of multivariable form, then the fuzzy rule-based algorithm may be represented in the first order TSK style:

$$
\begin{aligned}
&\text{Rule 1 :} \quad \text{If } x_1 \text{ is } A_1 \text{ and } x_2 \text{ is } B_1 \text{ and ... and } x_n \text{ is } G_1 \\
&\qquad \text{then} \quad f_{11} = a_{11}\, x_1 + b_{11}\, x_2 + \ldots + h_{11} \\
&\qquad \text{and} \quad f_{12} = a_{12}\, x_1 + b_{12}\, x_2 + \ldots + h_{12} \\
&\qquad\quad . \quad . \quad . \quad . \quad . \\
&\qquad\quad . \quad . \quad . \quad . \quad . \\
&\qquad \text{and} \quad f_{1k} = a_{1k}\, x_1 + b_{1k}\, x_2 + \ldots + h_{1k} \\
&\text{Rule 2 :} \quad \text{If } x_1 \text{ is } A_2 \text{ and } x_2 \text{ is } B_1 \text{ and ... and } x_n \text{ is } G_1 \\
&\qquad \text{then} \quad f_{21} = a_{21}\, x_1 + b_{21}\, x_2 + \ldots + h_{21} \\
&\qquad \text{and} \quad f_{22} = a_{22}\, x_1 + b_{22}\, x_2 + \ldots + h_{22} \\
&\qquad\quad . \quad . \quad . \quad . \quad . \\
&\qquad\quad . \quad . \quad . \quad . \quad . \\
&\qquad \text{and} \quad f_{2k} = a_{2k}\, x_1 + b_{2k}\, x_2 + \ldots + h_{2k} \\
&\qquad\quad . \quad . \quad . \quad . \quad . \\
&\qquad\quad . \quad . \quad . \quad . \quad . \\
&\text{Rule m :} \quad \text{If } x_1 \text{ is } A_k \text{ and } x_2 \text{ is } B_k \text{ and ... and } x_n \text{ is } G_k \\
&\qquad \text{then} \quad f_{m1} = a_{m1}\, x_1 + b_{m1}\, x_2 + \ldots + h_{m1} \\
&\qquad \text{and} \quad f_{m2} = a_{m2}\, x_1 + b_{m2}\, x_2 + \ldots + h_{m2} \\
&\qquad\quad . \quad . \quad . \quad . \quad . \\
&\qquad\quad . \quad . \quad . \quad . \quad . \\
&\qquad \text{and} \quad f_{mk} = a_{mk}\, x_1 + b_{mk}\, x_2 + \ldots + h_{mk}
\end{aligned}
\tag{5.1}
$$

where $m = k^n$ is the number of fuzzy rules per output

By encoding such a fuzzy rulebase as an adaptive network structure the proposed architecture (for two TSK rules with two outputs per rule) can be taken as that of Figure 5.2. Obviously, the actual architecture employed within this study is far more complex than that shown.

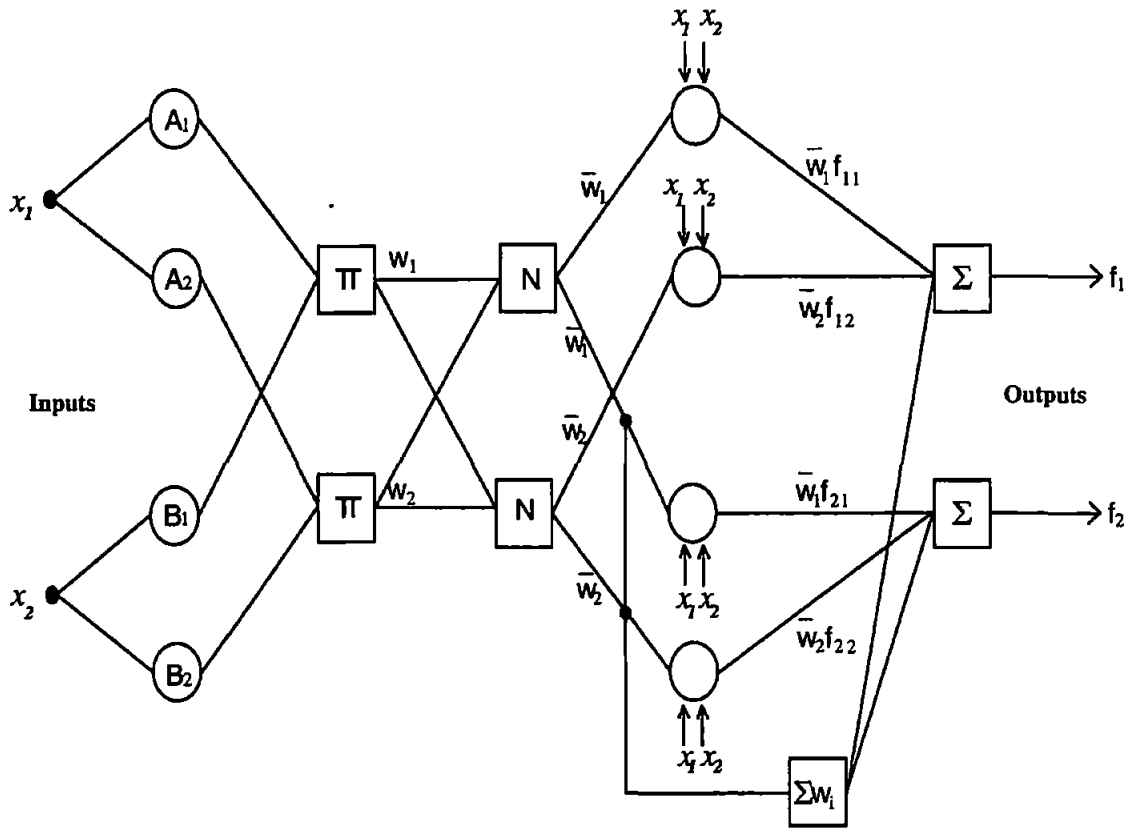


Figure 5.2: A typical CANFIS tuning architecture for two outputs.

Recalling the least squares estimator for systems with multiple outputs Eqn.(4.33)

$$\begin{cases} P_{k+1} = P_k - \dfrac{P_k \underline{a}_{k+1} \underline{a}_{k+1}^T P_k}{1 + \underline{a}_{k+1}^T P_k \underline{a}_{k+1}} \\ x_{k+1} = x_k + P_{k+1} \underline{a}_{k+1} \left(\underline{b}_{k+1}^T - \underline{a}_{k+1}^T x_k \right) \end{cases} \tag{5.2}$$

the hybrid learning rule can be applied to the process of tuning the parameters of this encoded multivariable fuzzy inference system.

5.4 Results and Discussion

This section discusses various autopilot design approaches for simultaneous control of the AUV yaw and roll dynamics. Initially, a classical non-interacting design approach is adopted to provide benchmark AUV responses. Subsequently, results pertaining to ANFIS style course-changing and roll-regulating autopilots illustrate the ineffectiveness of the ANFIS technique for combined yaw and roll control. Consequently, the proposed CANFIS structure is tuned to produce a course-changing and roll-regulating multivariable autopilot. Results illustrate the effectiveness of such an approach for the chosen application, due to the parallel tuning of the co-acting parameter set, which enables cross-coupling to be considered internally.

The open loop responses of Figure 3.4 and Figure 3.6 indicate the significant degree of cross-coupling between the yaw and roll channels. The design of a suitable autopilot or autopilot subsystem should therefore accommodate the cross coupled motions and effectively compensate for them.

5.4.1 Non-Interacting Multivariable Control System Design

One means of reducing cross-coupling between the AUV yaw and roll channels is to design decoupling elements (Doebelin (1985)). Given transfer functions for the interactions between controlled outputs, decoupling elements can be designed to cancel out the interaction terms. Subsequently, SISO controllers can be designed for each individual degree of freedom. However, this approach assumes that good approximations can be found to these interactions. Thus exact cancellation of the interactions cannot be expected or achieved in real systems. Additionally, implementation of the decoupling elements can lead to problems as these transfer function expressions are typically of high order and may be unrealizeable.

Figure 5.3 illustrates the block diagram representation of the non-interacting design approach whereby a 2 input - 2 output system is shown. Each controller is specifically designed in order that the interactions within the process are exactly cancelled.

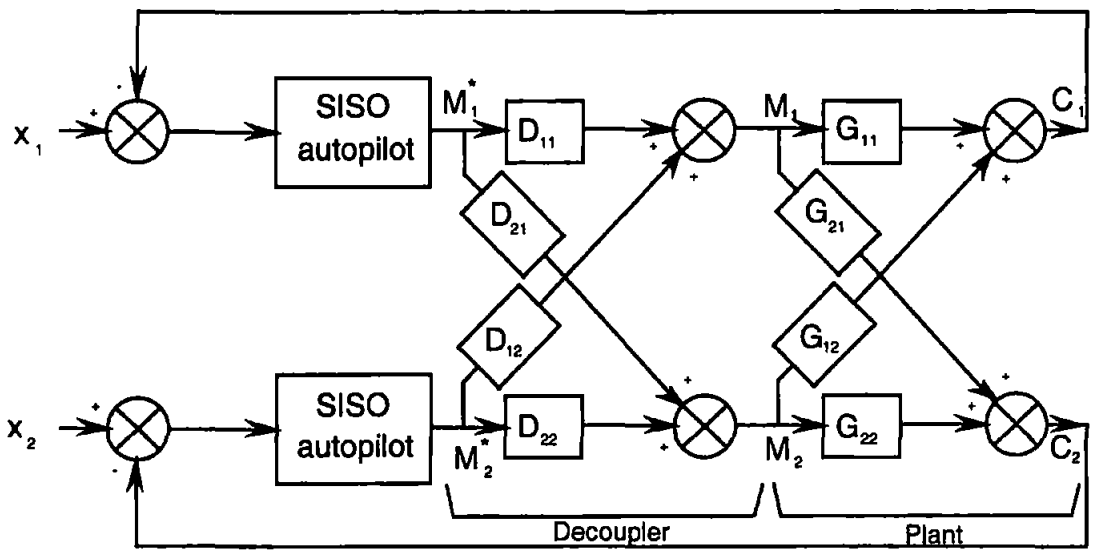


Figure 5.3: Non-Interacting control system design (after Doebelin (1985)).

Based upon this approach, decoupling elements were designed for the transfer functions relating yaw to roll and roll to yaw cross-coupling (elements G_{12} and G_{21} in Figure 5.3). This work is reproduced in Appendix E for brevity and assumed that D_{11} and D_{22} are zero. This required the approximation of the roll to canard and yaw to stern hydroplane transfer functions, as shown in Figures 5.4 and 5.5; these approximations are represented by the respective transfer functions:

$$G_{21} = \frac{\phi(s)}{\delta_{canard}(s)} = \frac{9}{s(s+1)} \tag{5.3}$$

$$G_{12} = \frac{\psi(s)}{\delta_{stern\_hy}(s)} = \frac{0.229}{0.1s^4 + s^3 + 4s^2 + 5s + 1} \tag{5.4}$$

where G_{21} is the roll to yaw coupling element and G_{12} is the yaw to roll coupling element. These are considered with the transfer function approximations to the roll and yaw dynamics, recall Eqn.(3.12) and Eqn.(3.13)

$$G_{22} = \frac{20}{s(s+1)} \tag{5.5}$$

$$G_{11} = \frac{33.518}{s(1.45s+1)}. \tag{5.6}$$

These approximations are depicted in Figures 5.6 to 5.7. Figure 5.5 clearly indicates the insignificant degree to which the stern hydroplanes perturb the open loop yaw dynamics. However, the cross-coupling motion exhibited in the roll channel as a product of canard actuator movement (Figure 5.4) is of comparatively large magnitude. Based upon these transfer function approximations, the following decoupling elements were obtained:

$$D_{21} = \frac{-9}{20} \tag{5.7}$$

$$D_{12} = \frac{-0.0068\,s(1.45s+1)}{(0.1s^4+s^3+4s^2+5s+1)} \tag{5.8}$$

leading to decoupled loop transfer functions as follows:

$$\frac{C_1}{M_1^*} = \frac{33.518(0.1s^4+s^3+4s^2+5s+1)-0.1031\,s(1.45s+1)}{s(1.45s+1)(0.1s^4+s^3+4s^2+5s+1)} \tag{5.9}$$

$$\frac{C_2}{M_2^*} = \frac{20(0.1s^4+s^3+4s^2+5s+1)-0.0615\,s(1.45s+1)(s+1)}{s(s+1)(0.1s^4+s^3+4s^2+5s+1)} \tag{5.10}$$

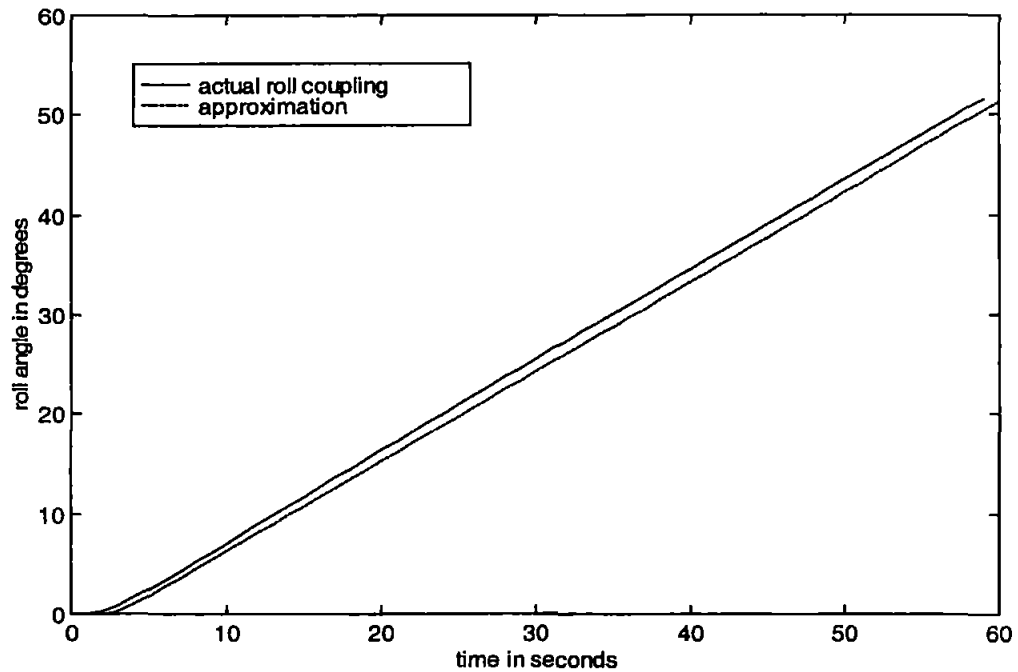


Figure 5.4: Roll cross-coupling with respect to a canard rudder step of 10°, and the approximation of Eqn.(5.3).

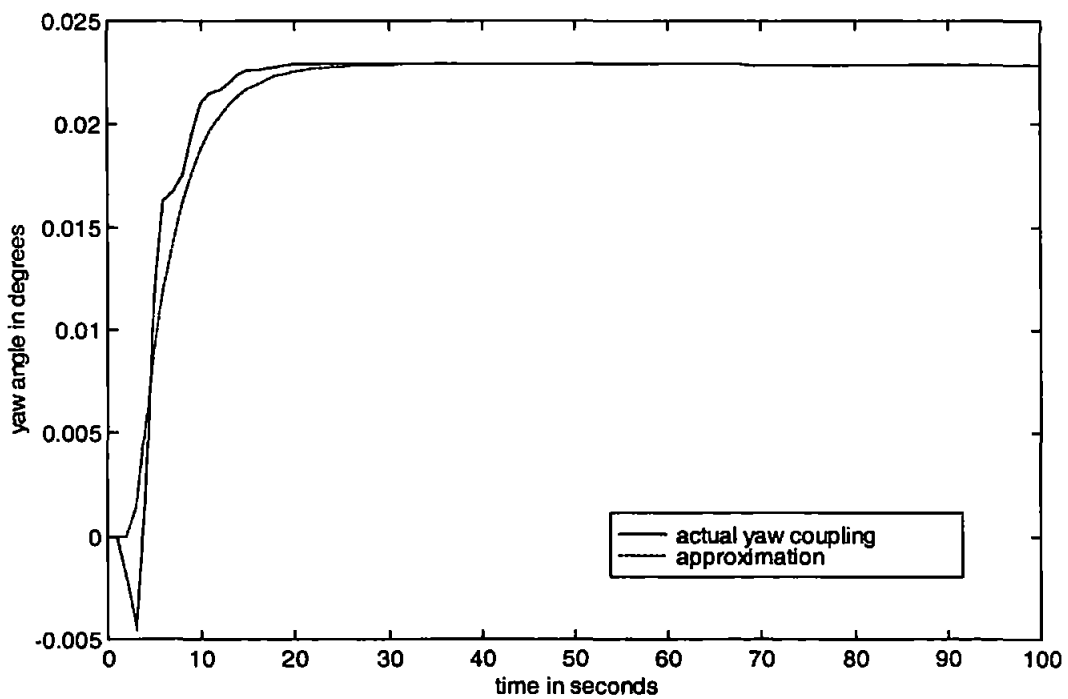


Figure 5.5: Yaw cross-coupling with respect to a stern hydroplane step of 10°, and the approximation of Eqn.(5.4).

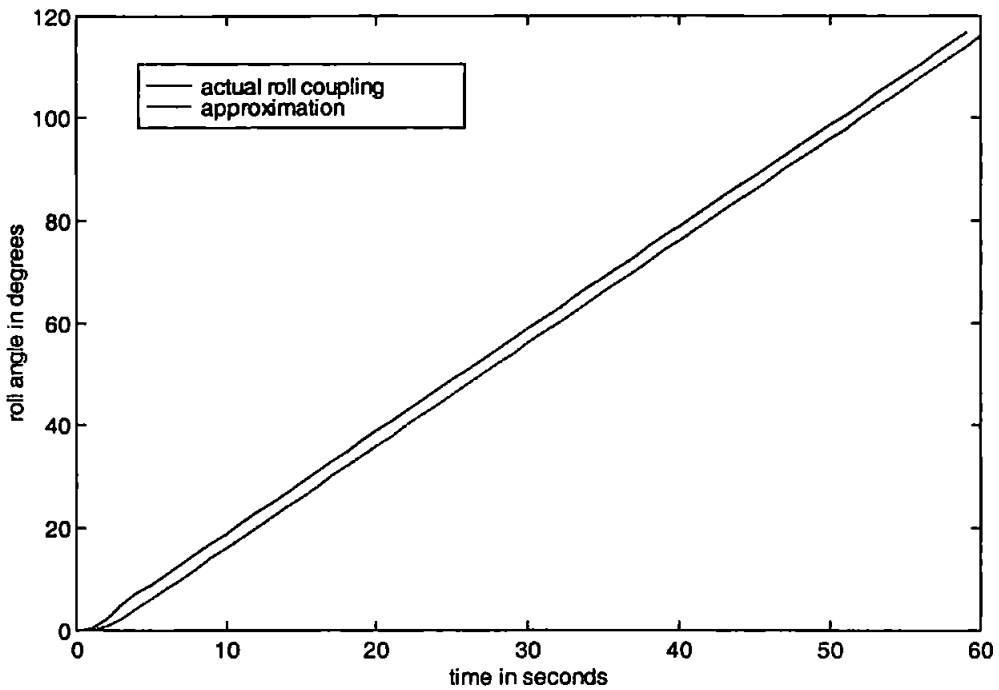


Figure 5.6: Roll cross-coupling with respect to a stern hydroplane step of 10°, and the approximation of Eqn.(5.5).

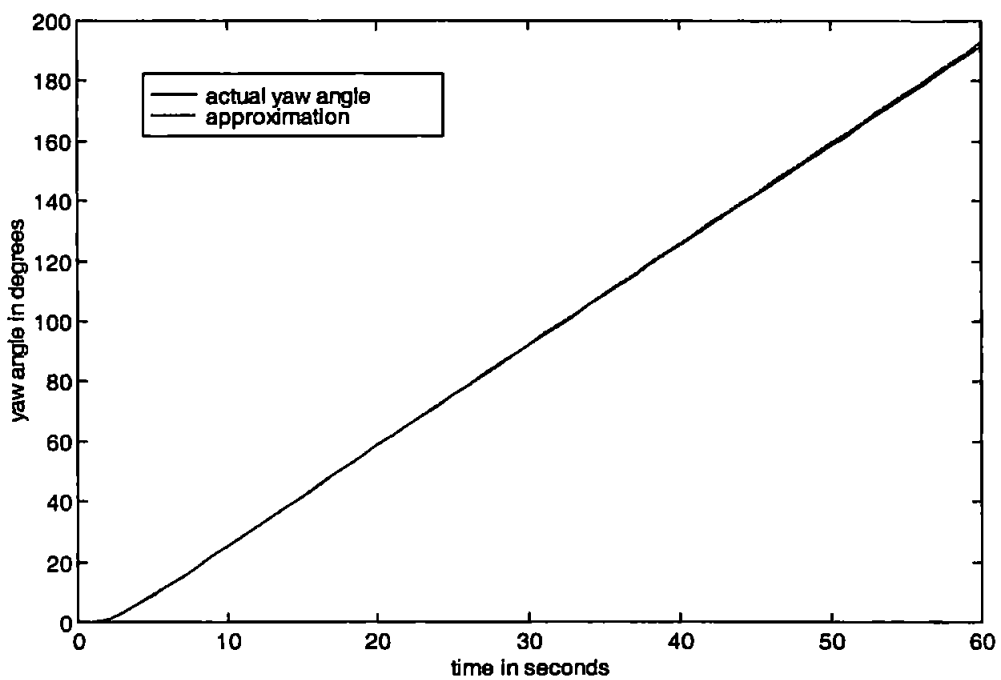


Figure 5.7: Yaw cross-coupling with respect to a canard rudder step of 10°, and the approximation of Eqn.(5.6).

which both represent proper, type 1 transfer functions.

Examination of the Nichols plots corresponding to the open loop approximations to the yaw and roll dynamics (Eqn.(5.6) and Eqn.(5.5)) suggested that the design of phase lead compensators for yaw and roll control would introduce appropriate phase advance yet retain moderate bandwidth to allow suppression of high frequency noise. Upon introduction of suitable phase lead, the course-changing PD autopilot of Eqn.(3.14) is thus implemented as the yaw controller (Eqn.(5.11))

$$G_{c,\psi}(s)=\frac{1+1.204s}{1+0.708s} \tag{5.11}$$

and the roll autopilot was taken as the following phase lead network:

$$G_{c,\phi}(s)=\frac{1+0.735s}{1+0.489s}. \tag{5.12}$$

The course-changing and roll regulating responses of the AUV when employing these autopilots, in comparison to the interacting form of this strategy are given in Figure 5.8. Additionally, the low canard rudder and stern hydroplane responses are included in Figure 5.9 for completeness.

The non-interacting design clearly eliminates a little cross-coupling between the yaw and roll channels. The yaw induced roll angle however, is not significantly improved through the use of this technique. The main benefit of this approach is the reduced degree of canard control effort involved in achieving the responses of Figure 5.8. Indeed, the cost of this minor reduction in roll cross-coupling is a slightly faster yaw response with reduced damping. Clearly, the implementation of the decoupling elements may prove difficult in reality, due to their high complexity. Notwithstanding, this approach can be very effective if more accurate transfer function representations of the yaw to roll and roll to yaw dynamics are available.

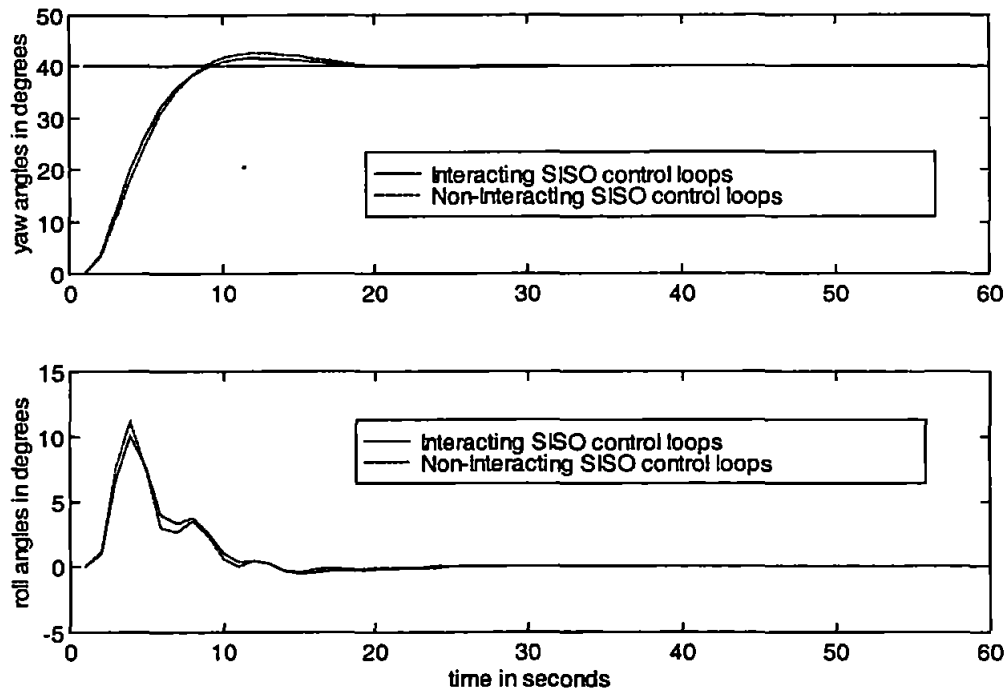


Figure 5.8: Yaw and roll responses for the AUV employing the decoupling elements G_{12} and G_{21}.

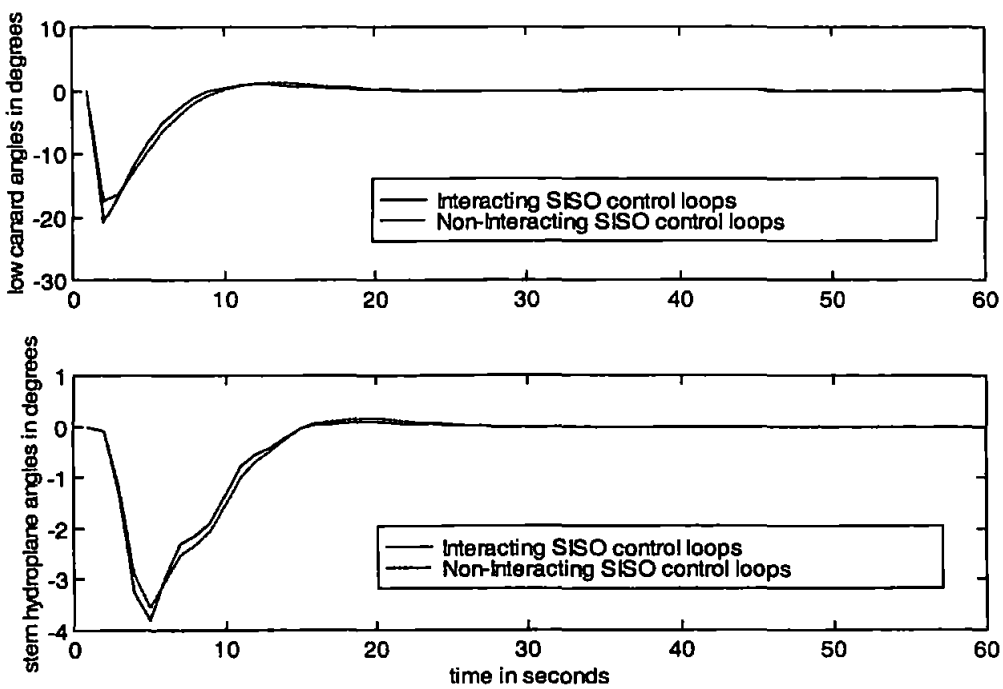


Figure 5.9: Low canard and stern hydroplane responses for the AUV employing the decoupling elements G_{12} and G_{21}.

5.4.2 Applying ANFIS to Simultaneously Control the Yaw and Roll Channels

The ANFIS architecture was employed to encode and tune a 9 rule TSK fuzzy autopilot for roll regulatory control (Eqn.(5.13)). This autopilot was subsequently used in concert with the ANFIS tuned course-changing autopilot of Eqn.(4.39). Clearly, the application of this tuning regime ignores interactions that may exist between control loops, as each autopilot is considered individually. Consequently, degradation in overall control system performance should be anticipated, when employing ANFIS models to control multiple, coupled AUV degrees of freedom. This is indeed seen to be the case.

The roll-regulating autopilot was taken after 300.5 epochs of tuning as:

$$
\begin{aligned}
&\text{If } \phi_\varepsilon \text{ is N and } \dot{\phi} \text{ is N then } \delta_\rho = -0.486\phi_\varepsilon - 0.879\dot{\phi} - 0.029\\
&\text{If } \phi_\varepsilon \text{ is N and } \dot{\phi} \text{ is Z then } \delta_\rho = -0.489\phi_\varepsilon - 0.902\dot{\phi} + 0.001\\
&\text{If } \phi_\varepsilon \text{ is N and } \dot{\phi} \text{ is P then } \delta_\rho = -0.486\phi_\varepsilon - 0.896\dot{\phi} + 0.003\\
&\text{If } \phi_\varepsilon \text{ is Z and } \dot{\phi} \text{ is N then } \delta_\rho = -0.299\phi_\varepsilon + 0.703\dot{\phi} - 0.123\\
&\text{If } \phi_\varepsilon \text{ is Z and } \dot{\phi} \text{ is Z then } \delta_\rho = -0.488\phi_\varepsilon - 0.891\dot{\phi} + 0.004\\
&\text{If } \phi_\varepsilon \text{ is Z and } \dot{\phi} \text{ is P then } \delta_\rho = -0.305\phi_\varepsilon - 0.306\dot{\phi} - 0.037\\
&\text{If } \phi_\varepsilon \text{ is P and } \dot{\phi} \text{ is N then } \delta_\rho = -0.590\phi_\varepsilon - 0.839\dot{\phi} - 0.117\\
&\text{If } \phi_\varepsilon \text{ is P and } \dot{\phi} \text{ is Z then } \delta_\rho = -0.481\phi_\varepsilon - 1.081\dot{\phi} - 0.061\\
&\text{If } \phi_\varepsilon \text{ is P and } \dot{\phi} \text{ is P then } \delta_\rho = -0.659\phi_\varepsilon - 1.311\dot{\phi} + 0.781
\end{aligned}
\qquad (5.13)
$$

The evolution of the fuzzy sets during this learning phase was as depicted in Figure 5.10. Full details pertaining to the parameter set of this autopilot are given in Appendix F. Figure 5.11 depicts the course-changing and roll-regulating responses of the AUV when employing the SISO ANFIS autopilots of Eqn.(4.39) and Eqn.(5.13) simultaneously. Additionally, Figure 5.12 shows the corresponding low canard rudder and stern hydroplane responses. The autopilots have been de-tuned herein to allow comparative assessment with the non-interacting controllers of section 5.4.1. Table 5.1

provides a comparison of the AUV autopilot strategies developed within sections 5.4.1 and 5.4.2.

This fuzzy SISO approach to multivariable yaw and roll control reduces the roll cross-coupling by 0.9° to 9.2° for a comparable course-changing response. Although the roll angle cross-coupling is actually reduced through employing this strategy, the overall roll motion ITSE value is excessive if side scan sonar packages are to be deployed effectively (Zehner and Thompson (1996)).

As aforementioned, each autopilot has been tuned with no comprehension of the others behaviour, and thus neither autopilot makes provision for the interaction or conflicting interests of the other. The following section discusses the implementation of a novel structure for multivariable autopilot designs which yields improved roll cross-coupling whilst performing course-changing manoeuvres.

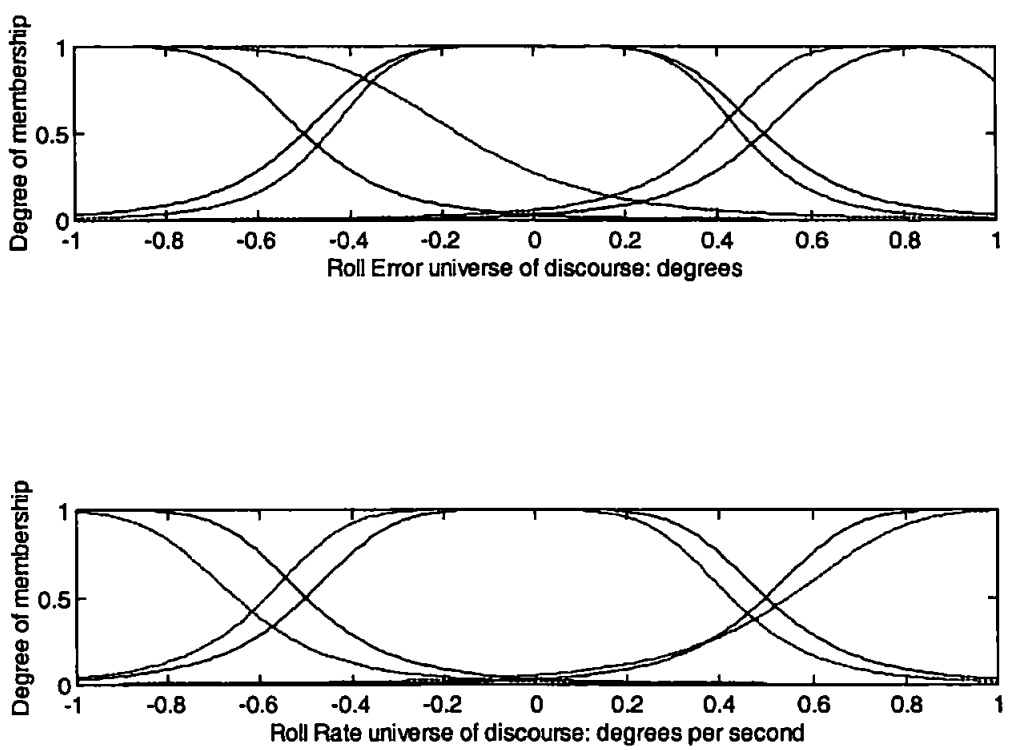


Figure 5.10: The evolution of the fuzzy sets for the roll regulating autopilot during learning. (Dashed lines show the tuned set positions.)

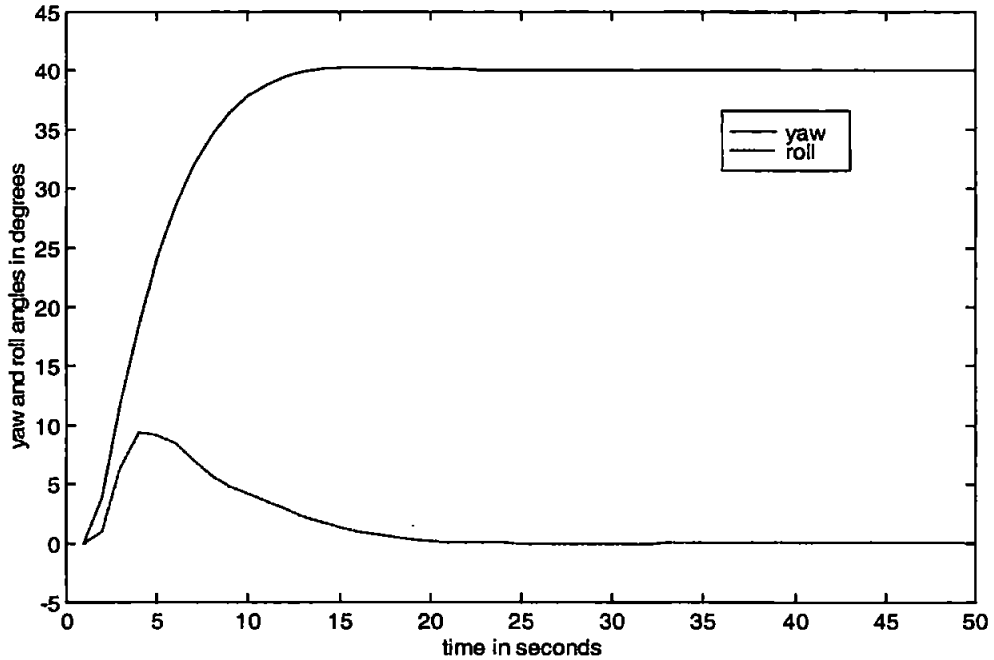


Figure 5.11: Yaw and roll responses of the AUV when employing the individual course-changing and roll-regulating 9 rule TSK autopipots at 7.5-knots for a 40° course-changing manoeuvre.

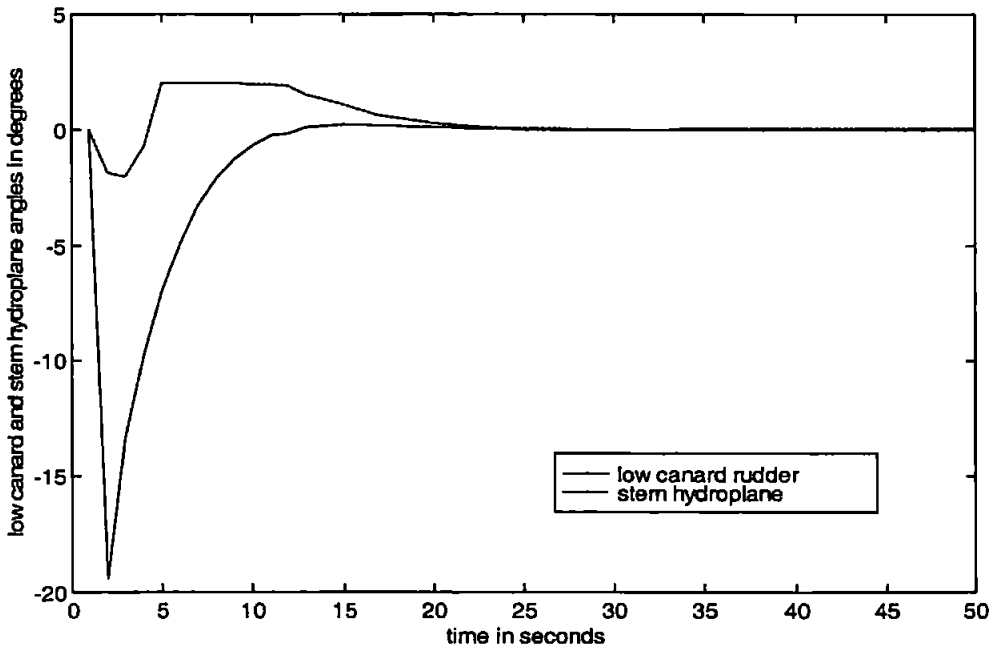


Figure 5.12: Low canard rudder and stern hydroplane responses of the AUV when employing the individual course-changing and roll-regulating 9 rule TSK autopilots at 7.5 knots for a 40° course-changing manoeuvre.

	Performance Indices – Yaw and Roll MISO Autopilots							
Autopilot strategy	ψ_ε $(^\circ)^2$	δ_ε $(^\circ)^2$	ϕ_ε $(^\circ)^2$	δ_s $(^\circ)^2$	T_R secs	$M_p(t)$ %	sse %	Max ϕ
PD Yaw/Roll strategy	3606.20	1127.50	272.79	52.90	8.8	4.1	0.0	11.2
Non-Interacting strategy	3886.40	1041.00	254.02	52.46	8.6	6.6	0.0	10.1
ANFIS Yaw/ Roll strategy	3045.50	1451.70	575.80	44.22	7.65	1.75	0.0	15.8
De-tuned ANFIS Strategy	3884.30	955.16	446.58	58.77	12.21	0.83	0.0	9.2

Table 5.1: Performance comparisons of yaw and roll control autopilot strategies developed within sections 5.4.1 and 5.4.2.

5.4.3 CANFIS Control of the AUV Yaw and Roll Channels

In order to arrive at a fuzzy design solution that accommodates the interaction amongst plant degrees of freedom, it is necessary to tune the premise and consequent functions for each output variable collectively. By employing the architecture of Figure 5.2 to encode a multivariable TSK style fuzzy autopilot as an adaptive neural network, the interaction between AUV yaw and roll channels (or indeed any other channels) may be considered during the learning phase. In essence the cost function minimized during tuning exhibits components from each channel. Consequently gradient descent using this cost function leads to an autopilot parameter set which satisfies the simultaneous requirements of multiple degrees of freedom.

Initially, to avoid problems associated with the curse of dimensionality, the number of membership functions considered within each input universe of discourse was taken to be 2, a negative and a positive fuzzy set. Thus, a 4 input - 2 output structure containing 16 (2^4) rules was developed. The number of parameters available for tuning within the

architecture were 184 (24 non-linear premise and 160 linear consequent parameters). Clearly, the selection of the optimal parameter set involves an extensive search if using a random search approach. However, by ensuring that the matrix containing the training data set is of suitable rank and is non-singular, it is possible to adapt the parameters with the hybrid learning algorithm to obtain a parameter solution set which is locally tuned.

Training data were collected based on the AUV simulated at 7.5-knots, and the 16 rule autopilot structure was tuned for 300.5 epochs using the aforementioned hybrid learning rule.

The fuzzy sets pertaining to the resulting multivariable autopilot (dashed-dot line) are plotted with the original sets (bold line) in Figure 5.13. Clearly some of the CANFIS fitting to the function relating the input to the output variables has been encompassed within the premise parameters of Figure 5.13. An abbreviated form of the tuned rule base (which is reproduced in Appendix G) is given below in Eqn.(5.14):

$$
\begin{aligned}
&\text{If } \psi_\varepsilon \text{ is N and } \dot{\psi} \text{ and } \phi_\varepsilon \text{ is N and } \dot{\phi} \text{ is N} \\
&\text{then } \delta_\psi = -0.094\psi_\varepsilon + 0.012\dot{\psi} + 0.092\phi_\varepsilon + 0.010\dot{\phi} - 0.945 \\
&\text{and } \delta_\phi = 0.029\psi_\varepsilon + 0.063\dot{\psi} - 0.028\phi_\varepsilon + 0.031\dot{\phi} - 1.121
\end{aligned}
$$

$$
\begin{aligned}
&\text{If } \psi_\varepsilon \text{ is N and } \dot{\psi} \text{ is N and } \phi_\varepsilon \text{ is N and } \dot{\phi} \text{ is P} \\
&\text{then } \delta_\psi = -0.098\psi_\varepsilon - 0.037\dot{\psi} - 0.073\phi_\varepsilon + 0.006\dot{\phi} + 0.025 \\
&\text{and } \delta_\phi = -0.033\psi_\varepsilon + 0.062\dot{\psi} + 0.041\phi_\varepsilon + 0.029\dot{\phi} - 0.014
\end{aligned}
$$

$$
\begin{array}{cccccccc}
. & . & . & . & . & . & . & . \\
. & . & . & . & . & . & . & . \\
. & . & . & . & . & . & . & .
\end{array}
\qquad (5.14)
$$

$$\text{If } \psi_\varepsilon \text{ is P and } \dot{\psi} \text{ is P and } \phi_\varepsilon \text{ is P and } \dot{\phi} \text{ is N}$$
$$\text{then } \delta_\psi = 0.071\psi_\varepsilon - 0.075\dot{\psi} - 0.005\phi_\varepsilon + 0.039\dot{\phi} - 0.030$$
$$\text{and } \delta_\phi = 0.043\psi_\varepsilon + 0.069\dot{\psi} + 0.074\phi_\varepsilon + 0.072\dot{\phi} - 0.074$$

$$\text{If } \psi_\varepsilon \text{ is P and } \dot{\psi} \text{ is P and } \phi_\varepsilon \text{ is P and } \dot{\phi} \text{ is P}$$
$$\text{then } \delta_\psi = 0.074\psi_\varepsilon + 0.029\dot{\psi} + 0.078\phi_\varepsilon + 0.022\dot{\phi} + 1.075$$
$$\text{and } \delta_\phi = -0.055\psi_\varepsilon - 0.045\dot{\psi} + 0.024\phi_\varepsilon - 0.060\dot{\phi} + 0.945$$

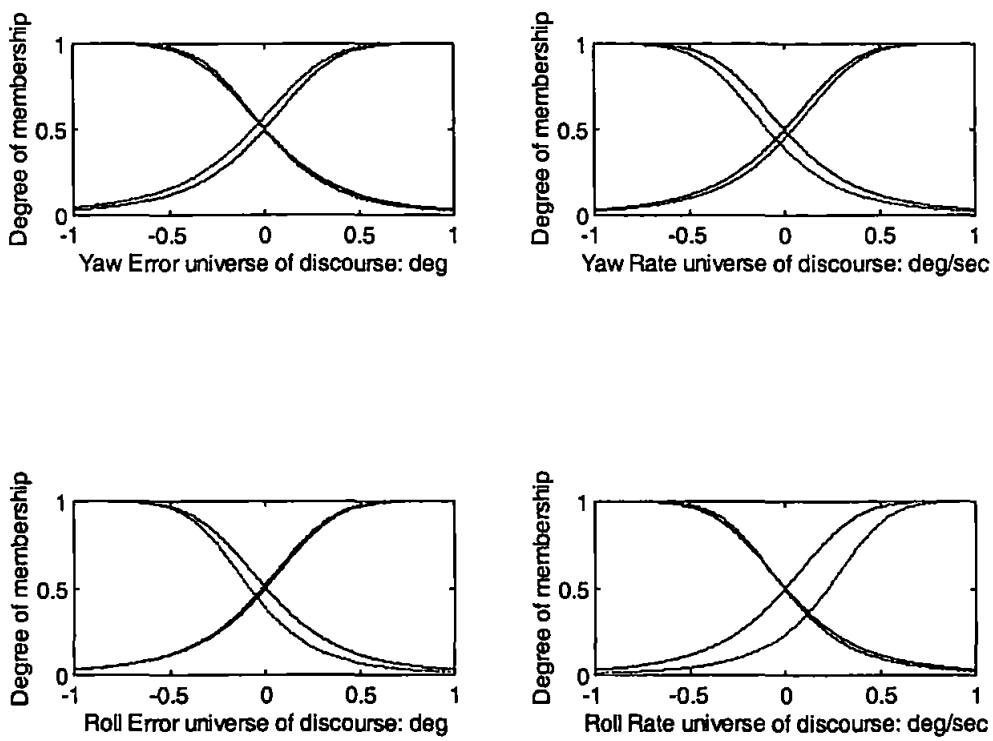


Figure 5.13: The original and tuned fuzzy sets for the 16 rule multivariable autopilot. (Dashed lines show the new set positions.)

An assessment of autopilot responsiveness was provided by the AUV model's reaction to a course change of 40° during which roll motion of zero magnitude was demanded. The responses of the AUV when employing the tuned 16 rule multivariable autopilot

for a 40° course-changing manoeuvre with roll regulation are illustrated in Figure 5.14; the corresponding control effort responses being given in Figure 5.15.

Clearly, the yaw induced roll motion is greatly reduced by employing the multivariable autopilot over the SISO control strategies of sections 5.4.1 and 5.4.2. The maximum roll angle for the multivariable strategy is 6.8° as opposed to 10.1° when employing non-interacting SISO control loops. Figure 5.15 highlights that the stern hydroplane effort employed in attaining this reduced roll cross-coupled motion is less oscillatory when using the multivariable autopilot, the responses being crisper than those of the

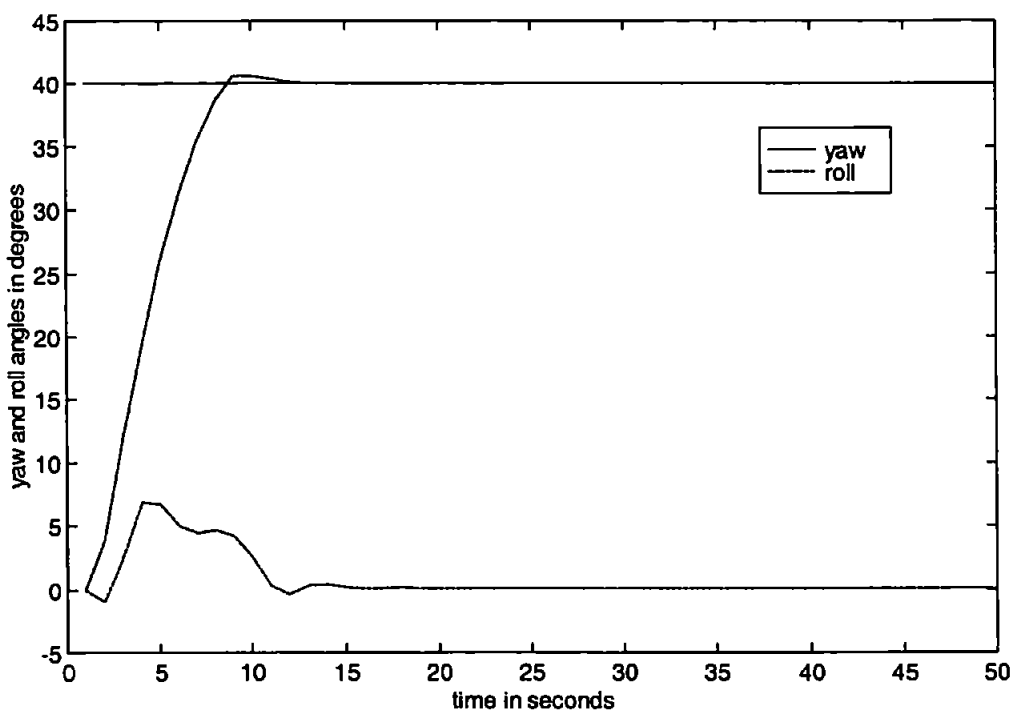


Figure 5.14: Yaw and roll responses of the AUV when employing the 16 rule multivariable fuzzy autopilot at 7.5-knots for a 40° course-changing manoeuvre.

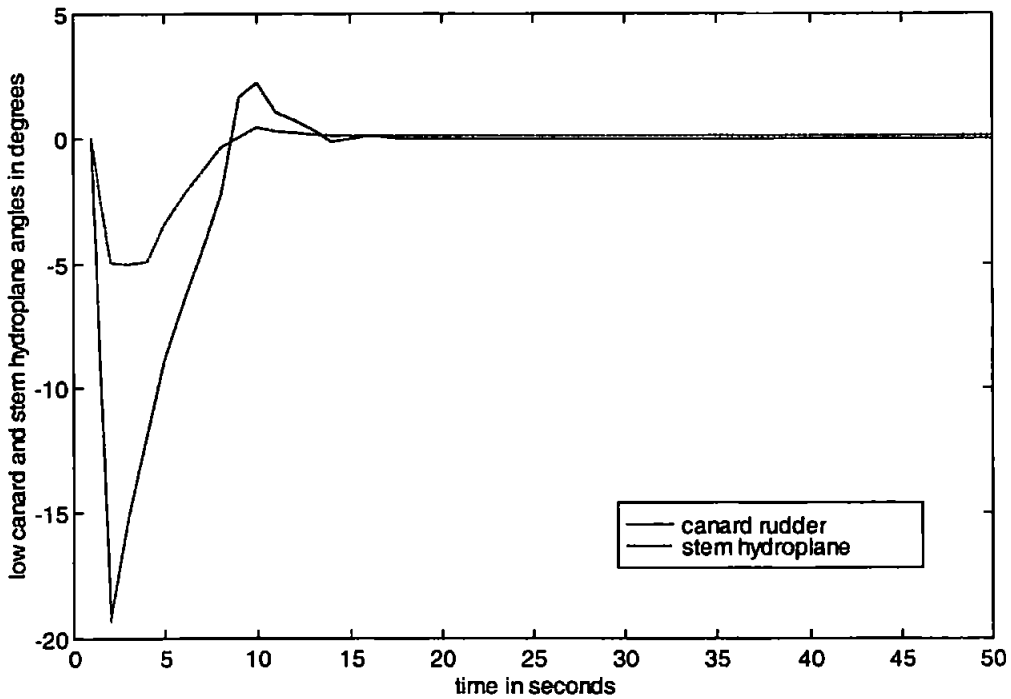


Figure 5.15: Low canard rudder and stern hydroplane responses of the AUV when employing the 16 rule multivariable autopilot at 7.5-knots.

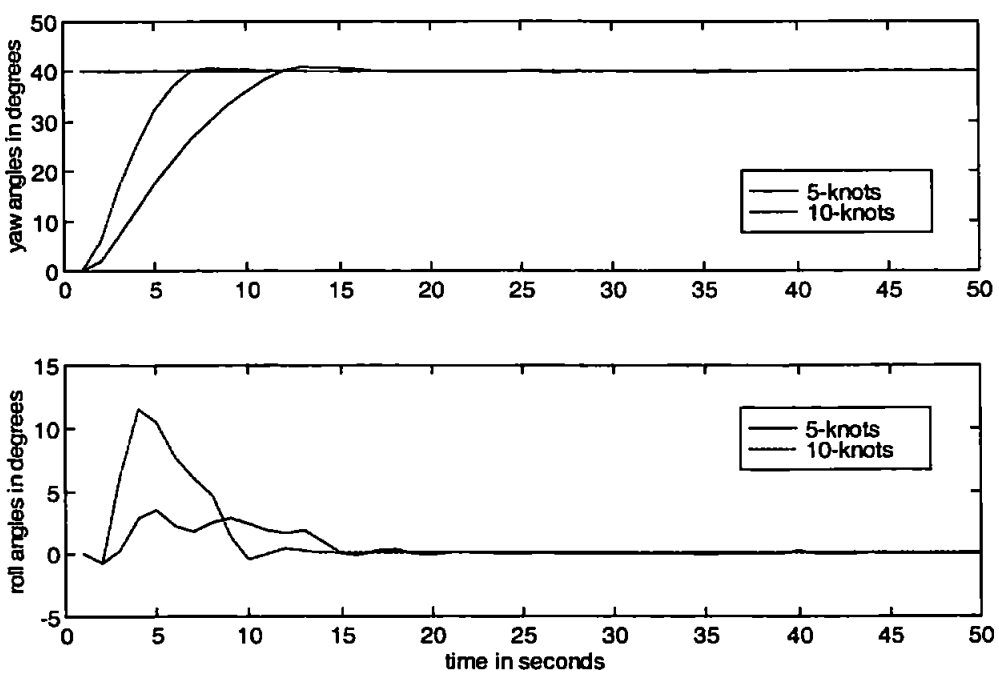


Figure 5.16: Yaw and roll responses of the AUV when employing the 16 rule multivariable fuzzy autopilot at 5 and10-knots.

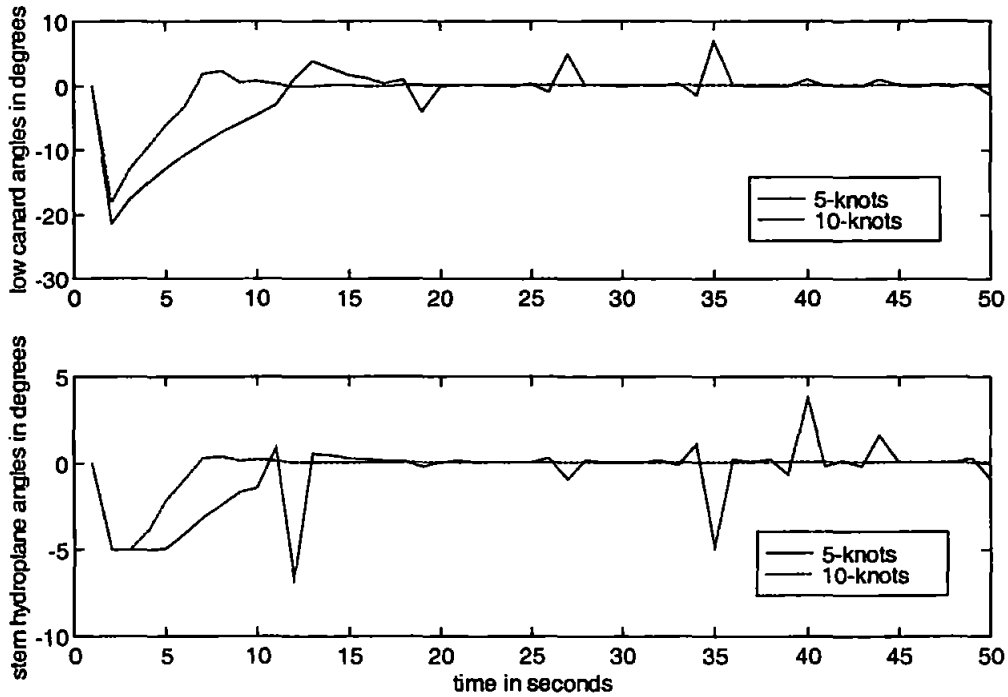


Figure 5.17: Low canard rudder and stern hydroplane responses of the AUV when employing the 16 rule multivariable autopilot at 5 and 10-knots.

non-interacting SISO control loops of Figure 5.9. Examination of the low canard rudder response provides an insight into the course-changing accuracy achieved when employing the CANFIS style autopilot. Whilst this canard response is more pronounced around the nominal 0° of effort, the sharp transition through this point explains the crisp accurate course change of Figure 5.14.

Table 5.2 presents the comparative performance of the non-interacting autopilot strategy of section 5.4.1 against the CANFIS 16 rule multivariable fuzzy autopilot of section 5.4.3. The multivariable autopilot strategy is clearly superior to the non-interacting strategy at the design speed of 7.5-knots. Course-changing response times for each method are comparable, yet the CANFIS technique displays a superior accuracy in terms of vastly reduced course overshoot. Additionally, the roll regulating

characteristics of the CANFIS strategy are superior to those of the classical approach. Notwithstanding, the classical non-interacting design produces lower demands in terms of stern hydroplane control effort. The stern hydroplane response of the CANFIS autopilot is not saturated, peaking at a maximum induced angle of 4.8°. Further verification of the CANFIS autopilot is clearly required in order to corroborate the conclusions drawn so far.

To assess the robustness qualities of the resulting CANFIS autopilot to forward speed variations, the AUV model was simulated at both 5 and 10-knots. The AUV responses of Figures 5.16 and 5.17 illustrate the excellent robustness qualities of the multivariable autopilot at 10-knots. However, at 5-knots the stern hydroplane control effort is oscillatory. This is in contrast to the yaw and roll responses pertaining to this control effort, which are smooth.

	Performance Indices – Yaw and Roll Autopilots							
Autopilot strategy	ψ_e $(^{\circ})^2$	δ_e $(^{\circ})^2$	ϕ_e $(^{\circ})^2$	δ_s $(^{\circ})^2$	T_R secs	$M_p(t)$ %	sse %	Max ϕ
Non-Interacting strategy	3886.40	1041.00	254.02	52.46	8.6	6.6	0.0	10.1
16 rule CANFIS autopilot	3637.50	1093.40	192.15	100.80	8.5	1.69	0.0	6.8

Table 5.2: Performance comparisons for non-interacting and 16 rule CANFIS yaw and roll control autopilot strategies at 7.5-knots.

In order that the robustness of the developed CANFIS autopilot be more comprehensively examined, the following section details the performance of this autopilot strategy in light of parameter variations and sea current disturbances.

5.5 Robustness Testing for the Yaw and Roll Autopilot

Comprehensive testing involving variations of the relevant hydrodynamic coefficients was performed for the CANFIS 16 rule multivariable fuzzy autopilot. Line of sight (LOS) guidance in the presence of sea current disturbances also proved useful in examining the robustness of the developed autopilot in comparison to the ANFIS design approach. It should be noted at this juncture that the CANFIS autopilot was compared to the ANFIS technique, and not the non-interacting control strategy, to illustrate clearly the effects of tuning the fuzzy parameters collectively. The following two sections document the results pertaining to these simulations, respectively.

5.5.1 Vehicle Coefficient Variations

Figures 5.18 and 5.19 depict the yaw and roll, and low canard and stern hydroplane responses of the AUV respectively, when employing the CANFIS autopilot for the nominal AUV mass and when the mass of the vehicle is increased to 175% of its nominal value.

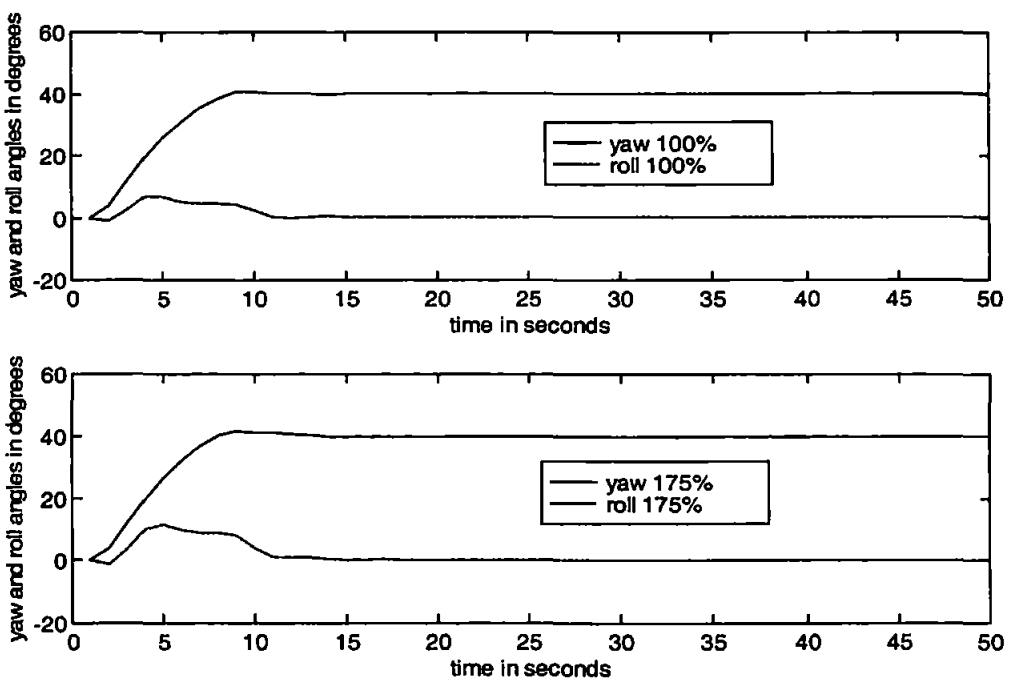


Figure 5.18: Mass variation during a 40° course-change when employing the CANFIS autopilot – yaw and roll responses.

Evidently, the CANFIS autopilot performs well in light of the increased AUV payload. Some evidence of increased overshoot is apparent as the overall system is now under-damped as compared to the nominal operating conditions. Additionally, the following perturbations in the hydrodynamic coefficients are to be considered:

- $\pm$ 20% variation in Y_{UV}, the sway damping coefficient
- $\pm$ 20% variation in Y_{UR}, the yaw into sway coefficient
- $\pm$ 20% variation in K_{UV}, the sway into roll coefficient
- $\pm$ 20% variation in K_{UR}, the yaw into sway coefficient
- $\pm$ 20% variation in K_{UP}, the roll damping coefficient
- $\pm$ 20% variation in N_{UV}, the sway into yaw coefficient
- $\pm$ 20% variation in N_{VR}, the roll into yaw coefficient
- $\pm$ 20% variation in N_{UR}, the yaw damping coefficient

As in Chapter 4, the hydrodynamic coefficients are considered as a function of the total velocity squared (Eqn.(4.42)) and are assumed to vary over a mission.

The responses of the AUV to a 40° course-changing manoeuvre when employing the CANFIS autopilot under these coefficient variations are illustrated in Figures 5.20 to 5.27.

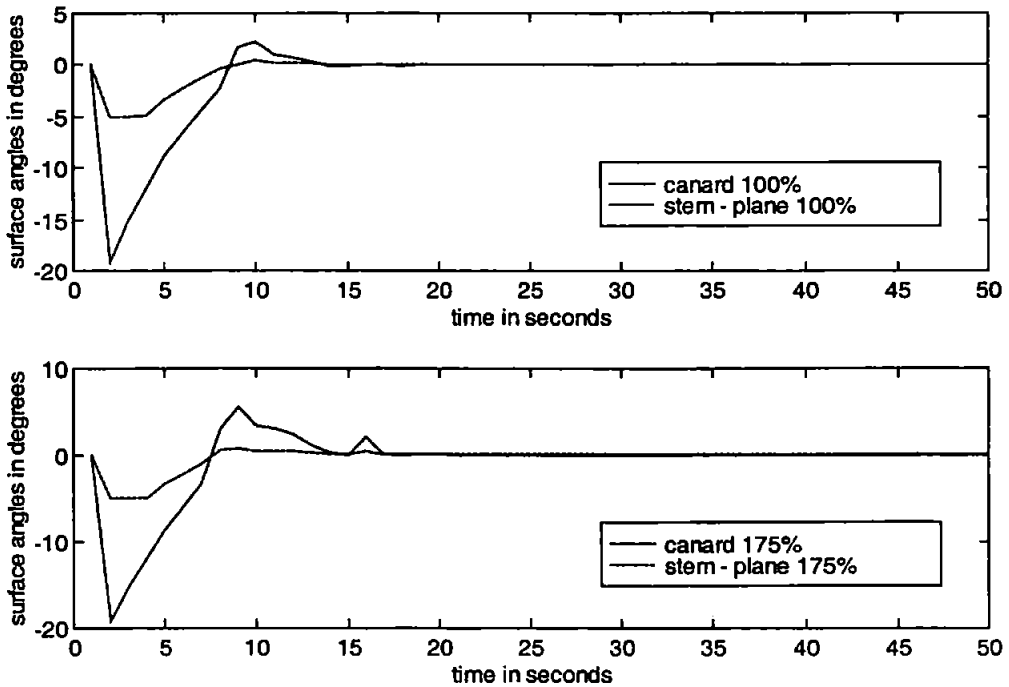


Figure 5.19: Mass variation during a 40° course-change when employing the multivariable CANFIS autopilot - low canard and stern hydroplane responses.

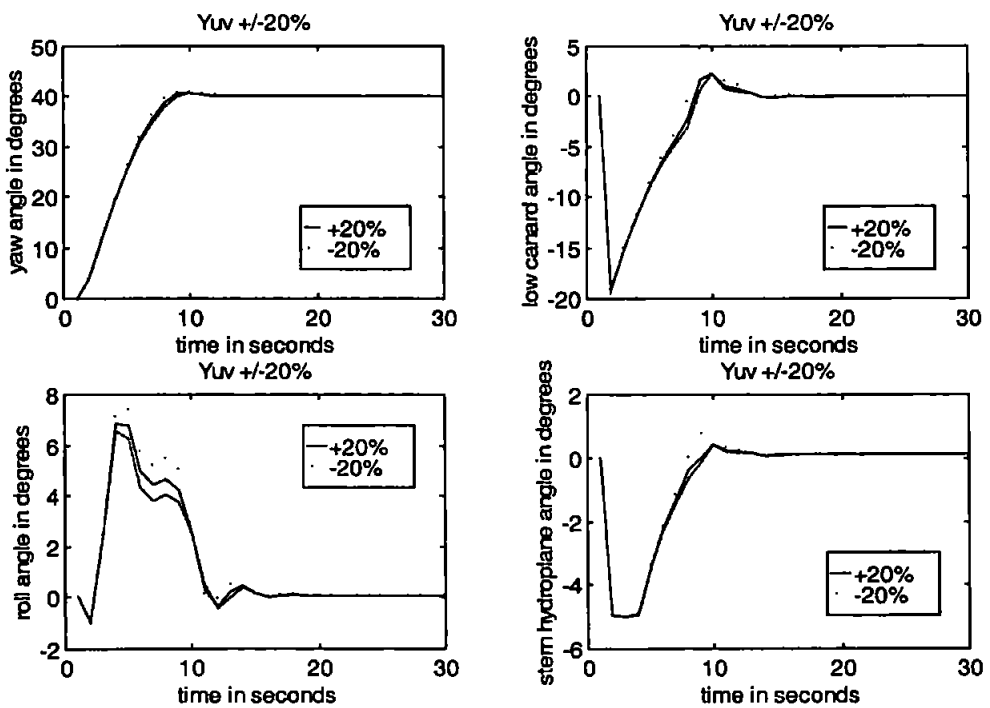


Figure 5.20: Varying Y_{uv} hydrodynamic coefficient during a 40° course-change when employing the multivariable CANFIS autopilot.

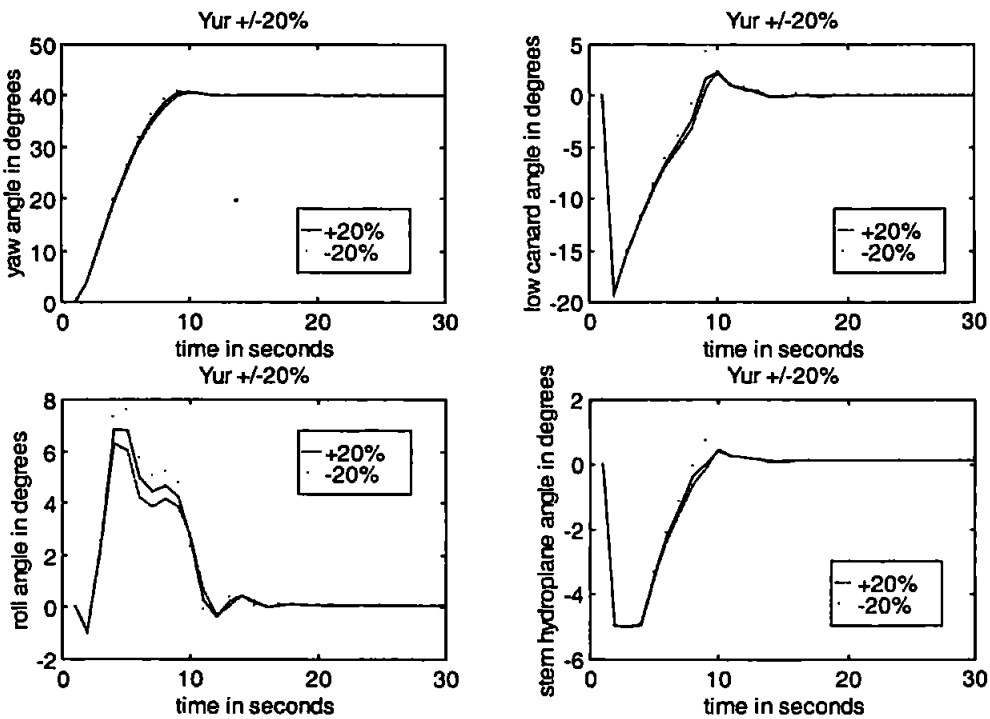


Figure 5.21: Varying Y_{ur} hydrodynamic coefficient during a 40° course-change when employing the multivariable CANFIS autopilot.

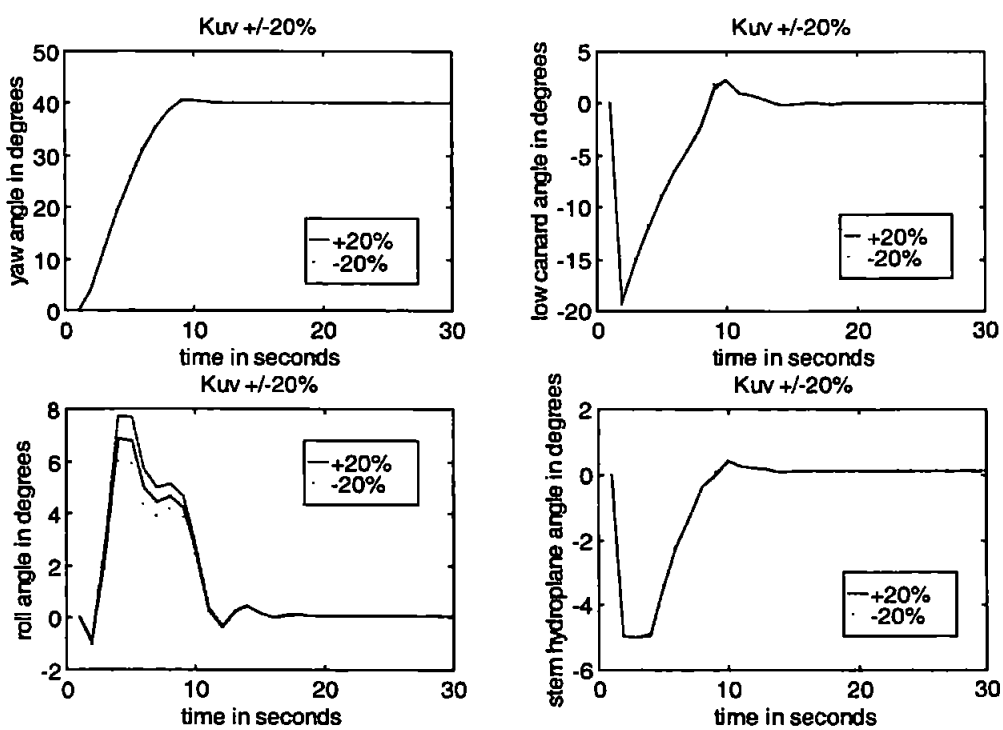


Figure 5.22: Varying K_{uv} hydrodynamic coefficient during a 40° course-change when employing the multivariable CANFIS autopilot.

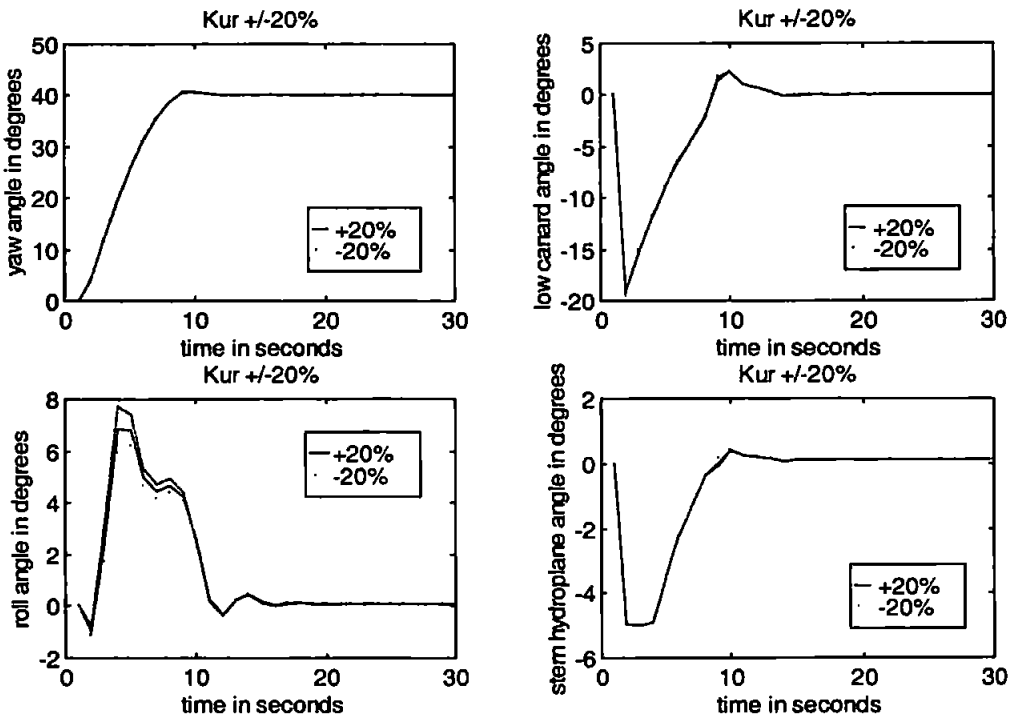


Figure 5.23: Varying K_{ur} hydrodynamic coefficient during a 40° course-change when employing the multivariable CANFIS autopilot.

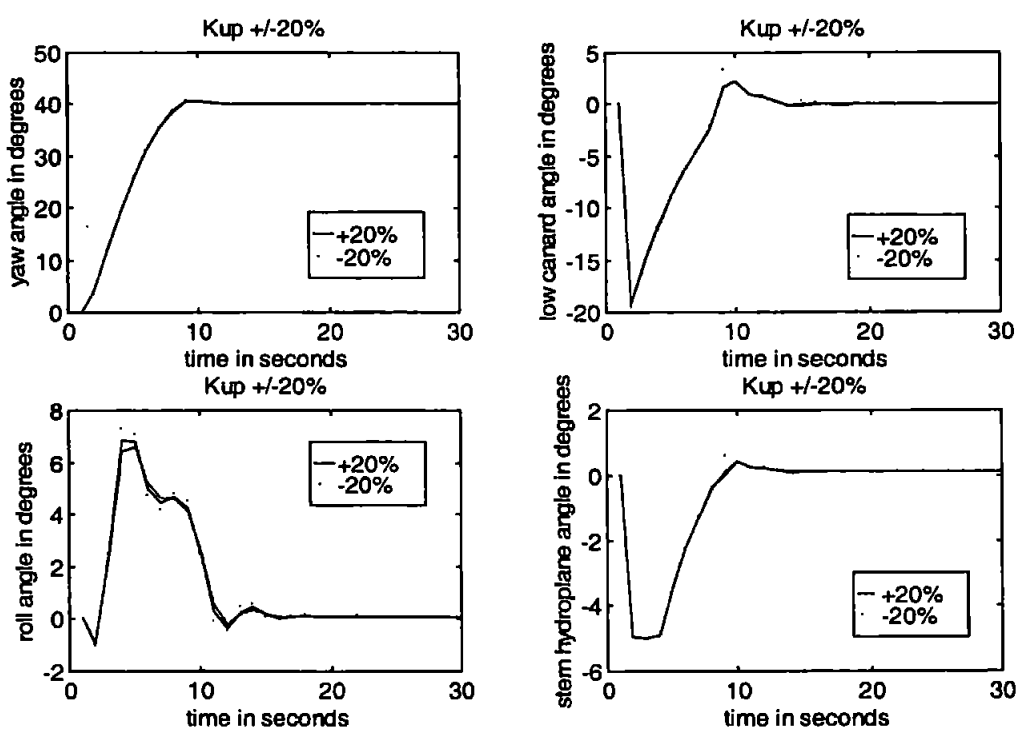


Figure 5.24: Varying K_{up} hydrodynamic coefficient during a 40° course-change when employing the multivariable CANFIS autopilot.

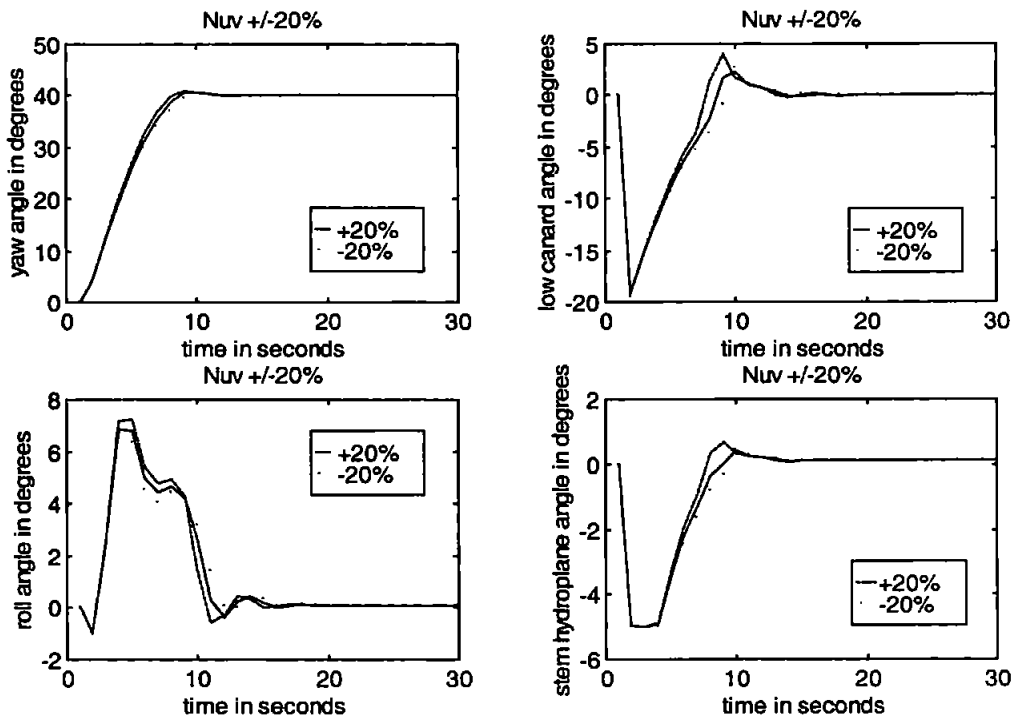


Figure 5.25: Varying N_{uv} hydrodynamic coefficient during a 40° course-change when employing the multivariable CANFIS autopilot.

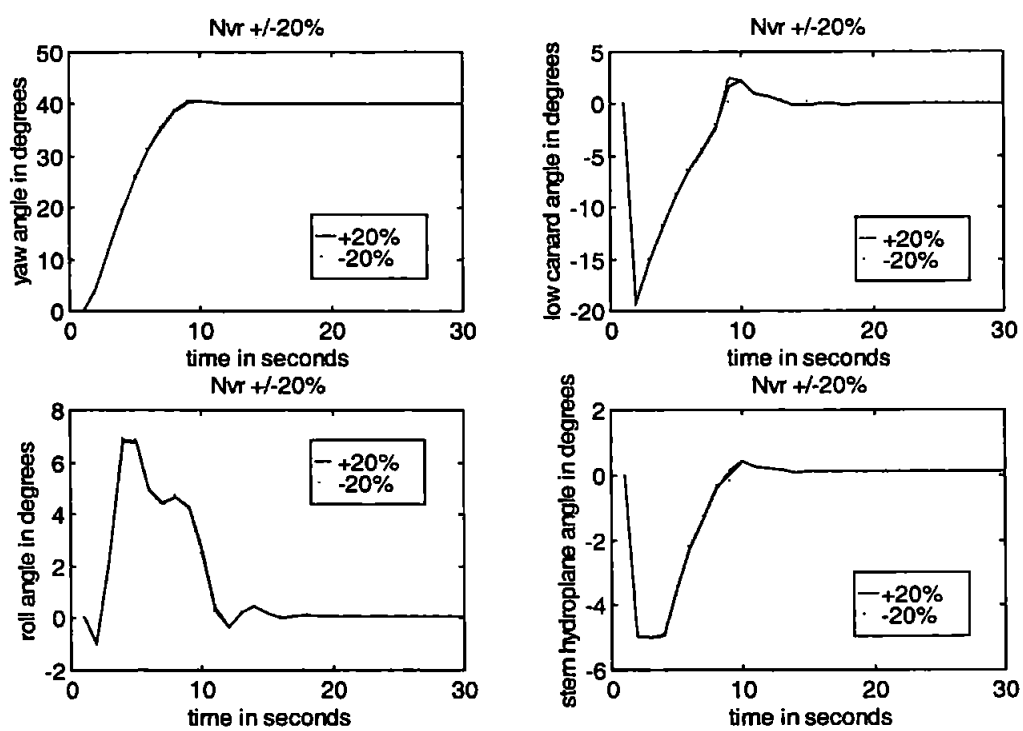


Figure 5.26: Varying N_{vr} hydrodynamic coefficient during a 40° course-change when employing the multivariable CANFIS autopilot.

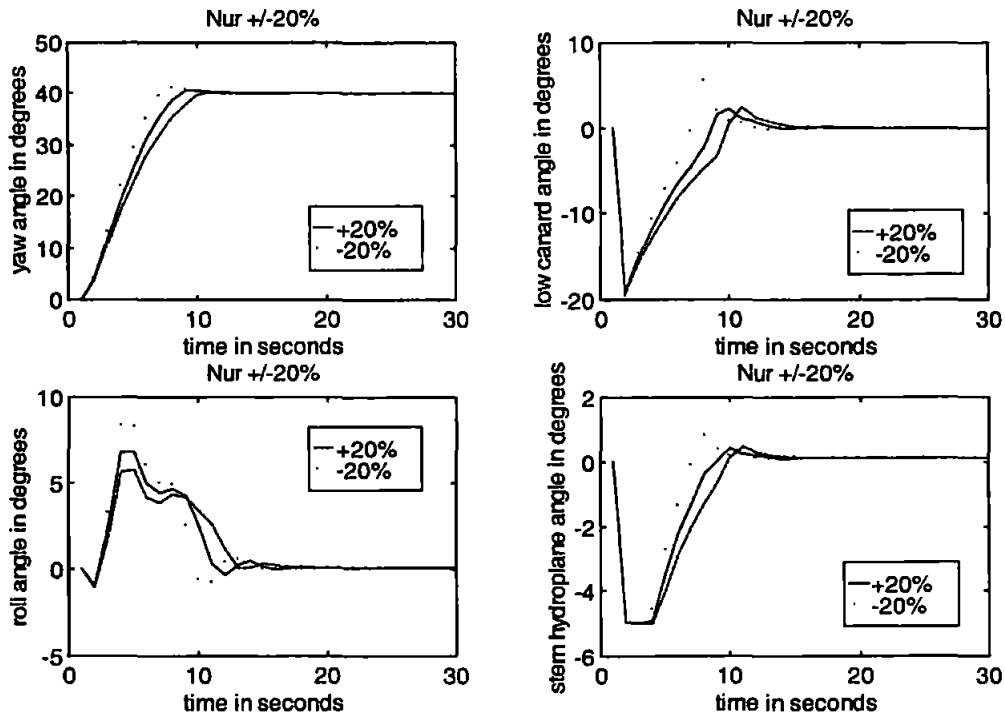


Figure 5.27: Varying N_{ur} hydrodynamic coefficient during a 40° course-change when employing the multivariable CANFIS autopilot.

The CANFIS autopilot strategy accommodates the parameter variations in a robust manner, the responses proving similarly damped at all three tested values (nominal and ± 20% variations).

5.5.2 Line of Sight Guidance

Autonomous guidance of the vehicle is again achieved by employing the LOS guidance algorithm. The verification track was employed to appraise the robustness properties of the CANFIS multivariable autopilot against the ANFIS autopilot strategy of Chapter 4. The radius of acceptance β was again fixed at 15 metres and the nominal vehicle speed at 7.5-knots. For ease of comparison, the ANFIS and CANFIS responses are presented collectively.

Simulating the AUV at 7.5-knots over the verification course in the presence of a Westerly current of 3 ms^{-1}, produced the course following responses of Figure 5.28. The AUV response when employing the CANFIS multivariable autopilot is more accurate and has better damping characteristics than that of the ANFIS autopilot response. Examination of the corresponding roll angular response (Figure 5.29) illustrates the reduced roll profile of the AUV over the course demand.

Applying a current of 2.5 ms^{-1} in the Northerly direction produced the responses of Figure 5.30. The CANFIS autopilot is again superior, yielding a more damped and accurate course following response.

Comparison of the corresponding roll responses (Figure 5.31) displays the reduced coupling achieved by employing this autopilot strategy. The yaw response of Figure 5.32 re-iterates the improved stability of the CANFIS strategy.

As a final experiment concerning autopilot robustness, a sea current disturbance of 2.83 ms^{-1} in the North Westerly direction was simulated. The AUV responses pertaining to this simulation are reproduced in Figures 5.33, 5.34 and 5.35. The ANFIS autopilot strategy results in oscillatory yawing and rolling motion over the verification course. Indeed, at approximately 390 seconds into the simulation the excessive yaw demand placed upon the autopilot system leads to the vehicle rolling completely over through approximately 820° (over 2 full revolutions). This behaviour is obviously unsuitable, and causes the AUV to overshoot the penultimate way point.

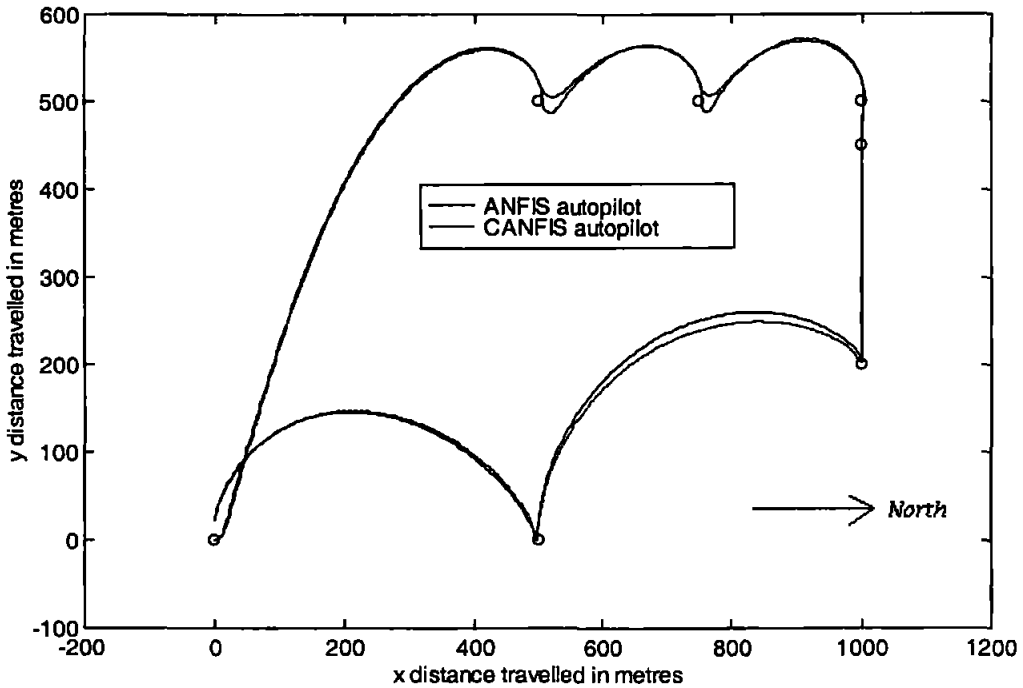


Figure 5.28: Line of sight responses over the verification track in the presence of a current disturbance of 3 ms^{-1} along the Westerly axis.

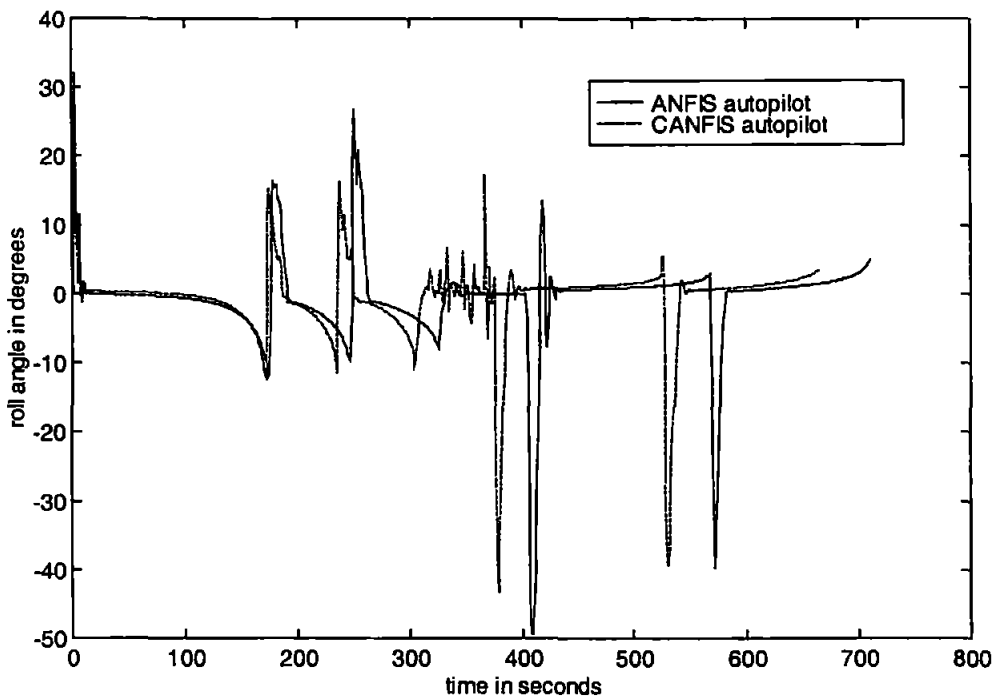


Figure 5.29: Roll responses over the verification track in the presence of a current disturbance of 3 ms^{-1} along the Westerly axis.

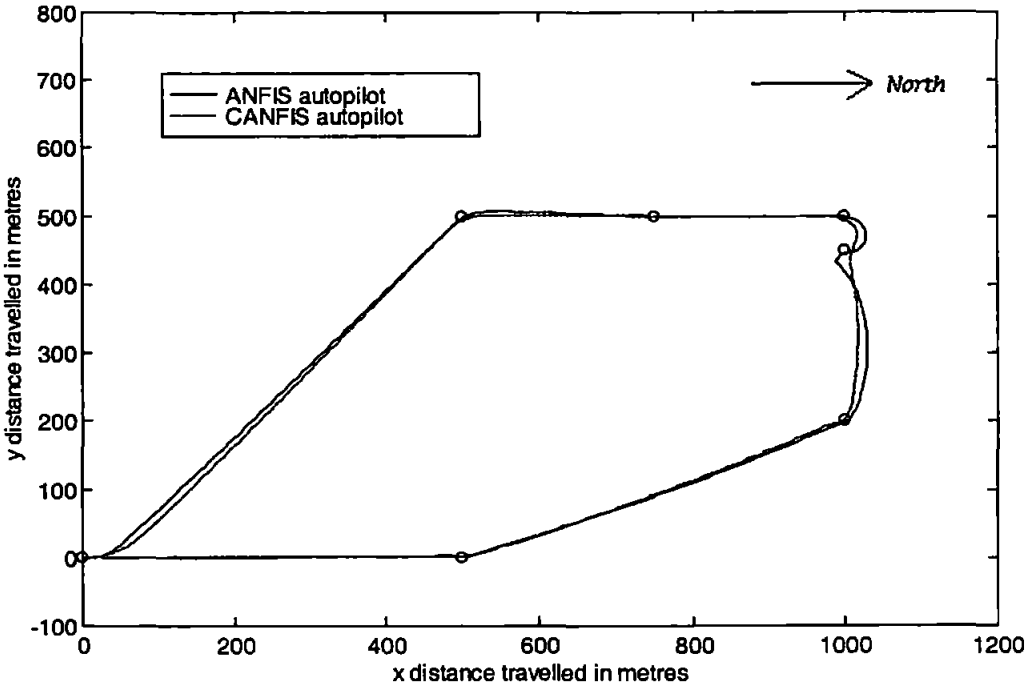


Figure 5.30: Line of sight responses over the verification track in the presence of a current disturbance of 2.5 ms^{-1} along the Northerly axis.

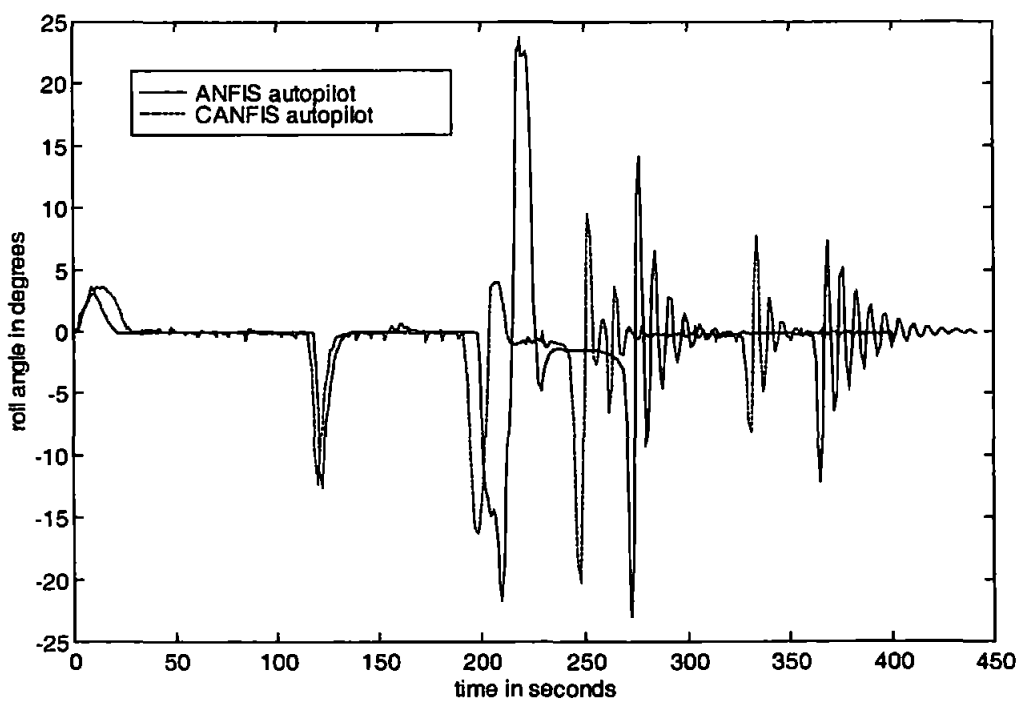


Figure 5.31: Roll responses over the verification track in the presence of a current disturbance of 2.5 ms^{-1} along the Northerly axis.

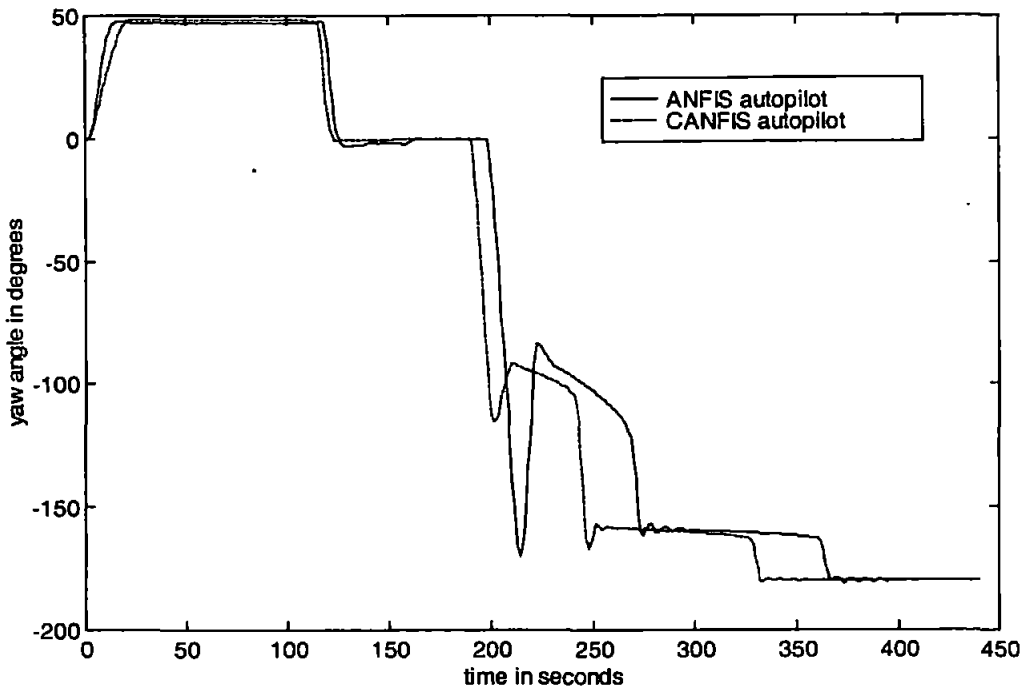


Figure 5.32: Yaw responses over the verification track in the presence of a current disturbance of 2.5 ms^{-1} along the Northerly axis.

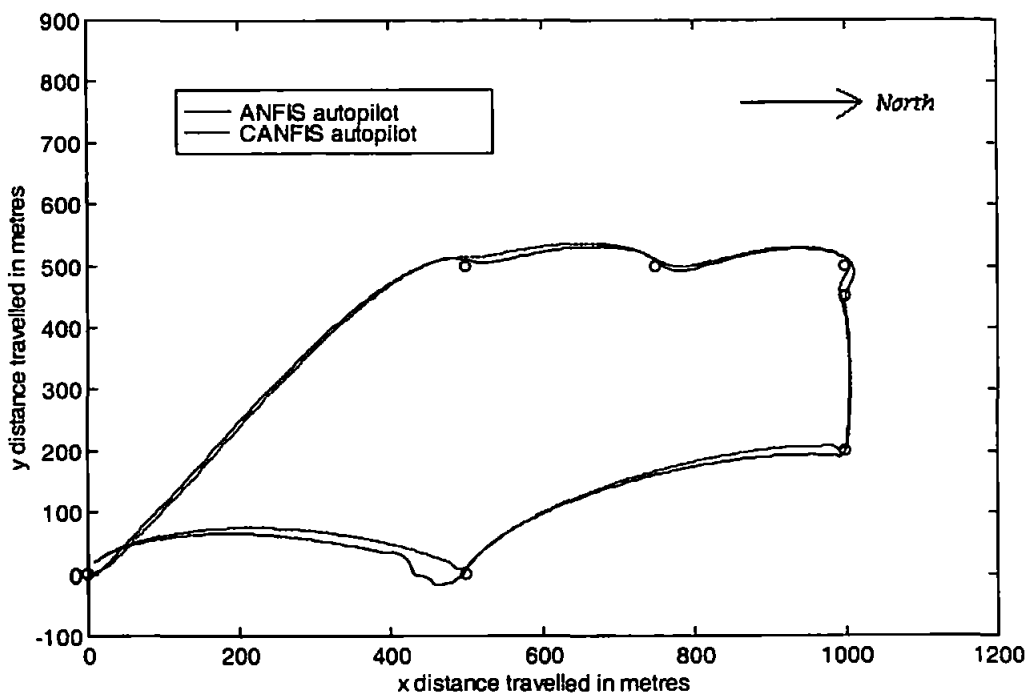


Figure 5.33: Line of sight responses over the verification track in the presence of a current disturbance of 2.83 ms^{-1} in the North Westerly direction.

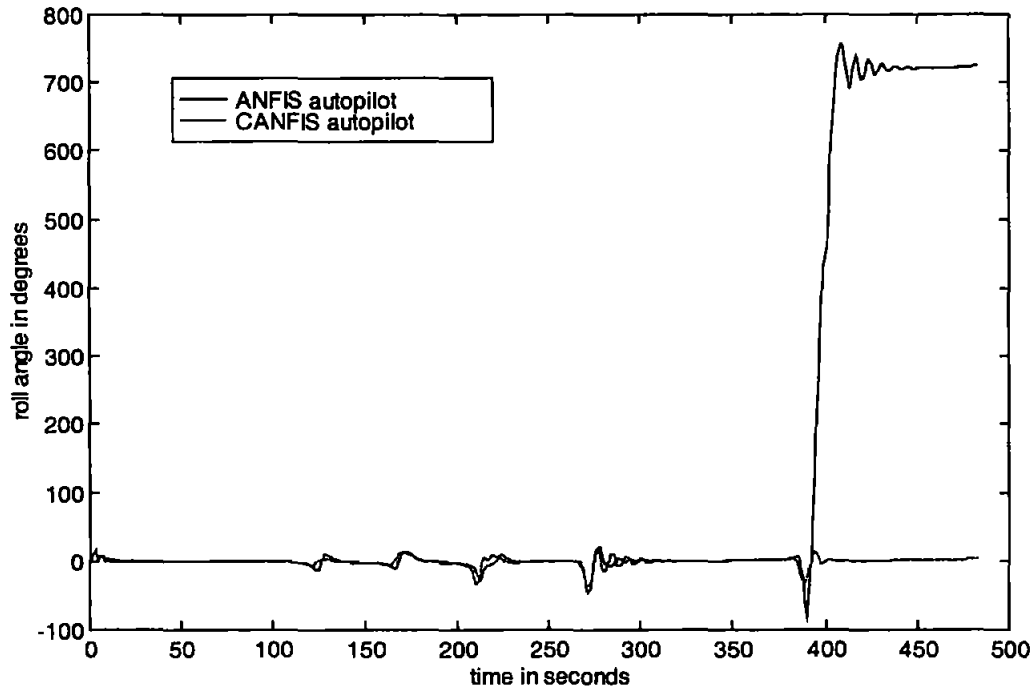


Figure 5.34: Roll responses over the verification track in the presence of a current disturbance of 2.83 ms^{-1} in the North Westerly direction.

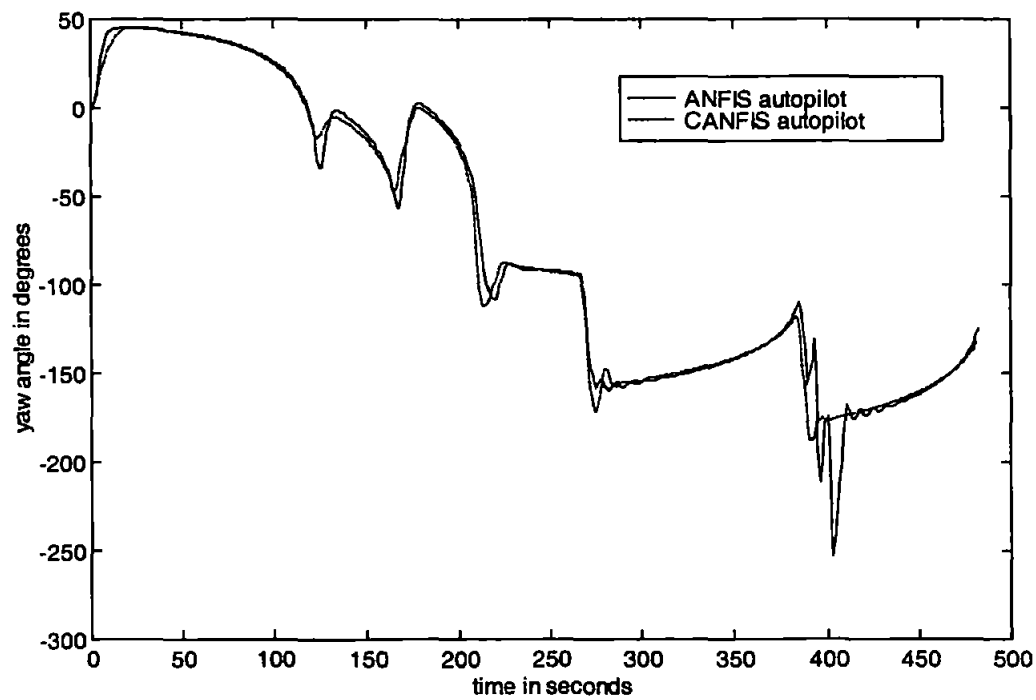


Figure 5.35: Yaw responses over the verification track in the presence of a current disturbance of 2.83 ms^{-1} in the North Westerly direction.

5.6 Concluding Remarks

This chapter has discussed the design and tuning of a novel multivariable TSK style fuzzy autopilot for an AUV. The presented results demonstrate the effectiveness of the new approach over a wide speed envelope and in light of hydrodynamic coefficient variations. However, whilst the multivariable autopilot solution yields superior performance to traditional SISO control approaches, dynamic variations in the vehicle environment cannot be incorporated into the control algorithm. Clearly, an autopilot system that provides a steady platform solution under dynamic changes is essential in order that the on-board sensor packages are used to their fullest. To this end, the following chapter documents the application of a novel on-line learning AUV control algorithm.

It should be noted that the technique used within this chapter was also applied to control the yaw and sway dynamics simultaneously, as discussed in Craven et *al.* (1998). Although the proposed tuning method employs a neural network architecture to tune the parameters of the autopilot it should be noted that the resulting autopilot is purely fuzzy.

References

Choi, J.L. and Hwang, C.-S. (1997). On the Fuzzy - Neural Controller for a Multivariable Non-linear System. Proceedings of the Second Asian Control Conference, Seoul, Vol. 3, pp631-634.

Cowling, D. and Corfield, S.J. (1995). Control Functions for Autonomous Underwater Vehicle On-Board Command and Control Systems. IEE Colloquium on Control and Guidance of Remotely Operated Vehicles, Savoy Place, London, Digest No.:1995/124, pp3/1-3/8.

Craven, P. J., Sutton, R., Burns, R. S. and Dai, Y.-M. (1998). Multivariable Intelligent Control Strategies for an Autonomous Underwater Vehicle. International Journal of Systems Science. Accepted for publication.

Doebelin, E.O. (1985). Control System Principles and Design. John Wiley and Sons Inc., New York, USA.

Goheen, K. R. and Jefferys, E. R. (1990). Multivariable Self-Tuning Autopilots for Autonomous and Remotely Operated Underwater Vehicles. IEEE Journal of Oceanic Engineering, Vol. 15, No.3, pp144-151.

Healey, A. and Lienard, D. (1993). Multivariable Sliding Mode Control for Autonomous Diving and Steering of Unmanned Underwater Vehicles. IEEE Journal of Oceanic Engineering, Vol. 18, No. 3, pp327-339.

Jang, J.-S. R. (1992). Neuro-Fuzzy Modelling: Architecture, Analyses and Applications, Ph.D Thesis, Department of Electrical Engineering, University of California, Berkeley, CA 94720, United States of America.

Jang, J.-S. R., Sun, C.-T. and Mizutani, E. (1997). Neuro-Fuzzy and Soft Computing. Prentice-Hall, New Jersey.

Maciejowski, J. M. (1989). Multivariable Feedback Design. Addison-Wesley Publishing Company.

Mizutani, E. and Jang, J.-S. R. (1995). Co-Active Neuro-Fuzzy Modelling. Proceedings of the International Conference on Neural Networks, pp760-765, Perth, Australia.

Nie, J. and Linkens, D.A. (1993). Learning Control using Fuzzified Self-Organizing Radial Basis Function Network. IEEE Transactions on Fuzzy Systems, Vol. 1, No. 4, pp280-287.

Takagi, T. and Sugeno, M. (1985). Fuzzy Identification of Systems and its Applications to Modelling and Control. IEEE Transactions on Systems, Man and Cybernetics, Vol. 15, pp116-132.

Trebi-Ollennu, A., King, J. and White, B.A. (1995). A Study of Robust Multivariable Control Designs For Remotely Operated Vehicle Depth Control Systems. IEE Colloquium on Control and Guidance of Remotely Operated Vehicles, Digest No.: 1995/124, No.2.

Zehner, W. J. and Thompson, R. L. (1996). Methods of Estimating Allowing Motion Perturbations in Side-Scan Sonar. IEEE Journal of Oceanic Engineering, Vol. 21, No. 3, pp245-255.

Chapter 6

On-Line Neuro-Fuzzy Autopilot Design and Simulation

6.1 Introduction

During a typical mission scenario an autonomous underwater vehicle (AUV) will experience various disturbances. These disturbances can arise from numerous sources including sea currents or variations within the AUV payload. Additionally, the forward velocity of the vehicle may vary over a mission, even when regulated. Therefore, the ability of the autopilot system to adapt in the presence of dynamic changes is a very attractive property, and must be incorporated if a control system is to be considered as intelligent.

This chapter discusses on-line control of the AUV model. Primarily, the adaptive network-based fuzzy inference system (ANFIS) of Jang (1992) is used for on-line

course-changing control. A novel multivariable autopilot structure, based on the Co-active network-based fuzzy inference system (CANFIS) architecture (Mizutani and Jang (1995)) of Chapter 5, is then used for simultaneous on-line course-changing and roll-regulating control. The hybrid learning algorithm is employed in both instances, yet in an on-line form, to tune the fuzzy sets in the premise and consequent portions of the rule base, which collectively constitute the fuzzy controller.

Simulation results are presented which illustrate the effectiveness of the novel multi input-single output (MISO) application. Results are also presented for the proposed multivariable on-line approach. These results are compared to the control approaches of the previous two chapters, which are not capable of varying their parameters on-line to cope with disturbances. The adopted approach leads to a more flexible autopilot in that dynamic changes within the vehicle and environment can be compensated for more effectively. Additionally, the results provide a glimpse of the applicability of the tuning approach to real time control of an AUV.

6.2 Existing On-Line Control Strategies

The use of conventional control systems for AUVs is limited in that the hydrodynamic coefficients of a particular vehicle are not usually known until after the vehicle has been completely designed. Consequently, the repetitious design of autopilot systems can be an expensive process, which involves extensive testing and reconfiguration in line with changing vehicle sub-systems and architectures. Subsequently, an intelligent adaptive control strategy is highly desirable to reduce design overheads and provide compensation of the time varying disturbances and dynamics that such vehicles encounter. In recent years, neural network control schemes have been applied extensively to the problem of AUV control system design with varying degrees of success and credibility. Neural control schemes that incorporate on-line learning capability facilitate adaptation of controller parameters in light of such time varying disturbances and dynamics.

Yuh (1990) provides an excellent paper on the feasibility of such an approach by applying a neural network to the design of an on-line underwater robotic vehicle (URV) control system. The autopilot takes the form of a discrete control law, which is subsequently employed within a continuous time dynamic model of the URV. The updating of the autopilot parameters is therefore performed at discrete sampling instances over the mission trajectory, producing a stage adaptive neural network (SANN) control system. The SANN control scheme is clearly described along with the design of specific robustness tests. The application of the backpropagation algorithm to on-line parameter adaptation is discussed and direct attention is given to the requirement for the Jacobian matrix of the plant to be known to elicit training.

Venugopal et *al.* (1992) present an interesting review of direct and indirect neural network control schemes. Consequently, a direct neural network control strategy is discussed within the context of a simulation package based on the *Ocean Voyager* AUV. Included within this scheme are suggestions for approximating the Jacobian of the AUV dynamics by means of a simple gain matrix, which can be adapted along with the updating of the controller parameters via the backpropagation algorithm. Results are presented for on-line control of the vehicle pitch, yaw and depth dynamics. However, overall vehicle control is performed using single input-single output (SISO) autopilots to manage each individual AUV degree of freedom. The results are shown to be highly dependent on the learning rates of the backpropagation algorithm for each individual autopilot. Cross-coupling effects between the vehicles pitching motion and the yaw, roll and sway channels are illustrated to highlight the validity of the decision to separate the control system into SISO autopilot sub-systems. The paper concludes by remarking that the presented control approach is effective in the presence of slow and fast varying disturbances, yet no results are reproduced for quickly varying disturbances.

Ishii et *al.* (1993) detail a self-organizing neural-network control system (SONCS) and apply it to real time adaptive on-line control of the *Twin Burger* AUV. The SONCS

consists of both an 'Imaginary World' (IW) element which computes imaginary training of the controller parameters, and a 'Real World' (RW) element which operates the AUV according to the control objective. Additionally, an identification network is employed as a feed-forward model of the AUV plant and is required to produce state estimates for the IWs update algorithm. Consequently, the synaptic weights within the RW controller are replaced with those found by the IW controller. Results are presented for yaw control of the AUV with and without adaptive SONCS capability to illustrate the effectiveness of the proposed approach. Indeed, the identification network is seen to learn the yaw dynamics of the AUV in spite of measurement noise and the overall control performance of the approach appears good. One obvious criticism of the method is that for on-line control the model of the AUV used within the identification network may become significantly more complex, especially when multiple degrees of freedom are considered within a multi input-multi output (MIMO) control structure.

More recently hybrid neuro-fuzzy control techniques have become increasingly attractive as they can incorporate fuzzy rule-based algorithms by which the autopilot system can be initialized. Juang and Lin (1998) have developed and applied a self-constructing neural fuzzy inference network (SONFIN) which possesses on-line learning capability. Based on a similar paradigm to the ANFIS of Jang (1992), the parameter set of the fuzzy inference system is tuned using a neural network architecture. However, the SONFIN has more sophisticated consequent functions. Initially the consequents are set as fuzzy singletons, but through learning can be self-constructed to add elements of the more typical Takagi-Sugeno-Kang (TSK) (Takagi and Sugeno (1985)) linear functions of ANFIS. This on-line construction of the architecture allows only those elements of these linear functions that are significant to be added to the overall consequent function, resulting in a more efficient use of parameters. The SONFIN is applied to various problems including temperature control of a water bath and prediction of a Mackey-Glass chaotic time series. The results provided illustrate the effectiveness of TSK linear consequent functions for control over

more typical fuzzy singleton consequent functions. However, due to the wide number of application examples considered within the paper, the results pertaining to control are somewhat limited in detail. Notwithstanding, the application of on-line neuro-fuzzy control is said to be very effective.

Other on-line approaches to the problem of control system design have been employed within the literature. For example, Corradini and Orlando (1997) presented a MIMO adaptive discrete-time variable structure approach to the problem of position and orientation control of a remotely operated vehicle (ROV) model. The resulting autopilot performed well under a variety of current disturbances, but disappointingly the simulation results were of limited detail.

6.3 The On-Line Learning Scheme

By adopting the well-documented technique of '*temporal backpropagation*', that is backpropagation over successive time intervals, the autopilot of the AUV can be encoded as a series of SANN, as discussed in the work of Jang (1992). Following the success of this approach, the AUV autopilot structure herein is initially implemented as a SANN, and the simulation is then interrupted at pre-specified discrete sampling points k over the mission time space.

Specifically, given the state of the plant at time $t = k \times h$, where h is the sampling interval width, the autopilot will generate an input to the plant based upon the modified parameter set. Implementing this procedure from $t = 0$ to $t = t_{final}$ yields the plant trajectory. This path is determined by the initial fuzzy autopilot and the output of each stage adaptive autopilot based on the ensuing parameter adaptations. This principle is shown conceptually in Figure 6.1.

Clearly, in Figure 6.1 it is necessary to make the assumptions that the delay through the controller is small, the plant is static where the next state is explicitly dependent on the last, and the required states are obtainable. Thus by initializing the system with either the knowledge of an expert operator or, as in this example, a previously designed autopilot that performs the required task, the hybrid learning rule can be employed to recursively adapt the parameters of the stage adaptive fuzzy autopilot.

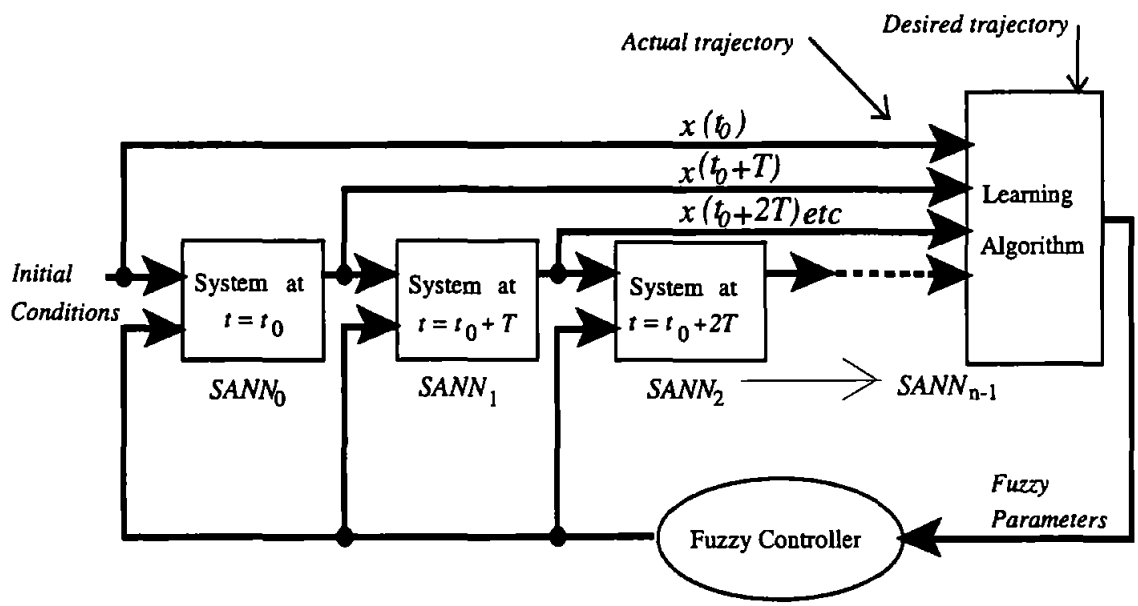


Figure 6.1: Conceptual schematic of the state transition diagram (after Jang (1992)).

The cost function to be minimized can be re-defined as

$$E = \sum_{k=1}^{n} \left\| \bar{x}(t_0 + k \times h) - \bar{x}_d(t_0 + k \times h) \right\| \tag{6.1}$$

where $\bar{x}(t_0 + h \times k)$ is the actual state and $\bar{x}_d(t_0 + h \times k)$ is the desired trajectory at the sampling interval $t = h \times k$. This propagation of the error signals through time has been employed elsewhere, for example by Kwiesielewicz et *al.* (1997).

6.3.1 The Hybrid Learning Rule - On-Line Control

In the case of a time varying system a forgetting factor can be introduced into the sequential least squares estimator (LSE) which places heavier emphasis on more recent data pairs. This produces an algorithm which can be employed to estimate the autopilot consequent parameters on-line. The ability of such an algorithm to take account of more recent data pairs, when estimating the parameters of a controller, can aid in tracking a desired trajectory in the light of varying vehicle dynamics.

Introducing a matrix W of forgetting factors

$$W = \begin{bmatrix} \lambda^{m-1} & 0. & . & 0 \\ 0 & \lambda^{m-2} & . & . \\ . & . & . & 0 \\ 0 & . & 0 & 1 \end{bmatrix} \tag{6.2}$$

where $(0 < \lambda \leq 1)$ and m is the dimension of matrix A in Eqn.(4.30), the corresponding LSE solution that minimizes the weighted error measure is defined by

$$x_k = \left(A^T W A\right)^{-1} A^T W \underline{b} \tag{6.3}$$

where the subscript k denotes the row dimension of $\underline{b}$ in Eqn.(4.30), and thus the number of data pairs used in the estimate x. Conveniently, x_{k+1} can be written as

$$\begin{aligned} x_{k+1} &= \left(\begin{bmatrix} A \\ \underline{a}^T \end{bmatrix}^T \begin{bmatrix} \lambda W & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} A \\ \underline{a}^T \end{bmatrix} \right)^{-1} \begin{bmatrix} A \\ \underline{a}^T \end{bmatrix}^T \begin{bmatrix} \lambda W & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \underline{b} \\ \underline{b} \end{bmatrix} \\ &= \left(\lambda A^T W A + \underline{a}\,\underline{a}^T\right)^{-1} \left(\lambda A^T W \underline{b} + \underline{a} b\right) \end{aligned} \tag{6.4}$$

To simplify the ensuing notation two $n \times n$ matrices are introduced

$$P_k = \left(A^T W A\right)^{-1} \tag{6.5}$$

$$P_{k+1} = \left(\begin{bmatrix} A \\ \underline{a}^T \end{bmatrix}^T \begin{bmatrix} \lambda W & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} A \\ \underline{a}^T \end{bmatrix} \right)^{-1}$$

$$= \left(\lambda A^T W A + \underline{a}\,\underline{a}^T\right)^{-1} \tag{6.6}$$

which are related by the expression

$$\lambda P_k^{-1} = P_{k+1}^{-1} - \underline{a}\,\underline{a}^T \tag{6.7}$$

Substituting Eqn.(6.5) into equation Eqn.(6.3), and Eqn.(6.6) into Eqn.(6.4) yields

$$x_k = P_k A^T W \underline{b} \tag{6.8}$$

$$x_{k+1} = P_{k+1}\left(\lambda A^T W \underline{b} + \underline{a} b\right). \tag{6.9}$$

To express x_{k+1} in terms of x_k it is necessary to eliminate $A^T W \underline{b}$ from Eqn.(6.8) and Eqn.(6.9). Pre-multiplying Eqn.(6.8) by P_k^{-1}

$$P_k^{-1} x_k = P_k^{-1} P_k A^T W \underline{b}$$

$$A^T W \underline{b} = P_k^{-1} x_k . \tag{6.10}$$

Substituting Eqn.(6.10) into equation Eqn.(6.9) gives

$$x_{k+1} = P_{k+1}\left(\lambda P_k^{-1} x_k + \underline{a} b\right)$$

$$= P_{k+1}\left[\left(P_{k+1}^{-1} - \underline{a}\,\underline{a}^T\right)x_k + \underline{a} b\right]$$

$$= x_k + P_{k+1}\,\underline{a}\left(b - \underline{a}^T x_k\right). \tag{6.11}$$

From Eqn.(6.7):

$$\lambda P_k^{-1} = P_{k+1}^{-1} - \underline{a}\,\underline{a}^T \tag{6.12}$$

Using the property of the matrix inversion formula:

$$(A + BC)^{-1} = A^{-1} - A^{-1}B\left(I + CA^{-1}B\right)^{-1} CA^{-1}$$

with $A = \lambda P_k^{-1}$, $B = \underline{a}$ and $C = \underline{a}^T$, Eqn.(6.12) becomes

$$P_{k+1} = \frac{1}{\lambda}P_k - \frac{1}{\lambda}P_k\,\underline{a}\left(I + \underline{a}^T\frac{1}{\lambda}P_k\,\underline{a}\right)^{-1}\underline{a}^T\frac{1}{\lambda}P_k$$

$$= \frac{1}{\lambda}\left(P_k - \frac{P_k\,\underline{a}\,\underline{a}^T P_k}{\lambda + \underline{a}^T P_k\,\underline{a}}\right), \tag{6.13}$$

which is the recursive least squares formula for a time varying system. Obviously if λ is chosen as 1, the sequential least squares formula for a time invariant system is re-formulated. The sequential least squares estimator for time variant systems with multiple outputs can be obtained almost identically as:

$$\begin{cases} P_{k+1} = \dfrac{1}{\lambda}\left[P_k - \dfrac{P_k\,\underline{a}_{k+1}\,\underline{a}_{k+1}^T\,P_k}{\lambda + \underline{a}_{k+1}^T\,P_k\,\underline{a}_{k+1}}\right] \\ x_{k+1} = x_k + P_{k+1}\,\underline{a}_{k+1}\left(\underline{b}_{k+1}^T - \underline{a}_{k+1}^T\,x_k\right) \end{cases} \tag{6.14}$$

The value of λ determines the rate at which the effect of old data pairs decays. A high value of λ ($\lambda \rightarrow 1$) produces a high rate of decay and vice-versa. However, lower values of λ can cause instability and thus the chosen value is typically taken above 0.95.

With respect to the backpropagation element of the hybrid learning rule, Eqn.(4.13) is employed as the update formula for the non-linear premise parameters, as opposed to Eqn.(4.14) which is used for off-line learning.

Additionally, the sampling rate and step size of the gradient algorithms transition through parameter space will have a direct effect on the stability of the AUV response. The effect of these rates will be examined more closely in the following sections.

6.4 Results and Discussion

The feasibility of the proposed SANN control scheme is again assessed through the AUV simulation model. The direct learning control system was implemented as discussed in section 6.3 (Figure 6.1), whereby autopilot parameter adjustment is performed at every discrete time stage. With respect to the yaw dynamics of the AUV model, the open loop time constant was calculated as approximately 1.5 seconds (Eqn.(3.13), Chapter 3). Consequently, the sampling interval of the SANN could be set at 0.1 seconds to achieve smooth overall trajectory.

6.4.1 Course Changing Results

To avoid costly oscillation whilst the algorithm adjusted from its initial parameters to a modified parameter set, the system was initialized with the hybrid rule tuned fuzzy autopilot of Chapter 4. Figure 6.2 depicts the course-changing responses of the AUV at 7.5-knots for selected values of λ. The corresponding low canard rudder responses are reproduced in Figure 6.3. Clearly the rate at which old data pairs decay has a direct effect on the autopilot stability and thus on the course-changing response of the AUV.

Throughout these simulations it should be noted that the gradient descent step size transition rate of Eqn.(4.15) remained fixed at 5%.

A well-known problem associated with the recursive least squares algorithm for on-line estimation of parameters is the unsuitability of the resulting estimates when the system dynamics are not continuously stimulated. That is, if the system is not sufficiently excited, the inverse of the covariance matrix of the recursive least squares algorithm can become ill defined or singular over time. This effect can be seen in Figure 6.4; this represents the transition of the first consequent coefficient over the course-changing manoeuvre of Figure 6.2. Attention should be given to the scale of each vertical axis in Figure 6.4 as this represents the magnitude of the parameter variation for each respective simulation.

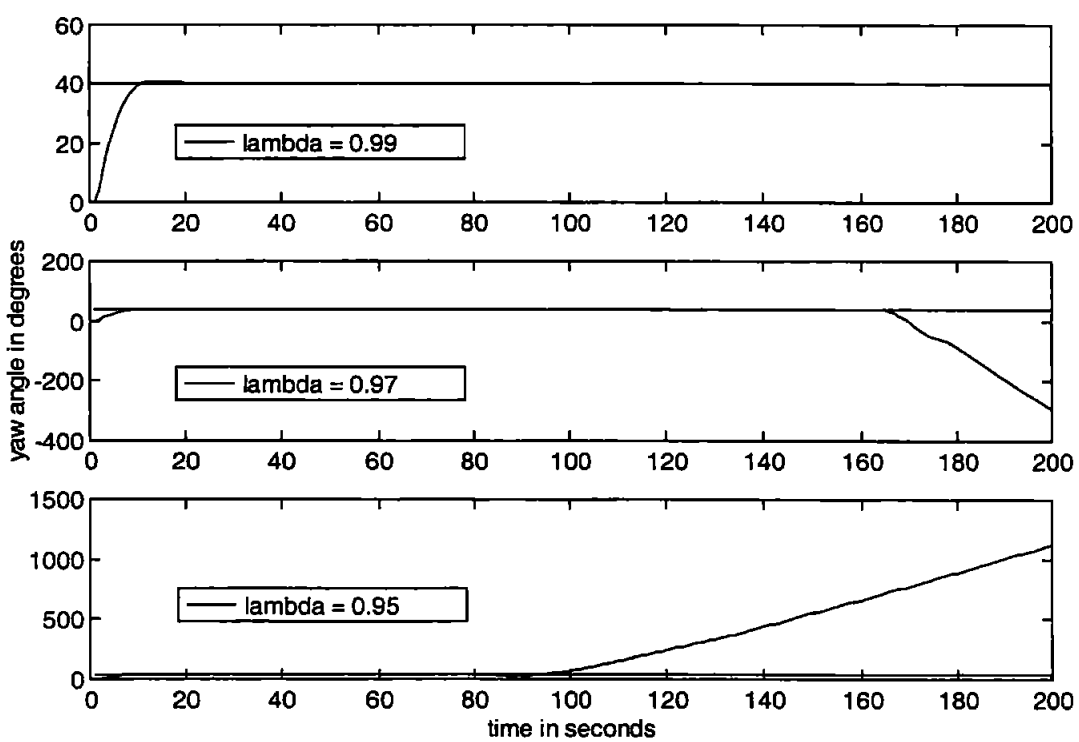


Figure 6.2: Yaw responses of the AUV for a 40° course-changing manoeuvre for λ=0.99, λ=0.97 and λ=0.95.

The results pertaining to λ=0.97 and λ=0.95 clearly show coefficient estimates of large magnitude in comparison to the parameter values for the λ=0.99 simulation. The excessive variations in these (typical) parameters lead to oscillatory control commands as shown by the low canard rudder plots of Figure 6.3 and consequently to the unstable course-changing results of Figure 6.2.

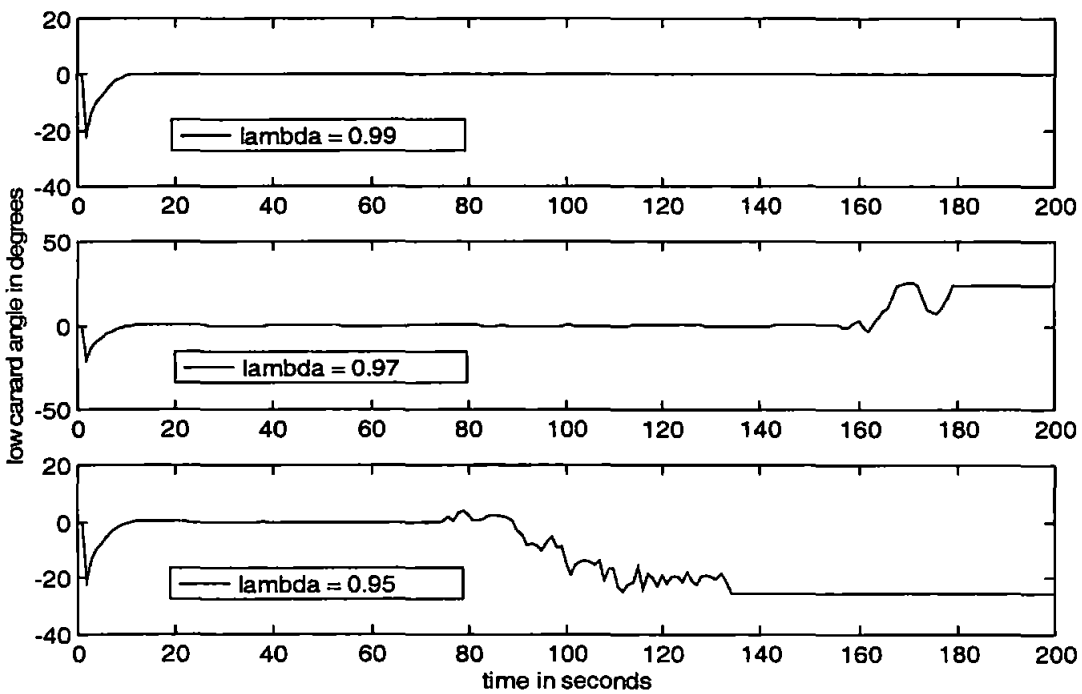


Figure 6.3: Low canard rudder responses of the AUV for a 40° course changing manoeuvre for λ=0.99, λ=0.97 and λ=0.95.

To avoid this singularity problem various methods can be employed:

(i) measurement noise can be added to the control signal to provide constant stimulation of the system dynamics, even in the steady state phase (however this may lead to a chattering effect in terms of control effort as the system effectively never reaches true steady state),

(ii) the covariance matrix can be reset periodically, when singularity is detected,

(iii) estimation of the parameters can be halted or 'switched off' when the system is in the steady state phase, and then 'switched on' again when transient motion is detected.

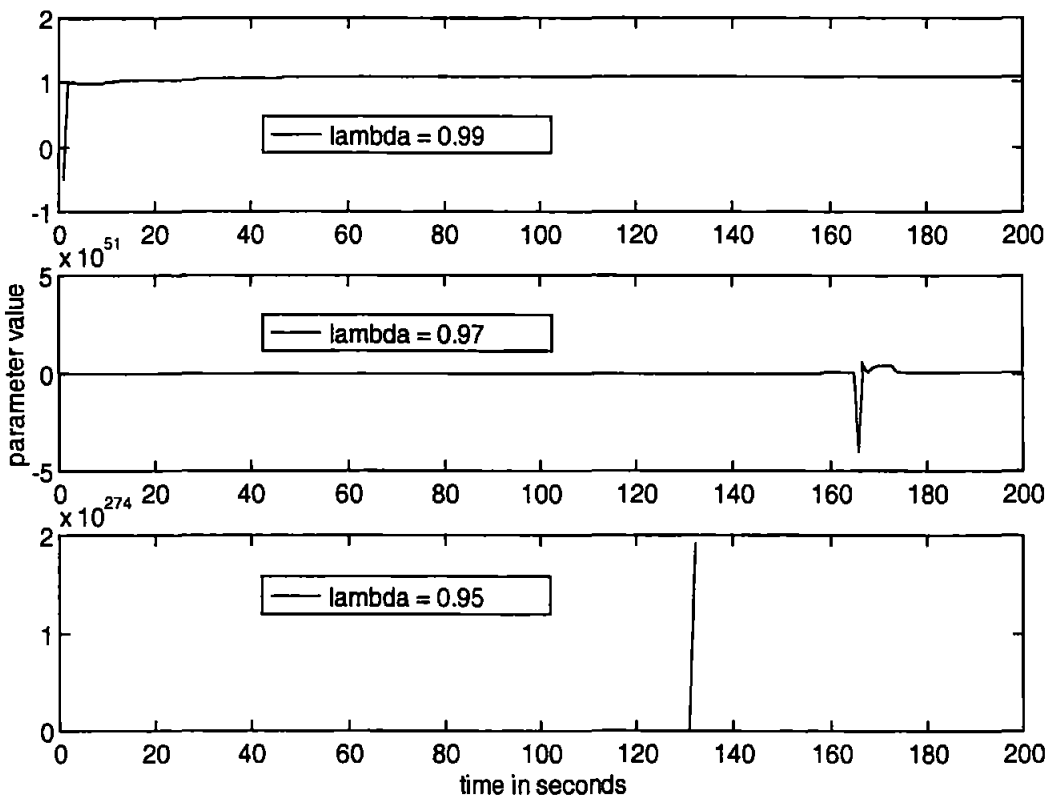


Figure 6.4: Transitions of the first consequent parameter for the 40° course changing manoeuvre for λ=0.99, λ=0.97 and λ=0.95.

The design of an autopilot for an AUV must consider the control effort employed within a particular manoeuvre due to the limited availability of power resources.

Consequently, a design solution which best utilises this control activity with respect to actuator efficiency is sought. Therefore, a combination of strategies (ii) and (iii) above was considered the most effective in minimizing control effort demand whilst maintaining the mathematical requirement that the inverse of the covariance matrix is well defined and non-singular.

A novel on-line algorithm was developed to implement these strategies, and is reproduced in Table 6.1. This rule is dependent upon a pre-specified tolerance level, which dictates the value of course-changing error at which the parameter estimation algorithm should be switched off. Ideally, this should occur in the steady-state phase of the motion. The value of the tolerance level was set at a course error of 0.1 degrees for the following course-changing autopilot simulations; in light of expected sensor accuracy, levels of course error accurate to 2 decimal places may not be physically realizeable. Cetrek Ltd. of Poole boast the ability of their latest compass to measure course angles to within 0.1° tolerances.

The results obtained when employing this refined adaptation algorithm in comparison to the original approach (for varying λ values) are reproduced in Figures 6.5, 6.6 and 6.7, and correspond to yaw response, low canard rudder response and first consequent parameter transition respectively.

The new on-line algorithm produces AUV course changes of notably superior performance for all three selected values of λ. The parameter estimates of Figure 6.7 display stable behaviour when adaptation is restricted to transient periods of motion. This is because the covariance matrix within the recursive least squares estimate of the consequent parameters has been frozen and reset to the identity matrix multiplied by a large scalar value.

1. Set the tolerance level for transition of the system parameters. This value typically corresponds to $\psi_\varepsilon = 0.1$.
2. Simulate the AUV course-change.
3. If the system exhibits $\psi_\varepsilon > 0.1$, adapt the fuzzy parameter set using the hybrid on-line learning rule.
4. Otherwise, retain the parameter set of the previous discretization level.
5. Continue this routine throughout the mission time span.

Table 6.1: The novel on-line tuning algorithm.

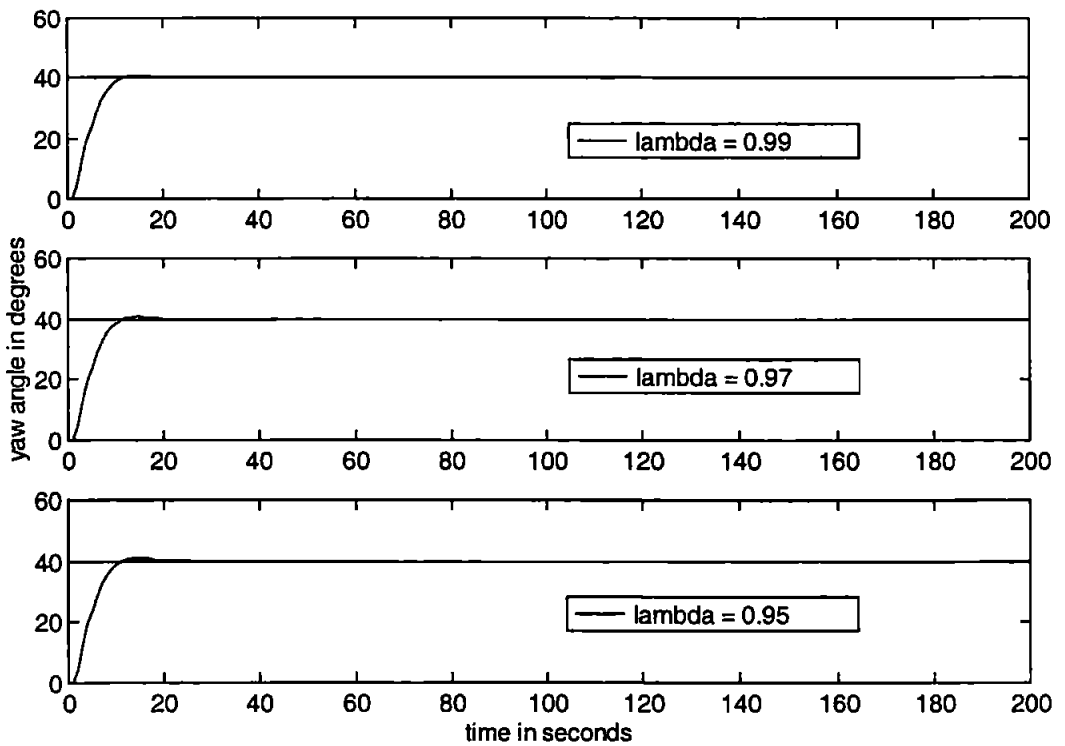


Figure 6.5: Yaw responses of the AUV for a 40° course-changing manoeuvre for λ=0.99, λ=0.97 and λ=0.95.

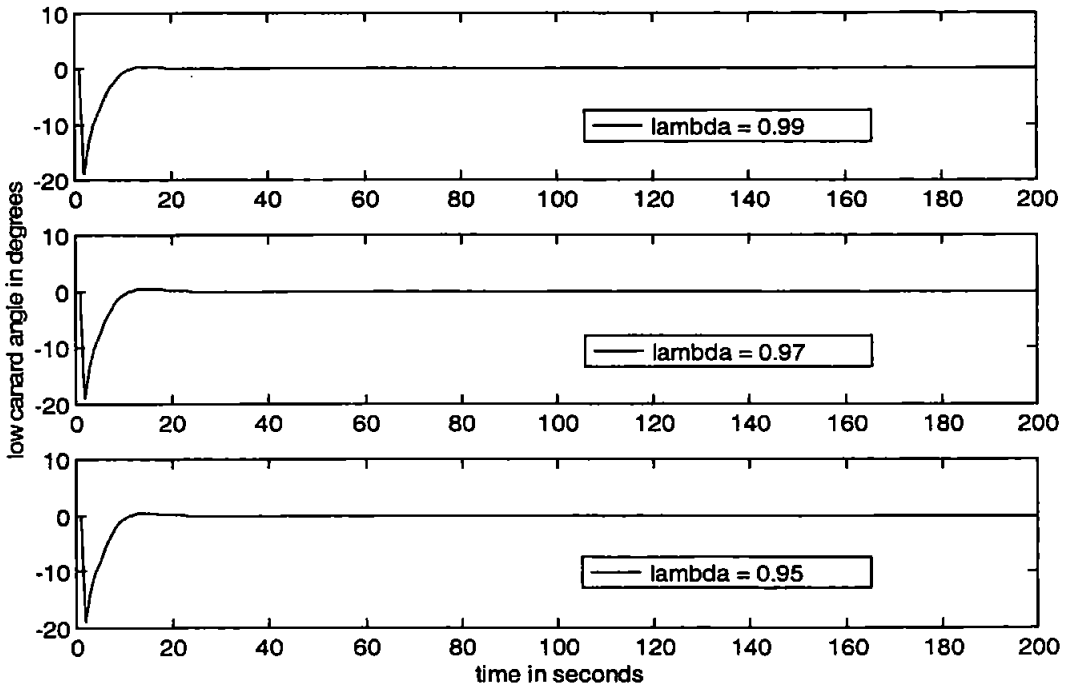


Figure 6.6: Low canard rudder responses of the AUV for a 40° course-changing manoeuvre for λ=0.99, λ=0.97 and λ=0.95.

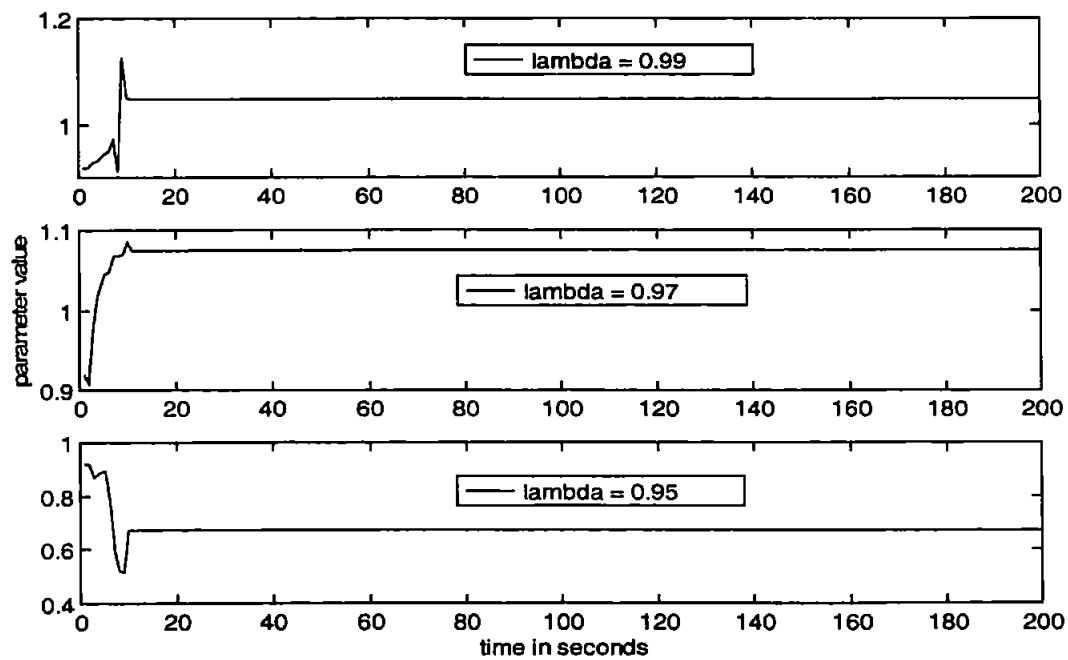


Figure 6.7: First consequent parameter transitions for the 40° course-changing manoeuvre for λ=0.99, λ=0.97 and λ=0.95.

In addition to varying λ, the step size of each gradient transition through parameter space can be altered at the beginning of each on-line simulation. Essentially, this determines the learning rate of the non-linear parameters of the premise membership functions within the neuro-fuzzy architecture, and consequently the rate of convergence of the non-linear parameter set. Figures 6.8, 6.9 and 6.10 illustrate the effects of varying this rate, with values of step size taken as 5%, 10% and 20%. It should be noted that to examine the effects of the step size on the AUV's behaviour, the novel on-line adaptation algorithm is replaced with the original on-line algorithm, whereby parameter adjustment is performed continuously throughout the simulation period.

From the responses it is not evident whether a learning rate of 5% or 10% produces the most superior result. However, the step size transition rate of 20% is clearly too large; the algorithm is attempting to converge too quickly. Figure 6.9 indicates that the 5% step size may produce the more superior results; the 10% plot appears to be approaching an unstable period at approximately 195 seconds into the simulation. This observation is not substantiated by the first consequent parameter transition plot of Figure 6.10, which does not display undue parameter variation during the final 5 to 10 seconds of the simulation. Examination of the premise parameter transitions over this simulation illustrated the instability of the response using a 10% step size, concluding that the 5% step size was the most suitable of those tested.

From these initial on-line simulations it is apparent that the forgetting factor and step size are important coefficients for ensuring the stability of the autopilot output, and thus the AUV. However, use of the modified adaptation algorithm (Table 6.1) alleviates the parameter instabilities caused through poor selection of these coefficients. To illustrate this statement, a forgetting factor of 0.95 and an unsuitably large step size of 20% were chosen for use with the novel adaptation algorithm. Figure 6.11 illustrates the course change, low canard rudder and first consequent parameter transition respectively. Although there is a slight steady state course error (1.63%), the problems of parameter instability associated with the original on-line algorithm are not present.

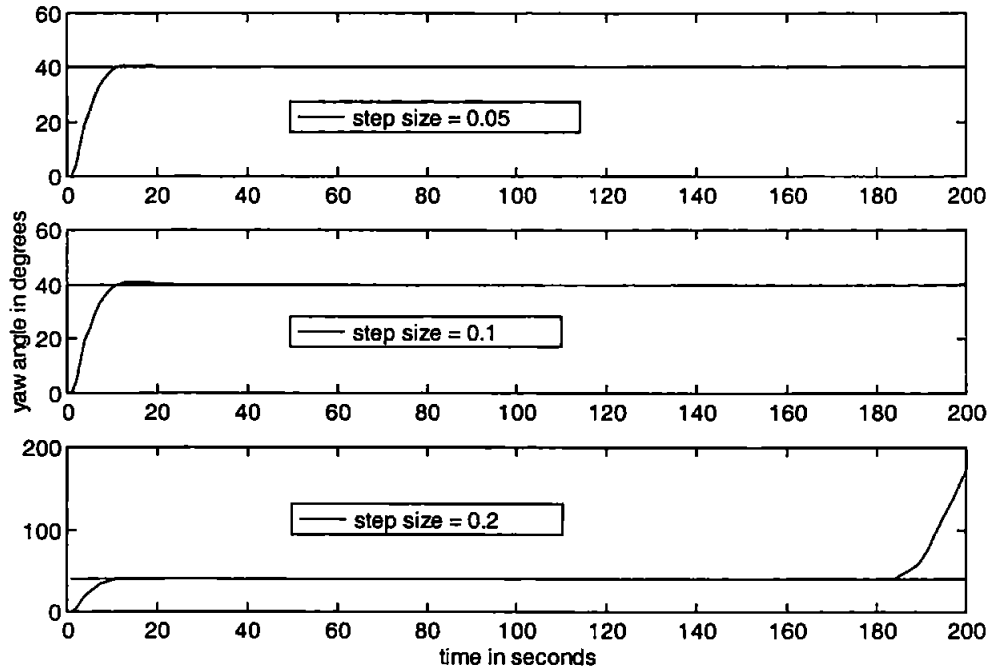


Figure 6.8: Yaw responses of the AUV for a 40° course-changing manoeuvre with a step size of 5%, 10% and 20%.

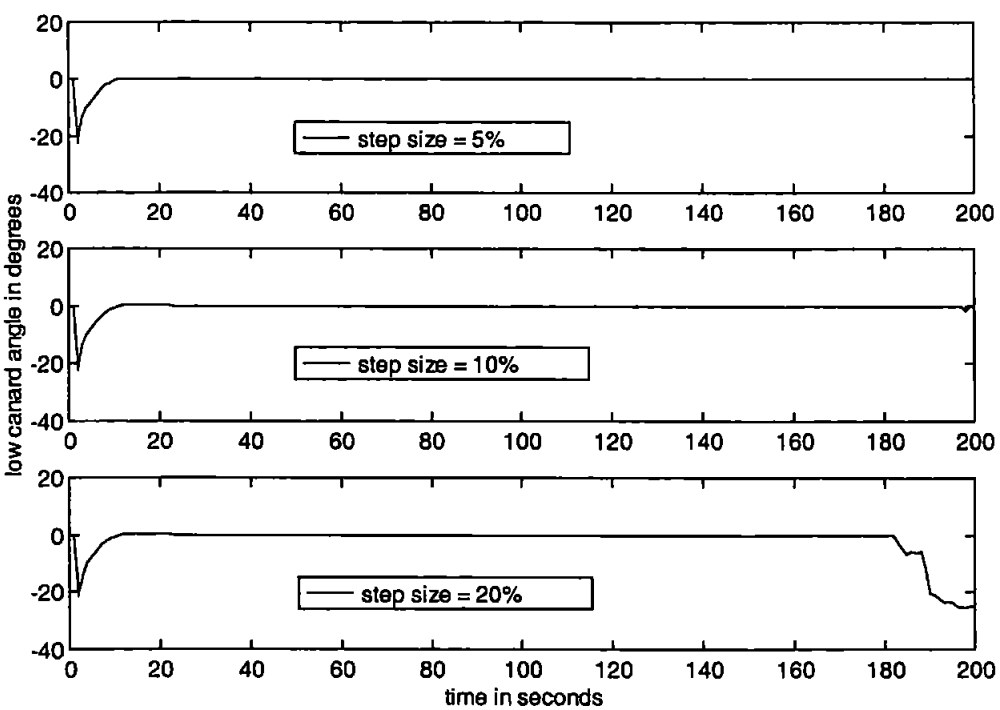


Figure 6.9: Low canard rudder responses of the AUV for a 40° course-changing manoeuvre with a step size of 5%, 10% and 20%.

As with the off-line tuning algorithm, the roll cross-coupling motion induced under such a course-change (Figure 6.8, step size 0.05) is pronounced, as shown in Figure 6.12. The following section discusses a novel approach to AUV control, whereby the CANFIS autopilot of Chapter 5 is tuned on-line to provide compensation for this roll cross-coupling.

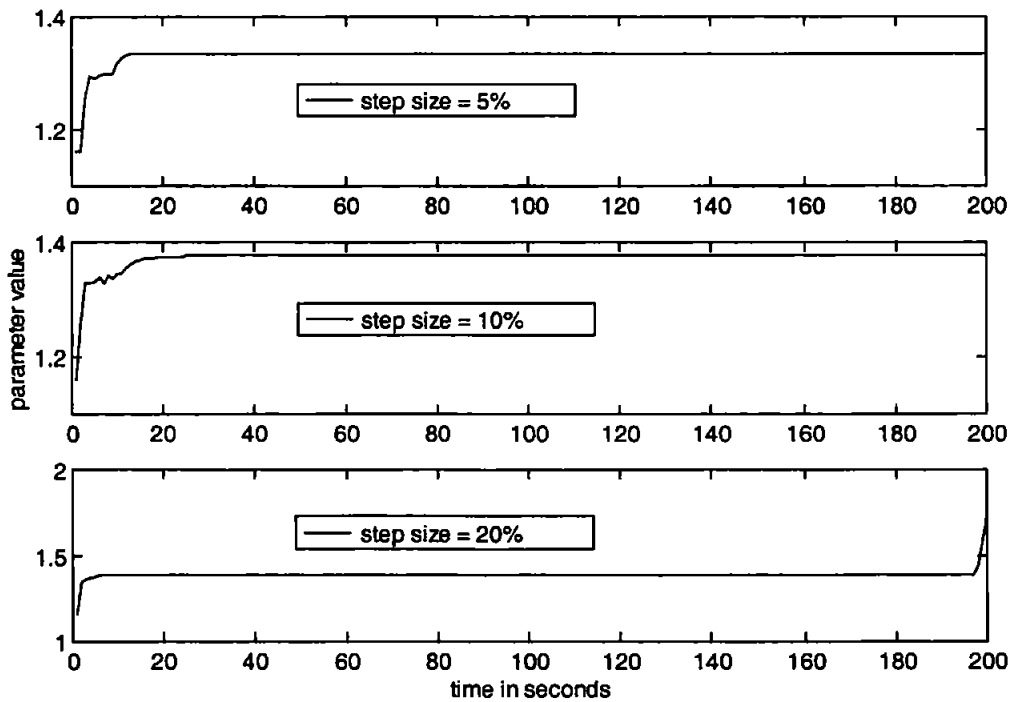


Figure 6.10: First parameter transitions for a 40° course-changing manoeuvre with a step size of 5%, 10% and 20%.

6.4.2 CANFIS On-Line Autopilot Results

This section discusses the implementation of an original approach to on-line multivariable control system operation. The CANFIS technique developed within Chapter 5 is extended beyond the current configuration to one that is suitable for on-line control. This extension manifests itself in the form of the novel on-line hybrid learning rule of Table 6.1, applied to the CANFIS architecture via Eqn.(6.14).

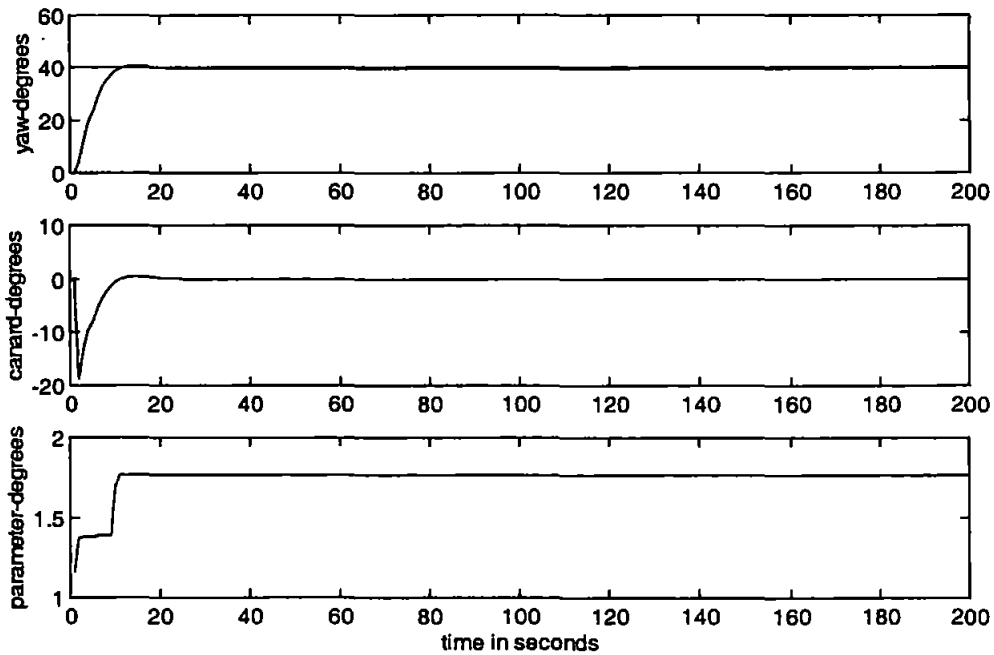


Figure 6.11: Yaw, low canard rudder and parameter transition responses for a 40° course-changing manoeuvre using a forgetting factor of 0.95 and a step size of 20%.

The 16 rule hybrid tuned multivariable fuzzy autopilot of Chapter 5 was used to initialize the SANN control system and thus provide a pre-tuned start point to the course-changing and roll-regulating simulation. The forgetting factor was initially set heuristically at λ=0.975.

Figure 6.13 displays the yaw and roll responses of the AUV during a 7.5-knot simulation whilst employing the proposed CANFIS on-line autopilot. Again, the corresponding low canard rudder and stern hydroplane responses are depicted in Figure 6.14 for completeness.

Although the autopilot steers the AUV smoothly towards the desired set-point with reduced roll cross-coupling, the yaw response indicates a steady state error of approximately 1.22% and the controlled roll response is oscillatory.

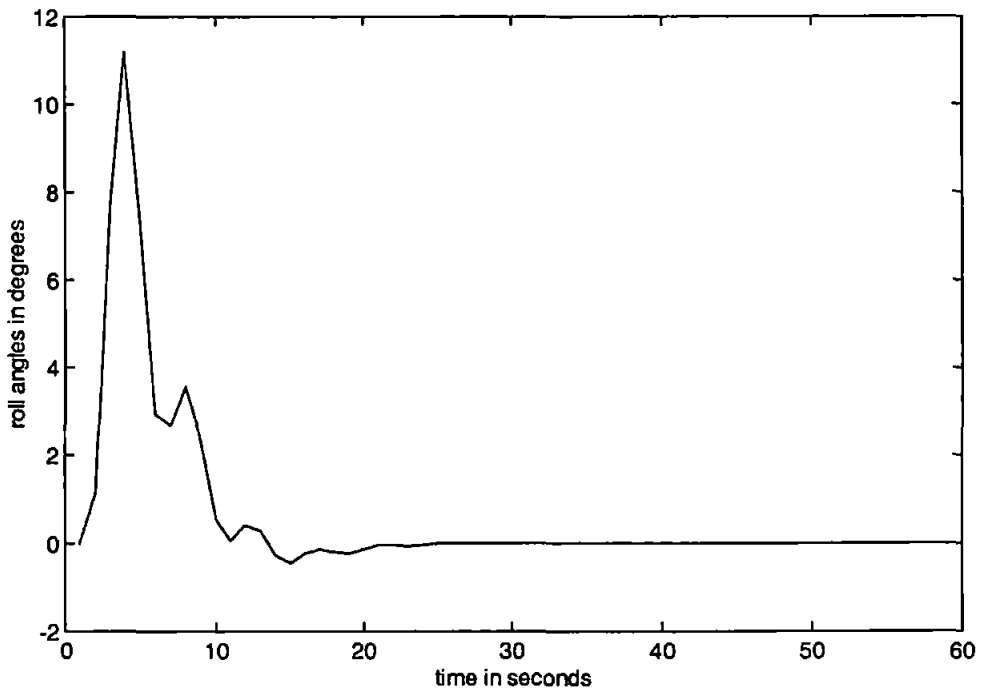


Figure 6.12: Roll response of the AUV for a 40° course-changing manoeuvre with a step size of 5%.

The responses of Figure 6.14 explain the oscillatory course-changing and roll regulating responses of Figure 6.13. The parameter set transition over the simulation period (Figure 6.15) highlights the oscillatory nature of some of the fuzzy autopilot's coefficients. Closer examination of this parameter set highlights the stable nature of the premise parameters in comparison to the oscillatory mode of many of the consequent parameters during the transient period of the course change. The final rulebase of this autopilot is reproduced in Appendix H.

This suggests that the premise parameter step size transition rate of 5% is suitable. Conversely, the estimates of the consequent coefficients, via the sequential least squares element of the hybrid algorithm, are varying excessively during the transient period of the motion. This variation is thus producing unstable AUV responses through oscillatory actuator demands.

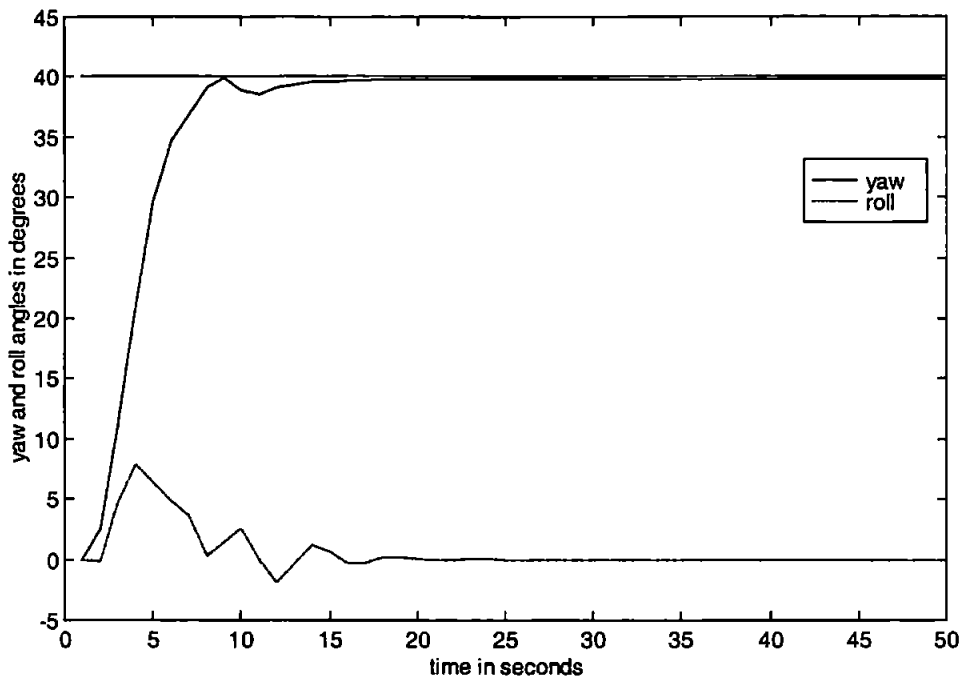


Figure 6.13: Yaw and roll responses of the AUV for a 40° course-changing manoeuvre using a forgetting factor of 0.975 and a step size of 5%.

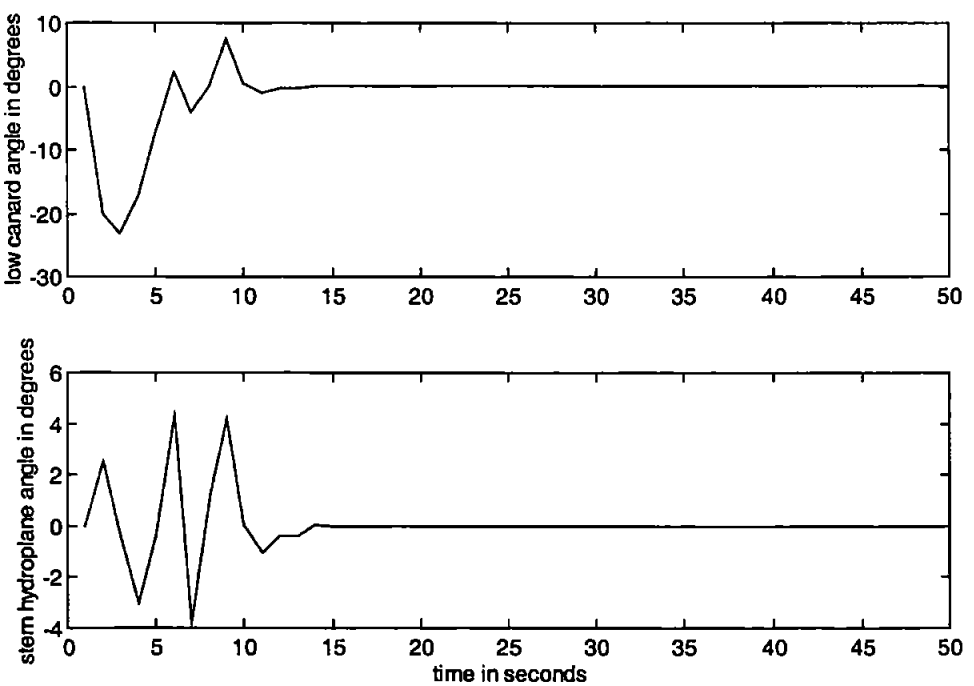


Figure 6.14: Low canard rudder and stern hydroplane responses of the AUV for a 40° course-changing manoeuvre using a forgetting factor of 0.975 and step size of 5%.

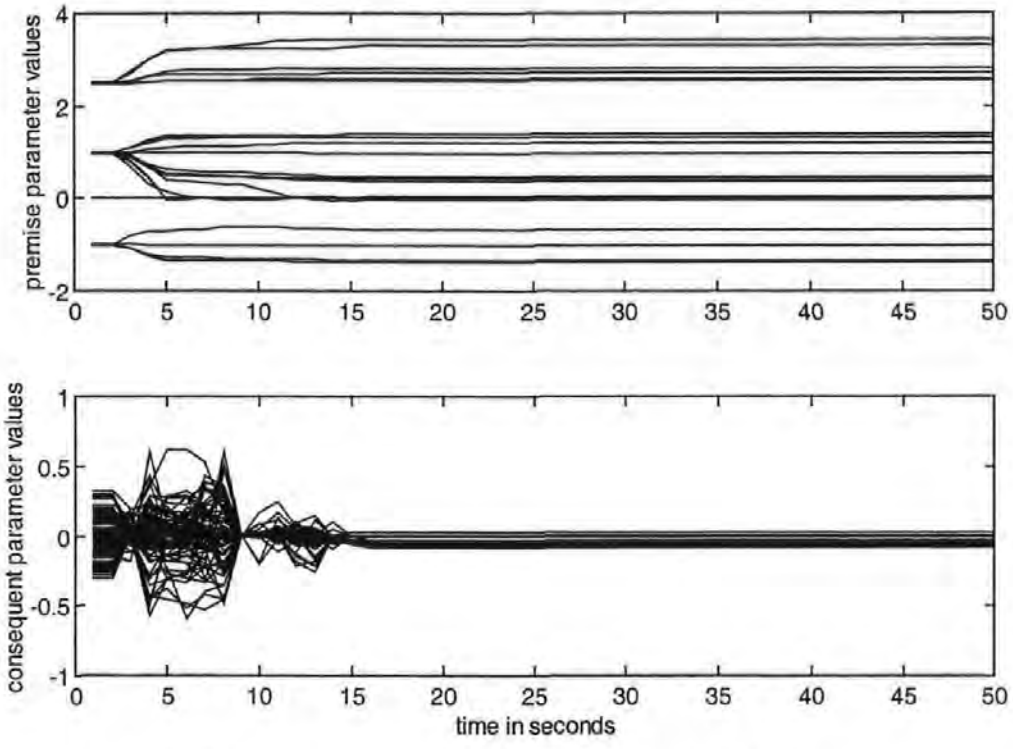


Figure 6.15: Parameter transition for a 40° course-changing manoeuvre using a forgetting factor of 0.975 and a step size of 5%.

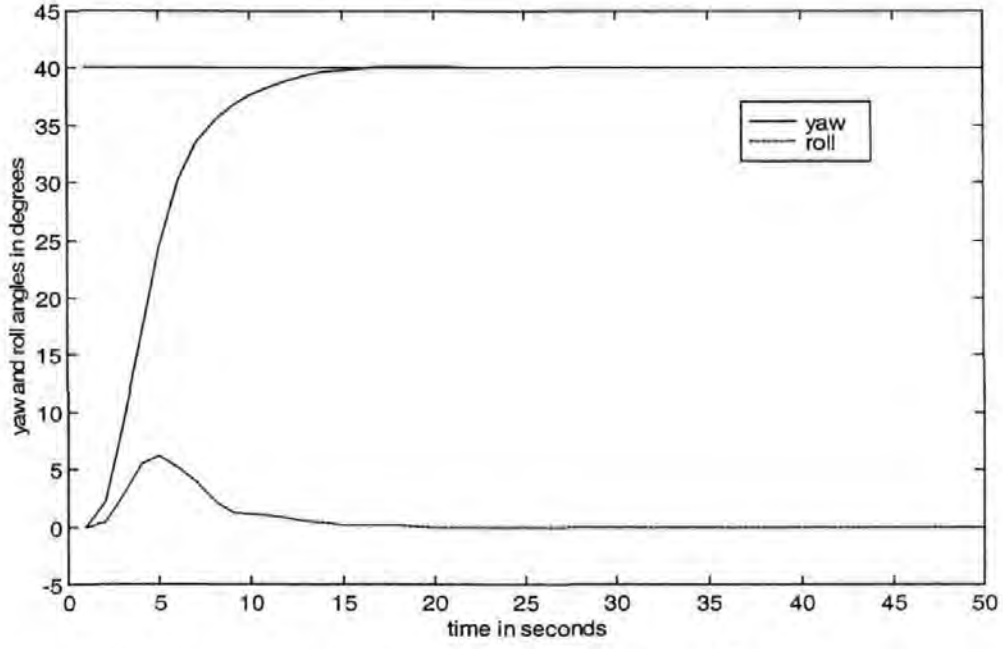


Figure 6.16: Yaw and roll responses of the AUV for a 40° course-changing manoeuvre using a forgetting factor of 0.99 and a step size of 5%.

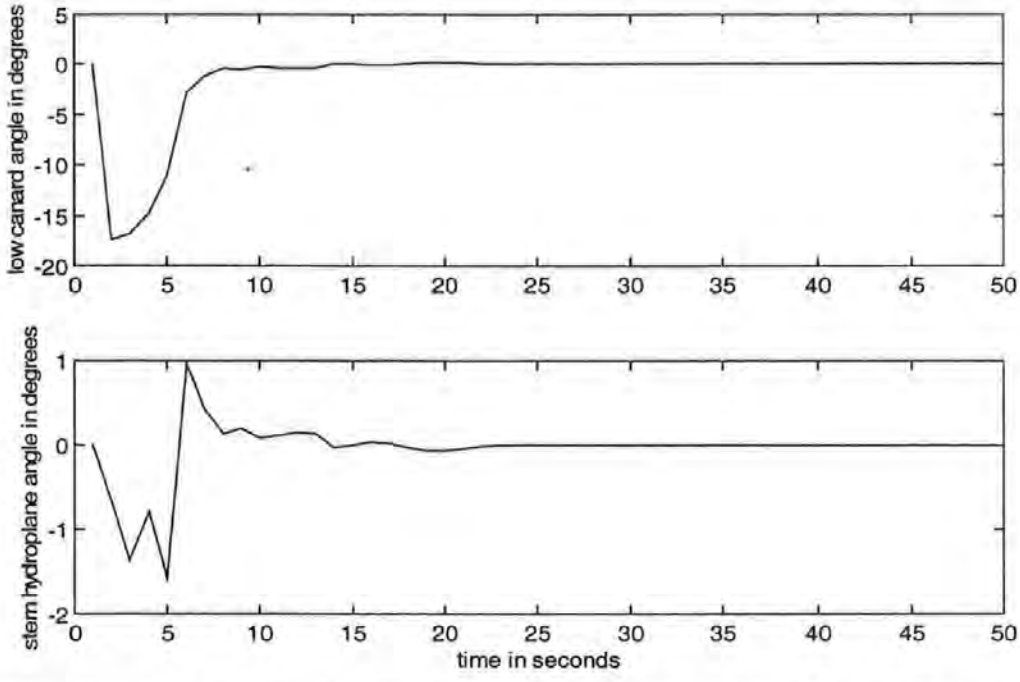


Figure 6.17: Low canard rudder and stern hydroplane responses of the AUV for a 40° course-changing manoeuvre using a forgetting factor of 0.99 and a step size of 5%.

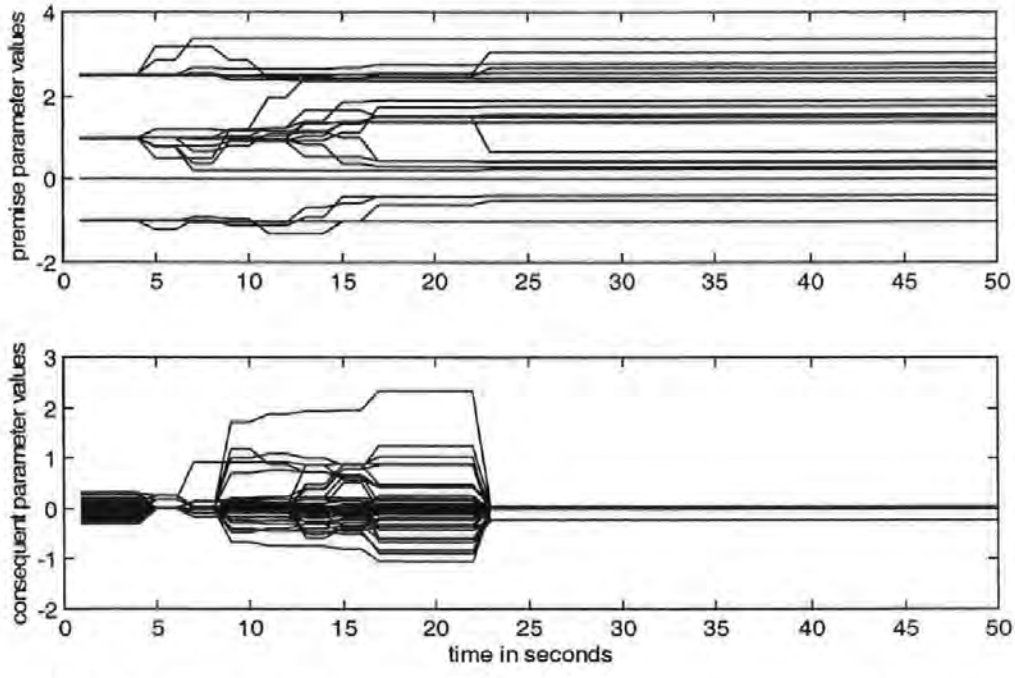


Figure 6.18: Parameter transition for a 40° course-changing manoeuvre using a forgetting factor of 0.99 and a step size of 5%.

Further simulations, examining the influence of the forgetting factor λ on the consequent parameter estimates, lead to a more suitable choice of λ=0.99. The responses of the AUV when employing this value are depicted in Figure 6.16, 6.17 and 6.18. Appendix I details the final rulebase of this autopilot.

The course-changing response of the AUV is slower under the conditions of Figure 6.16 than as previously documented in Figure 6.13. This is illustrated through the responses of Figure 6.17, the low canard rudder reaching a maximum value of 17.4° as opposed to 23.4° when employing a forgetting factor of 0.975. Clearly, employing a larger forgetting factor reduces the effects of new data pairs on the control system parameter estimates, as expected. This property suggests that the introduction of disturbances may require a smaller forgetting factor in order that parameter estimates are updated more frequently. Nevertheless, the responses achieved when employing a forgetting factor of λ=0.99 are superior to those attained with λ=0.975, the roll cross coupled motion being suppressed effectively as compared to that of Figure 6.12 for the unregulated case.

The following section discusses the robustness of the CANFIS on-line autopilot to parameter variations, sea current disturbances and measurement noise.

6.5 Robustness Properties of On-Line Control

This section examines the relative merits of employing the on-line control strategy presented in the preceding sections, in comparison to the most effective off-line tuned multivariable fuzzy autopilot. The robustness properties of the hybrid tuned CANFIS autopilot were discussed in section 5.5. However, these robustness simulations did not account for varying disturbances such as measurement noise. One would expect an on-line autopilot system to better accommodate the characteristics of a varying disturbance, in comparison to a pre-tuned or off-line system. This section is designated to examining this conjecture.

Primarily, the on-line CANFIS autopilot is assessed in light of various hydrodynamic coefficient variations. The autopilot is subsequently tested over the designated line of sight (LOS) verification track. Finally, autopilot robustness to measurement noise is appraised.

6.5.1 Vehicle Coefficient Variations

Figures 6.19 and 6.20 depict the yaw and roll, and low canard rudder and stern hydroplane responses of the AUV respectively, for the nominal AUV mass and when the mass of the vehicle is increased to 175% of its nominal value. During these simulations the CANFIS on-line autopilot was employed with a forgetting factor of λ=0.99, and a gradient descent step size of 5%. The AUV was initialized at a forward speed of 7.5-knots.

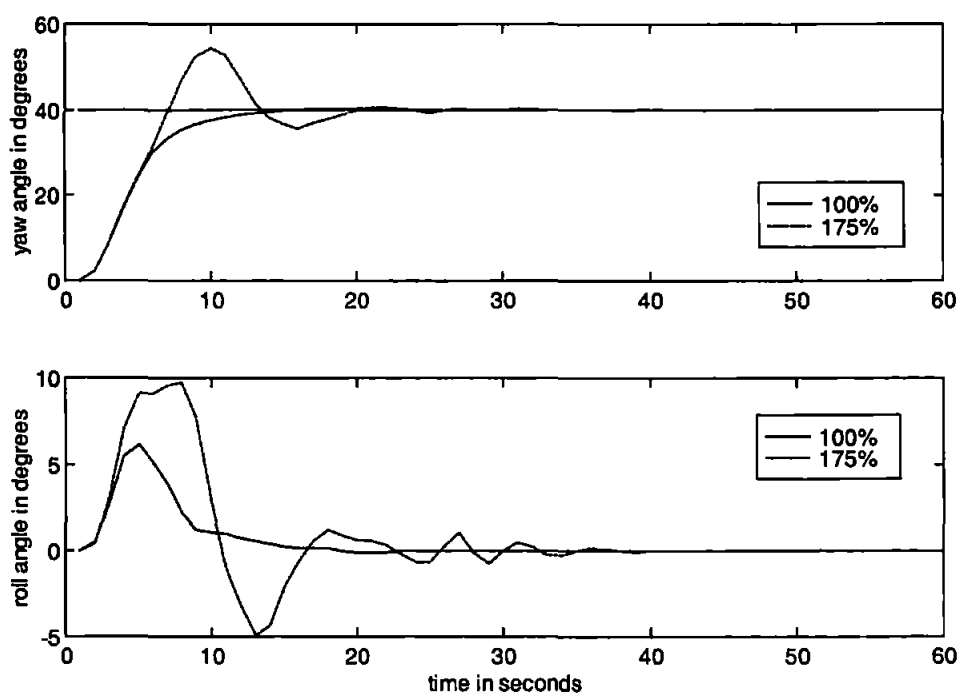


Figure 6.19: Mass variation during a 40° course-change when employing the CANFIS autopilot – yaw and roll responses.

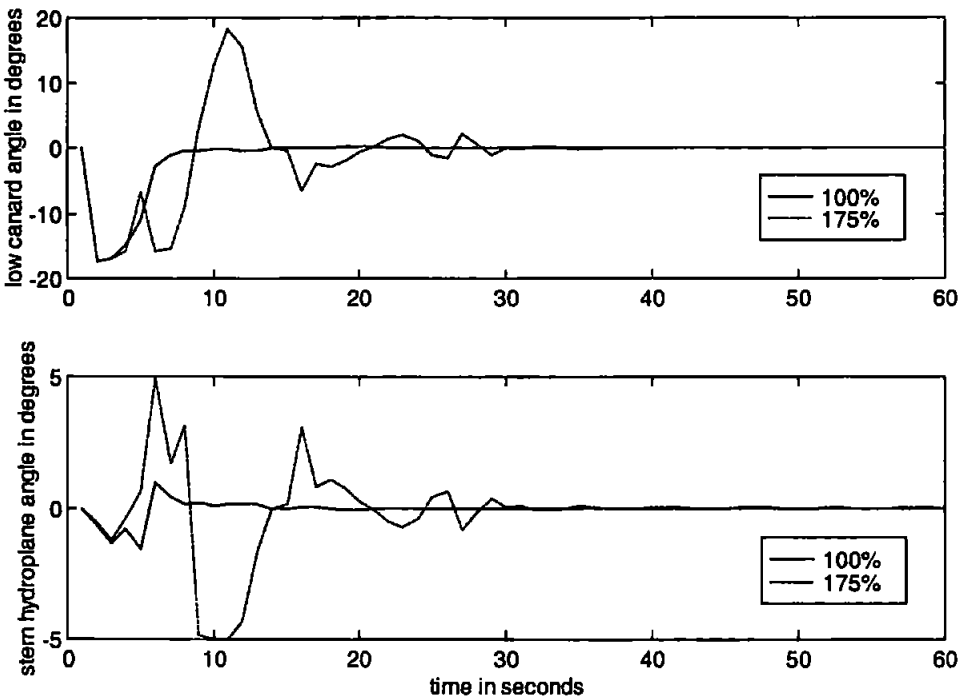


Figure 6.20: Mass variation during a 40° course-change when employing the on-line CANFIS autopilot – low canard rudder and stern hydroplane responses.

Evidently, the on-line CANFIS autopilot performs poorly in light of the increased AUV payload. The depleted damping of the yaw dynamics, arising from the increased AUV mass, leads to oscillatory responses in the yaw and roll channels under this control strategy. In an effort to remedy these poor responses, the step size of the parameter transitions was increased to 20%; this was the lowest integer value to effect significant improvements in AUV course-changing stability. The parameters were consequently allowed to vary more significantly at each transition. It was felt at this stage that this approach would enable the control system to compensate more quickly for the large overshoots involved in Figure 6.19. The results of this experiment are reproduced in Figures 6.21 and 6.22.

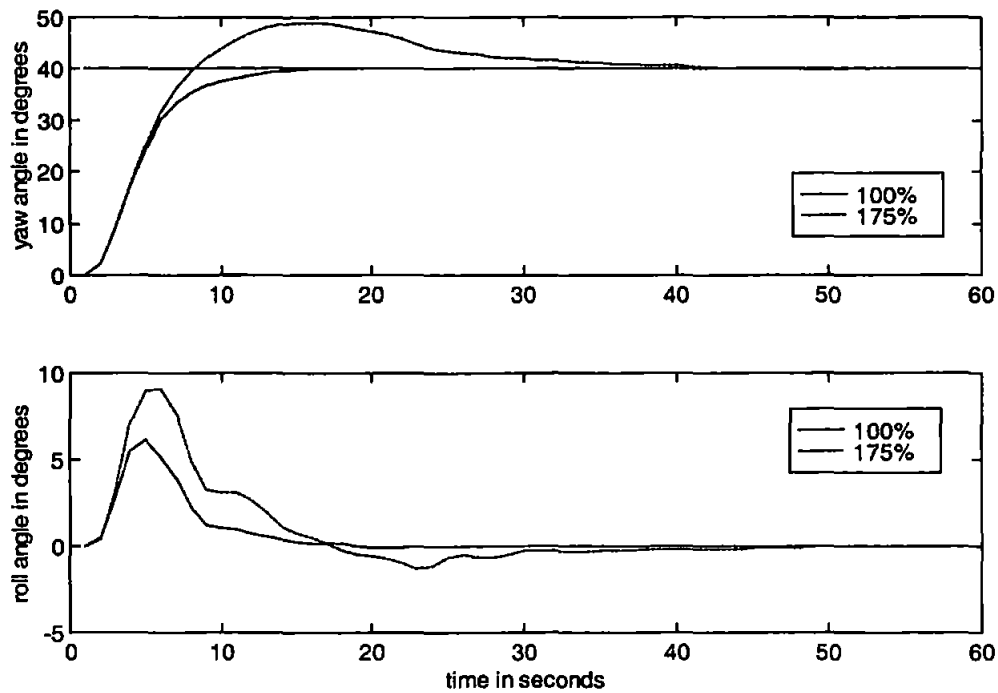


Figure 6.21: Mass variation during a 40° course-change when employing the CANFIS autopilot with a step size transition rate of 20%– yaw and roll responses.

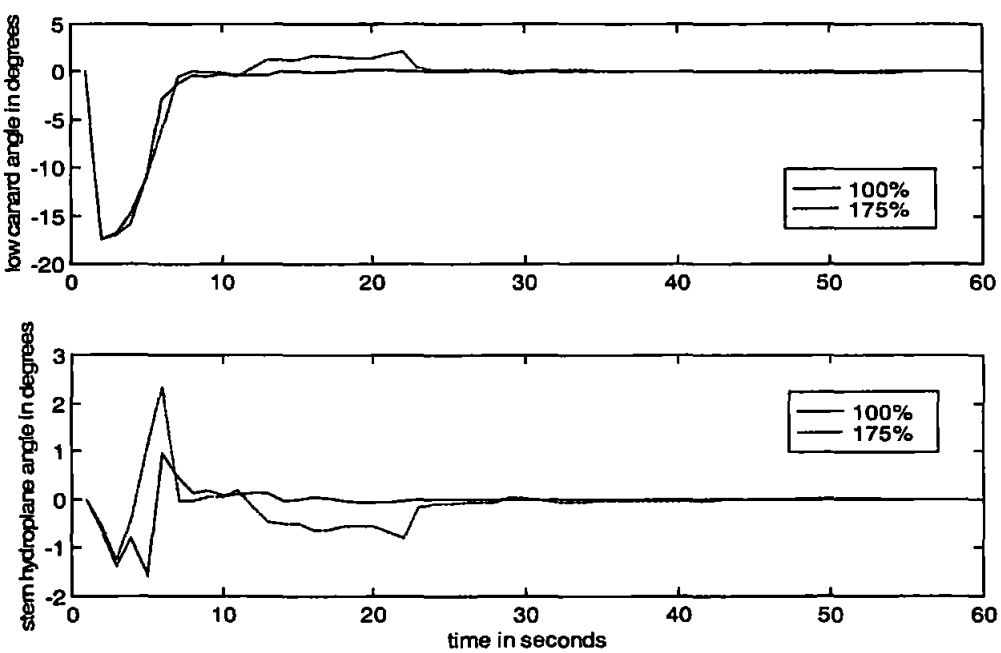


Figure 6.22: Mass variation during a 40° course-change when employing the CANFIS autopilot with a step size transition rate of 20%– low canard and stem hydroplane responses.

The premise parameter step size transition rate of 20% clearly invokes an improvement in the course-changing effectiveness of the on-line autopilot system. However, these results should be treated with some scepticism; the introduction of such a high transition rate can lead to parameter oscillation as highlighted in the simulations pertaining to Figures 6.8 – 6.10.

Additionally, the following perturbations in the hydrodynamic coefficients are to be considered, based upon the dependence of the hydrodynamic coefficients on the total velocity squared (Eqn.(4.42)):

- $\pm$ 20% variation in Y_{UV}, the sway damping coefficient
- $\pm$ 20% variation in Y_{UR}, the yaw into sway coefficient
- $\pm$ 20% variation in K_{UV}, the sway into roll coefficient
- $\pm$ 20% variation in K_{UR}, the yaw into sway coefficient
- $\pm$ 20% variation in K_{UP}, the roll damping coefficient
- $\pm$ 20% variation in N_{UV}, the sway into yaw coefficient
- $\pm$ 20% variation in N_{VR}, the roll into yaw coefficient
- $\pm$ 20% variation in N_{UR}, the yaw damping coefficient

The responses of the AUV to a 40° course-changing manoeuvre when employing the CANFIS autopilot under these coefficient variations are illustrated in Figures 6.23 to 6.30.

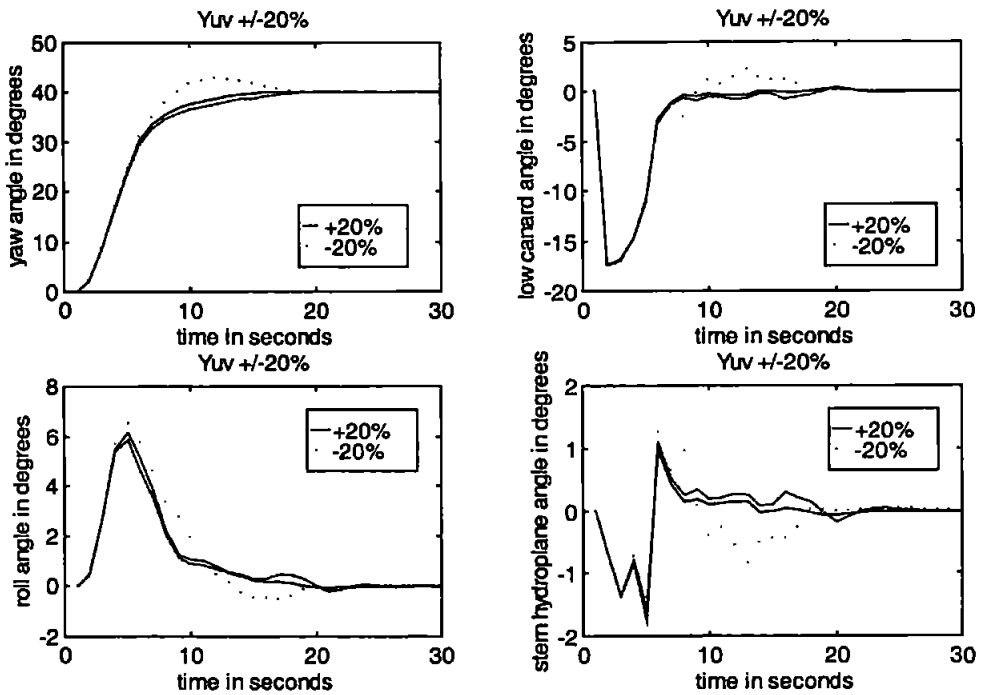


Figure 6.23: Varying Y_{uv} hydrodynamic coefficient during a 40° course-change when employing the on-line multivariable CANFIS autopilot.

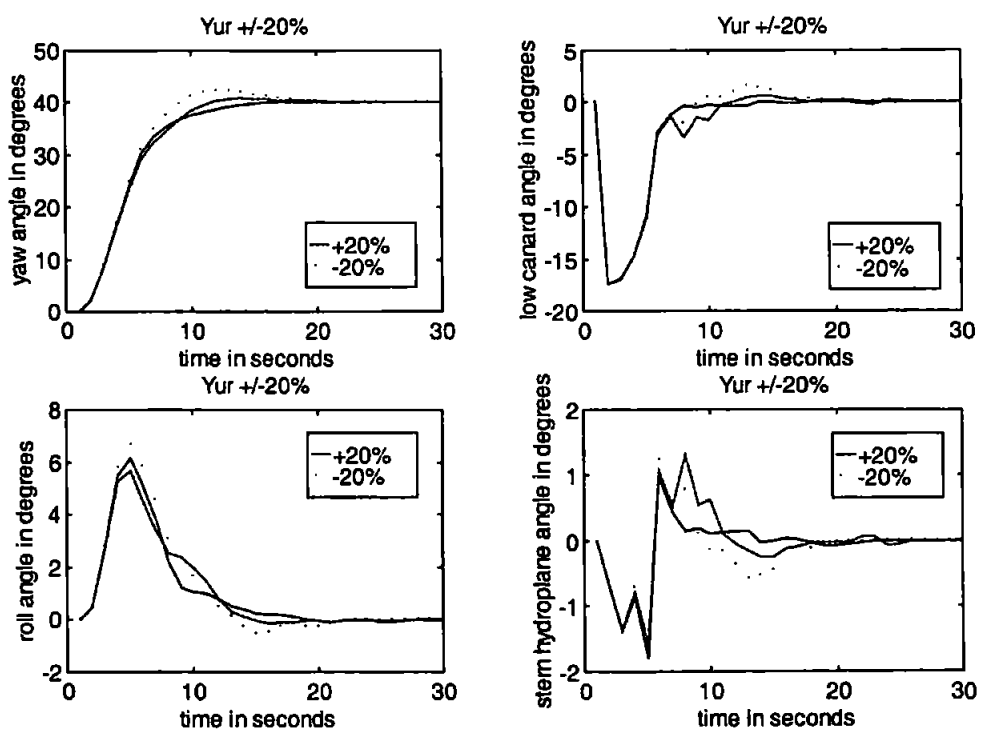


Figure 6.24: Varying Y_{ur} hydrodynamic coefficient during a 40° course-change when employing the on-line multivariable CANFIS autopilot.

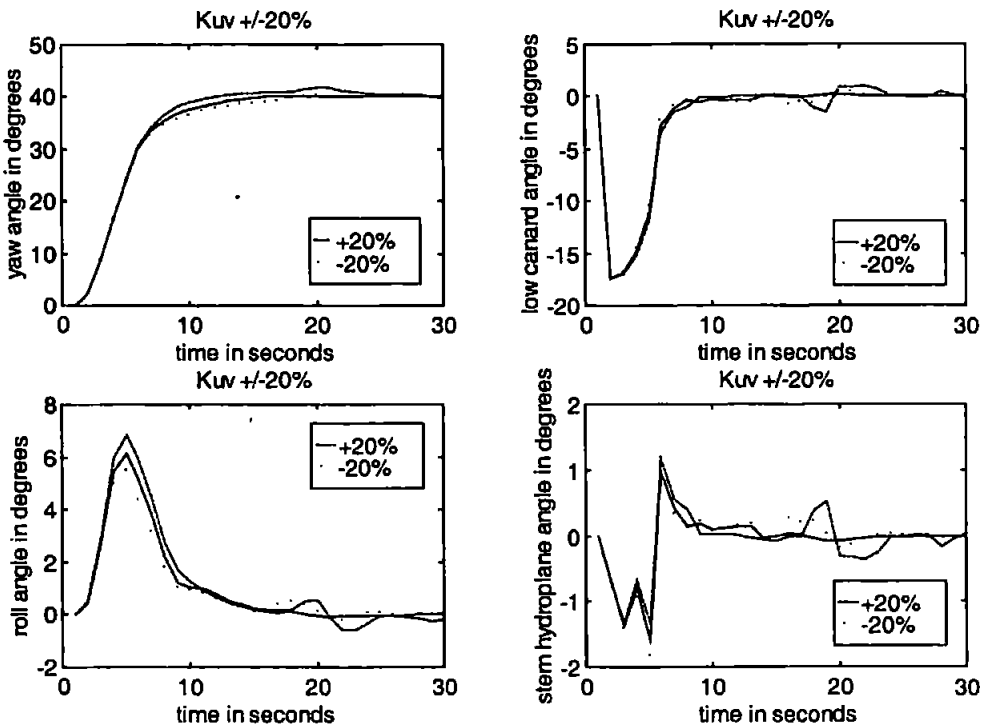


Figure 6.25: Varying K_{uv} hydrodynamic coefficient during a 40° course-change when employing the on-line multivariable CANFIS autopilot.

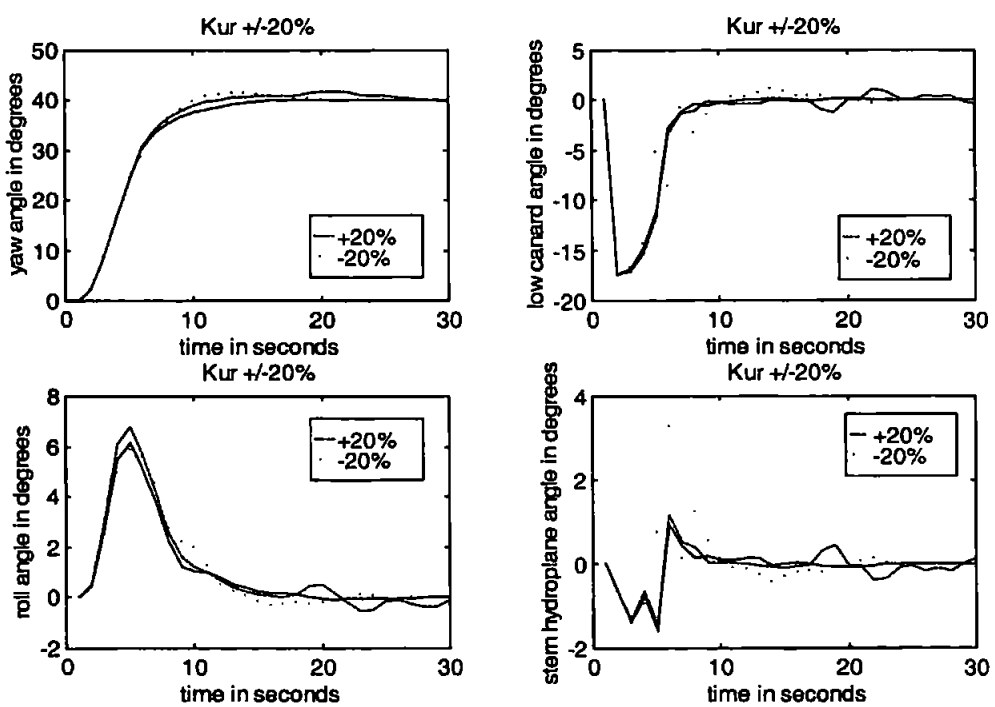


Figure 6.26: Varying K_{ur} hydrodynamic coefficient during a 40° course-change when employing the on-line multivariable CANFIS autopilot.

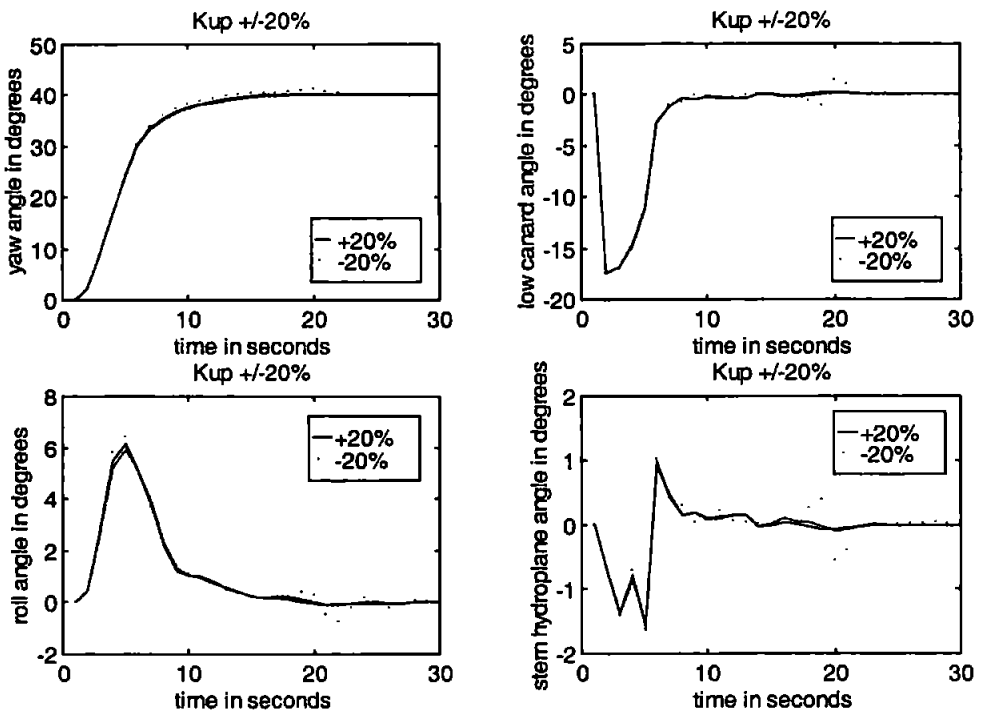


Figure 6.27: Varying K_{up} hydrodynamic coefficient during a 40° course-change when employing the on-line multivariable CANFIS autopilot.

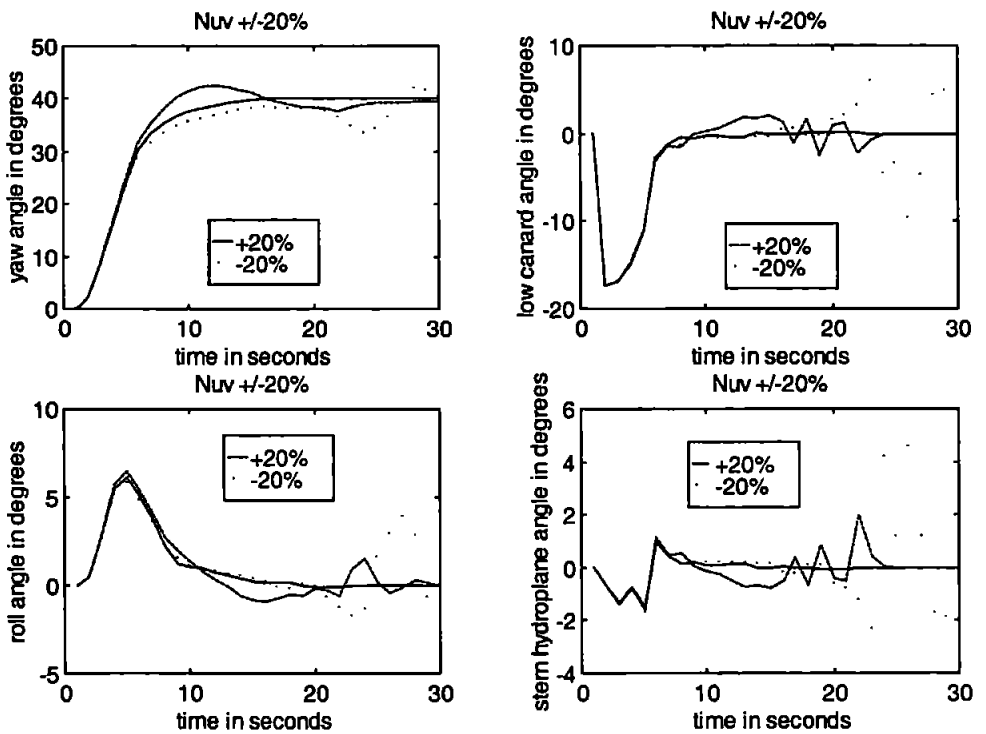


Figure 6.28: Varying N_{uv} hydrodynamic coefficient during a 40° course-change when employing the on-line multivariable CANFIS autopilot.

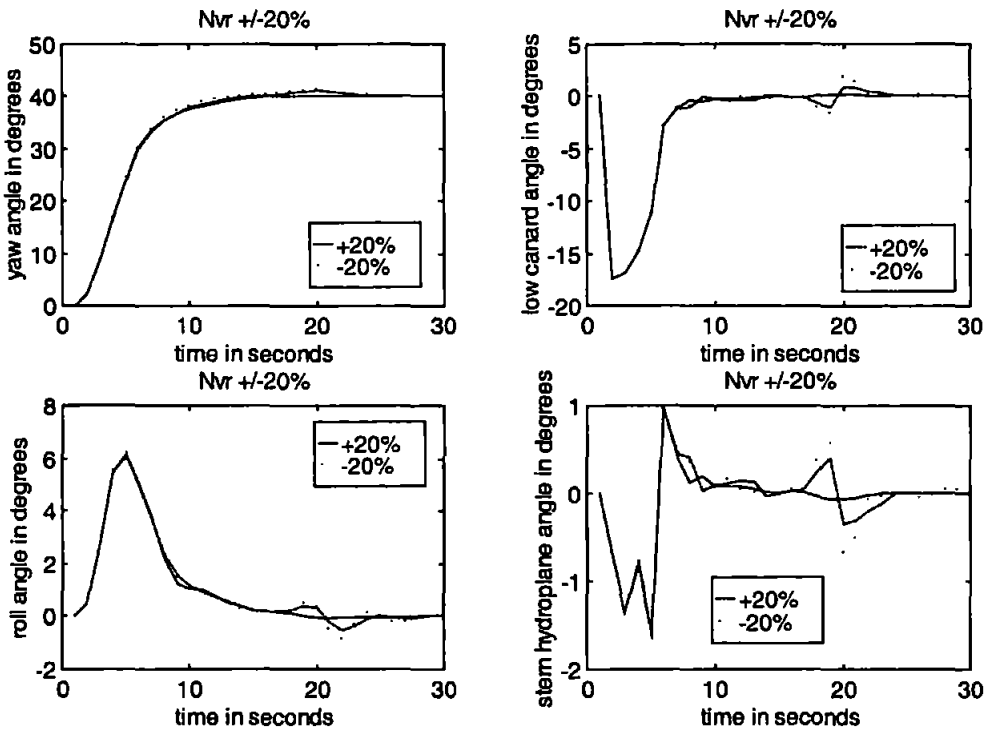


Figure 6.29: Varying N_{vr} hydrodynamic coefficient during a 40° course-change when employing the on-line multivariable CANFIS autopilot.

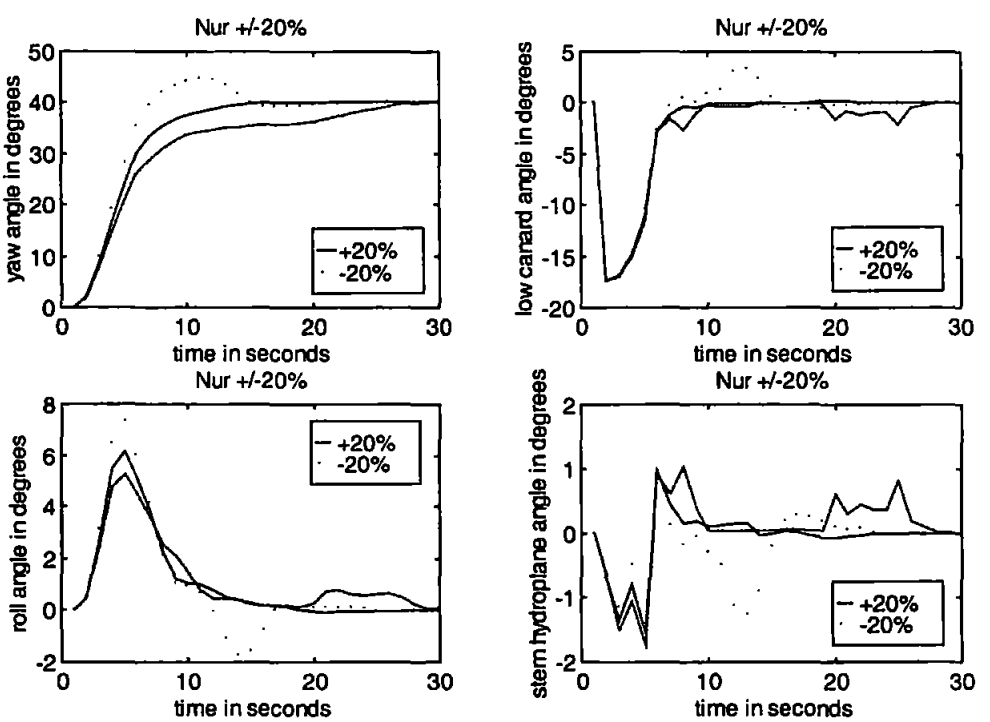


Figure 6.30: Varying N_{ur} hydrodynamic coefficient during a 40° course-change when employing the on-line multivariable CANFIS autopilot.

The on-line CANFIS autopilot strategy does not accommodate the parameter variations in such a robust manner as the off-line tuned CANFIS autopilot. Although none of the robustness tests resulted in unstable behaviour, oscillatory motion is evident in Figures 6.23, 6.26, 6.28 and 6.30. The use of the on-line adaptation algorithm causes a conflict of interests within the parameter tuning regime; whilst on-line parameter adaptation is required to improve autopilot effectiveness, the parameters must be restricted to vary less sharply.

Despite the ability of the presented on-line algorithm (Table 6.1) to control the AUV effectively, the robustness experiments pursued within this section have illustrated that the restriction of the parameter set to adaptation during transient motion limits the robustness of the resulting control algorithm.

6.5.2 Line of Sight Guidance

Autonomous guidance of the vehicle is again achieved by employing the LOS algorithm. The verification track was employed to appraise the robustness properties of the novel on-line CANFIS multivariable autopilot against the off-line multivariable CANFIS autopilot of Chapter 5; the radius of acceptance β was again fixed at 15 metres and the nominal vehicle speed at 7.5-knots.

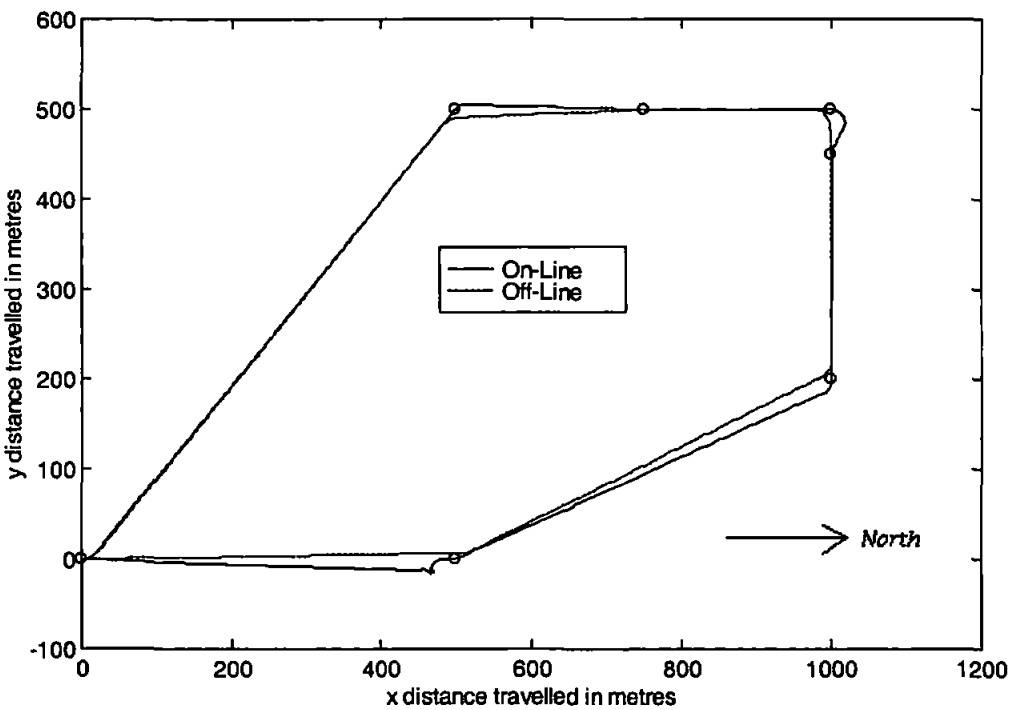


Figure 6.31: Line of sight responses over the verification track in the absence of current disturbances.

Simulating the AUV at 7.5-knots over the verification course in the absence of sea current disturbances produced the course following responses of Figure 6.31. The response when employing the off-line multivariable autopilot is more accurate and less oscillatory than when employing the on-line autopilot; the on-line autopilot becomes unstable when the line of sight algorithm selects the final way-point. This is re-iterated

by the yaw response of Figure 6.32 which shows how the on-line autopilot selects a heading of +180° when clearly a heading of -180° would achieve the desired course and would involve a smaller rudder demand. The low canard rudder and stern hydroplane responses of Figure 6.33 illustrate the saturation of the rudders and hydroplanes at this point in the simulation, respectively.

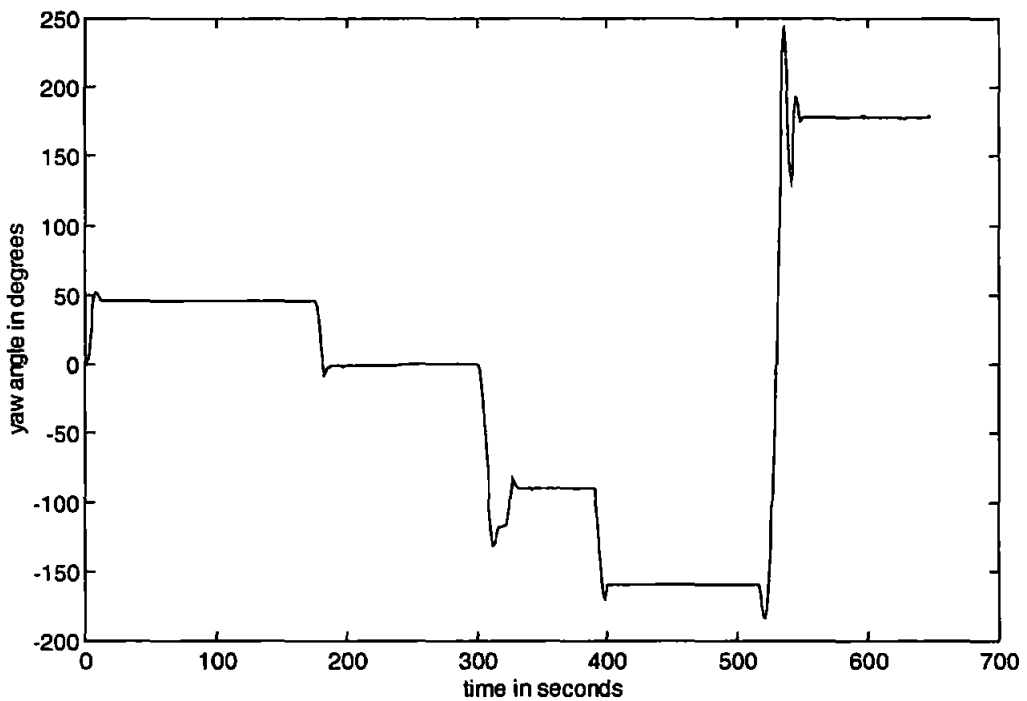


Figure 6.32: Yaw angle over the verification track in the absence of current disturbances.

Further experiments were undertaken to assess the relative performance of the on-line multivariable autopilot against the off-line multivariable autopilot strategy. The results of these simulations are detailed in Figures 6.34 to 6.39 and highlight the unsuitability of the on-line autopilot in the presence of sea currents.

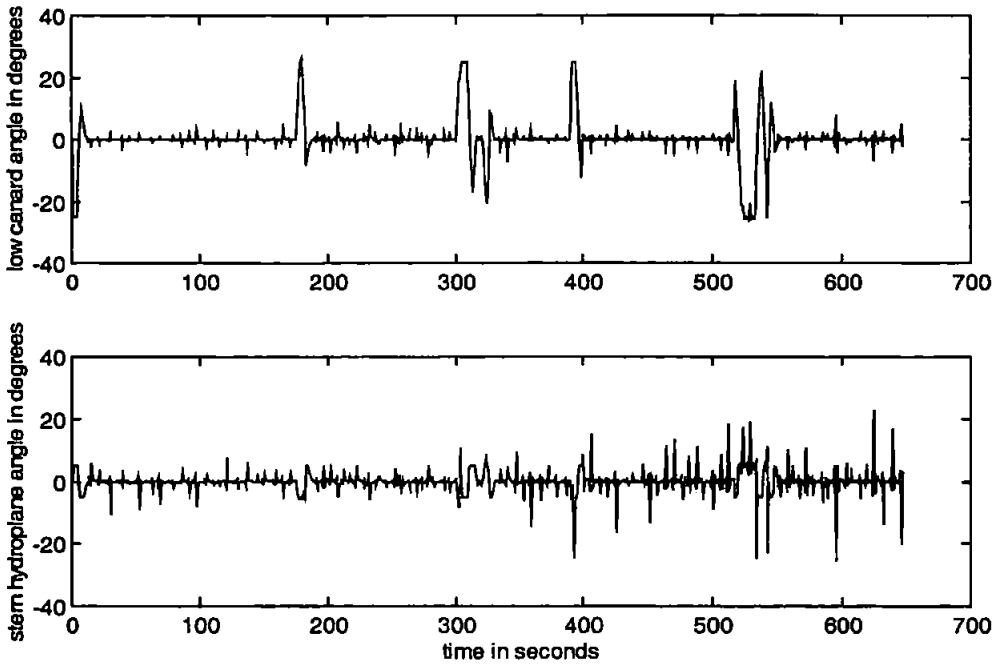


Figure 6.33: Low canard and stern hydroplane angles over the verification track in the absence of current disturbances.

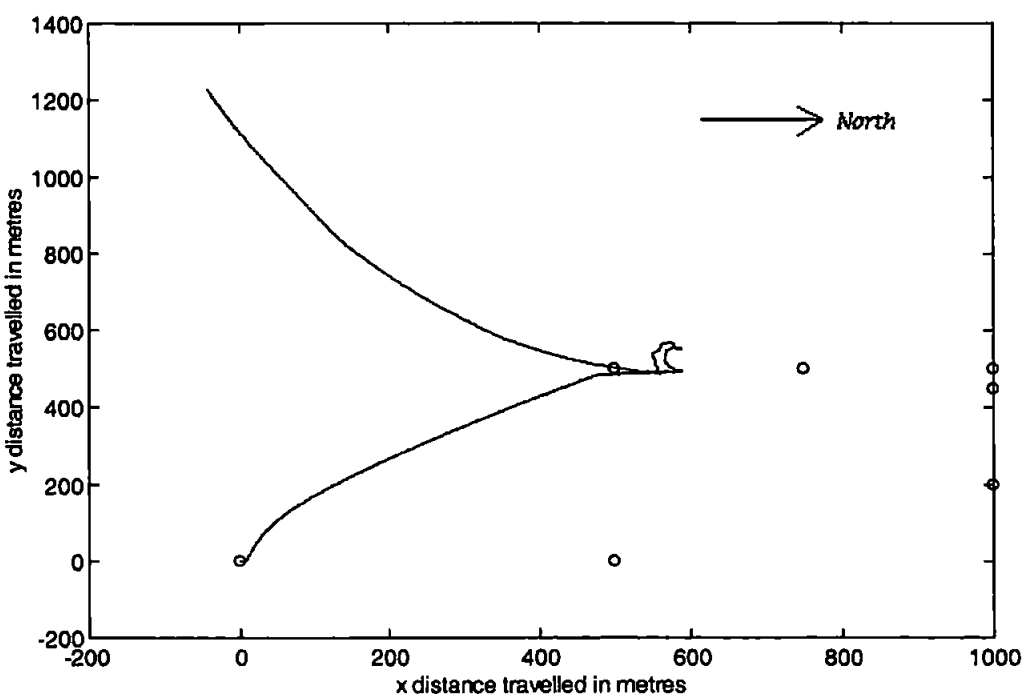


Figure 6.34: Line of sight response over the verification track in the presence of a current disturbance of 2.5 ms^{-1} along the Northerly axis.

Clearly, the introduction of a Northerly current of 2.5 ms^{-1} (as shown in Figure 6.34) renders the on-line autopilot unstable. Also, the introduction of a of 2.5 ms^{-1} current along the Westerly axis (Figure 6.35) produces unstable results when employing the multivariable on-line autopilot. Closer examination of the autopilot parameter set for each simulation revealed the oscillatory transition of both the premise and consequent parameters. Indeed, a tolerance setting of less than 0.04° was required to stabilize the modified on-line algorithm of Table 6.1, which is unachievable with respect to current compass tolerance levels.

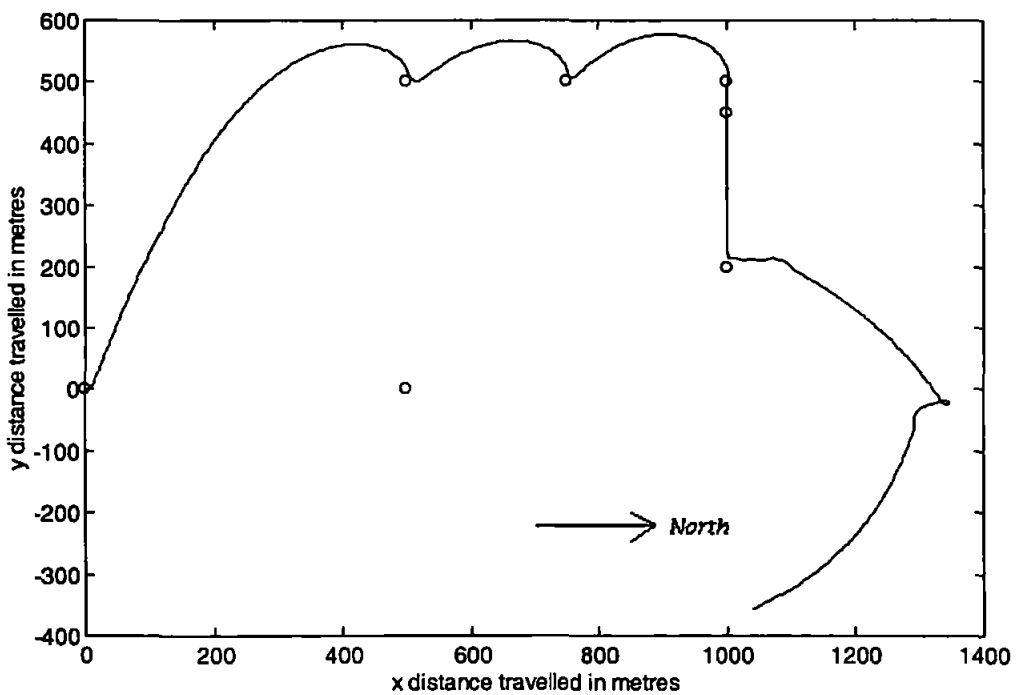


Figure 6.35: Line of sight response over the verification track in the presence of a current disturbance of 3 ms^{-1} along the Westerly axis.

6.5.3 Measurement Noise

To provide a final assessment of the effectiveness of the presented on-line neuro-fuzzy control scheme, comparisons were made with the off-line CANFIS autopilot of Chapter 5 in the presence of measurement noise. It should be noted that the CANFIS off-line

autopilot system was appropriately de-tuned to allow ease of comparison between the two control systems.

The following results illustrate the course-changing and roll-minimizing ability of the on-line and off-line CANFIS autopilots in the presence of 1%, 5% and 10% signal to noise ratio (SNR); these values represent a peak noise level of 0.25°, 1.25° and 2.5° with respect to the maximum canard rudder angle of 25°. Figure 6.36 illustrates these noise levels over a 100 second period.

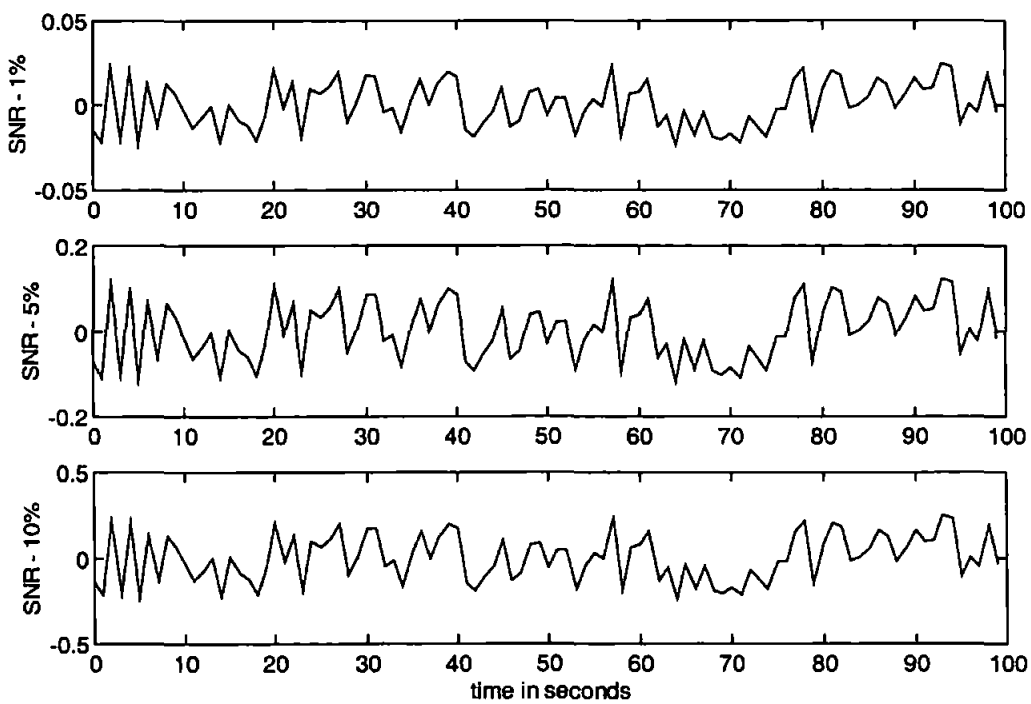


Figure 6.36: Noise sequences at 1%, 5% and 10% SNR.

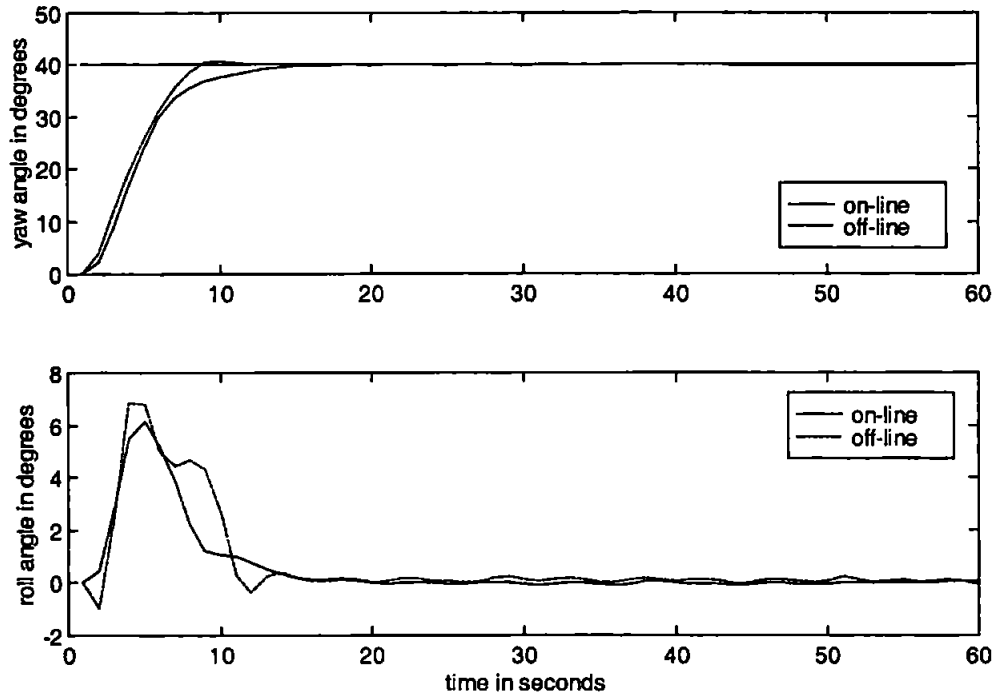


Figure 6.37: Yaw and Roll responses for the on-line and off-line autopilots in the presence of a 1% SNR.

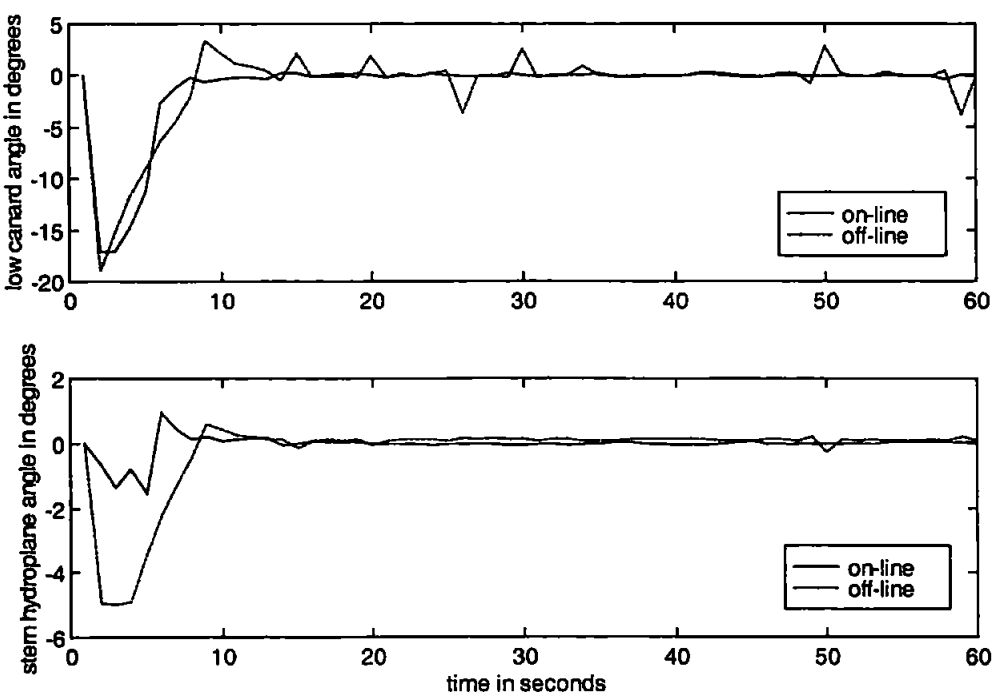


Figure 6.38: Low canard and stern hydroplane responses for the on-line and off-line autopilots in the presence of a 1% SNR.

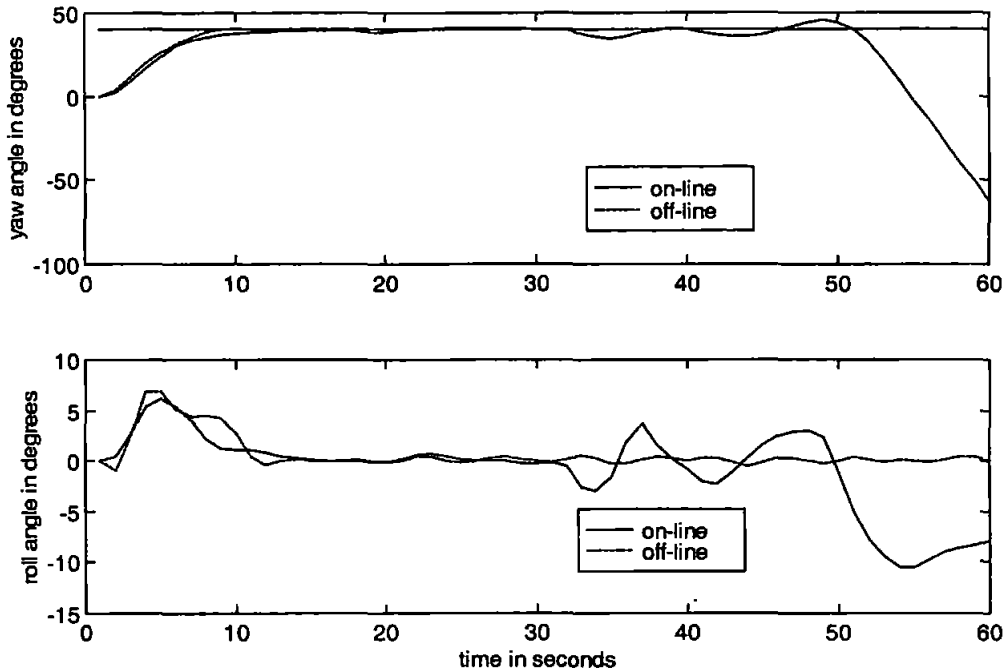


Figure 6.39: Yaw and Roll responses for the on-line and off-line autopilots in the presence of a 5% SNR.

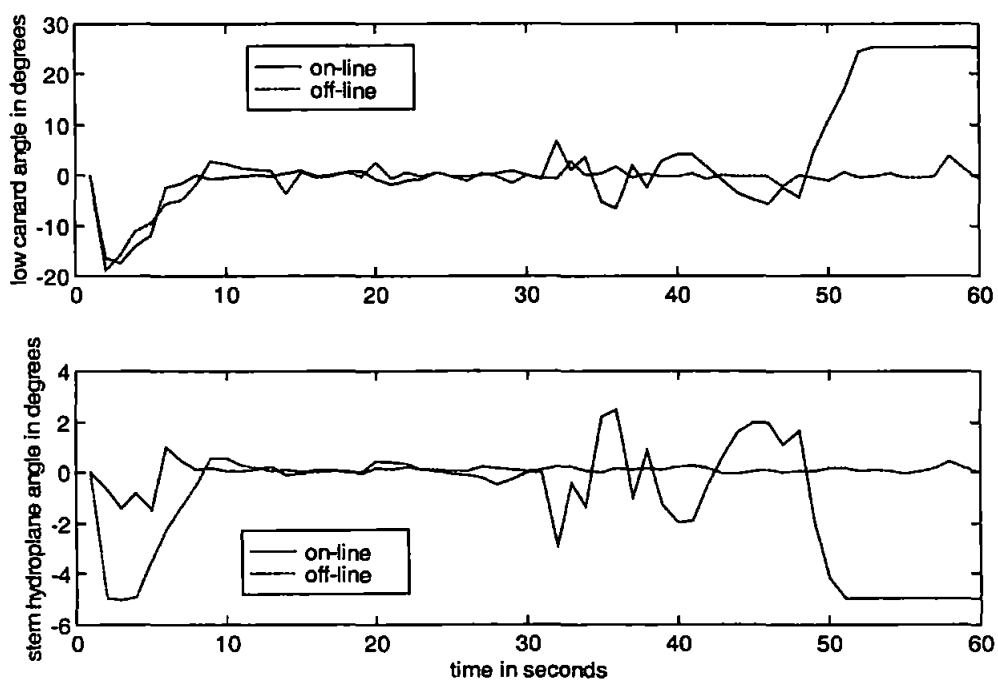


Figure 6.40: Low canard and stern hydroplane responses for the on-line and off-line autopilots in the presence of a 5% SNR.

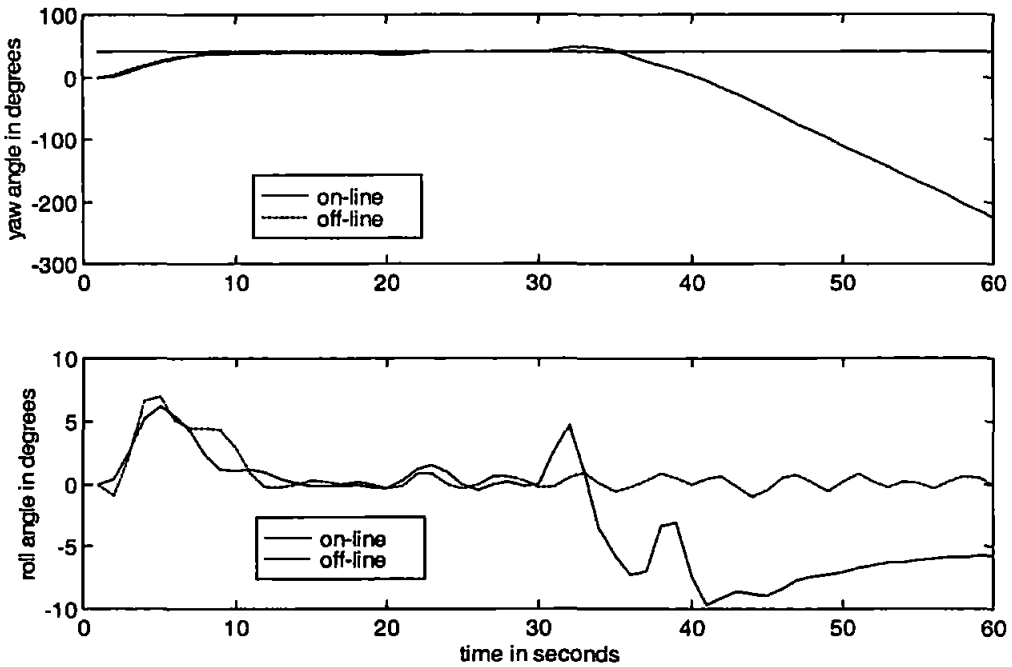


Figure 6.41: Yaw and Roll responses for the on-line and off-line autopilots in the presence of a 10% SNR.

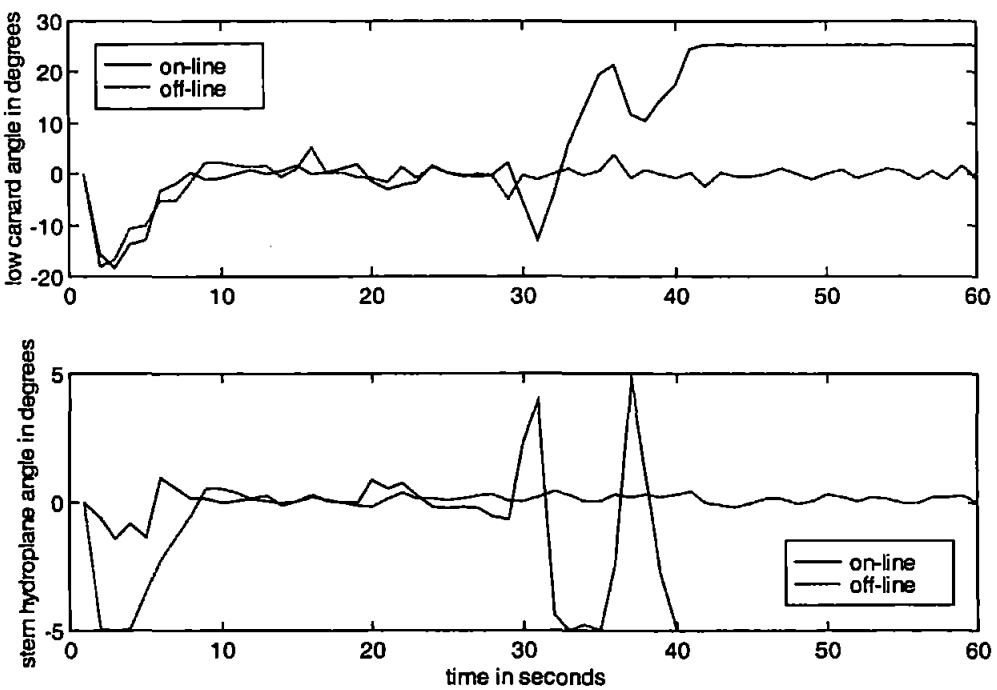


Figure 6.42: Low canard and stern hydroplane responses for the on-line and off-line autopilots in the presence of a 10% SNR.

Figure 6.37 depicts the course-changing and roll-minimizing responses of the AUV for both the on-line and off-line autopilot systems in the presence of a 1% SNR. The corresponding low canard rudder and stern hydroplane responses are reproduced in Figure 6.38. Both autopilot systems guide the AUV successfully onto the desired 40° course-change. However, the low canard rudder response of the off-line autopilot displays some oscillation in comparison to the on-line autopilot response, which provides smooth transition. This suggests that the sensitivity of the on-line autopilot is appropriate to accommodate slight variations in the control signal.

The responses of each autopilot system for a 5% SNR are detailed in Figures 6.39 and 6.40. The increase in the level of measurement noise causes instability in the on-line control algorithm; the low canard rudder and stern hydroplane responses are highly pronounced in comparison to those of the off-line autopilot. Indeed, varying the forgetting factor and step size transition rate did not improve the performance of the autopilot significantly. This instability is caused by the rapid variation of the fuzzy parameter set; sharp changes in the noise signal cause the parameter set to vary over larger ranges than in previous experiments. Conversely, the off-line autopilot coped well with the 5% SNR, providing accurate course-changing and roll-minimization.

In light of the failure of the on-line autopilot to control the AUV in the presence of a 5% SNR, it was anticipated that a 10% SNR would be likely to cause similar if not exaggerated effects. Figures 6.41 and 6.42 illustrate that this hypothesis is borne out. The on-line multivariable autopilot yields disappointing responses, whilst the off-line multivariable autopilot again provides accurate control.

It is evident from these simulations that the off-line tuned multivariable fuzzy autopilot can cope effectively with a SNR of 10%. It is also apparent that the equivalent on-line tuned autopilot cannot. Examination of the parameter sets pertaining to Figures 6.37 to 6.42 highlight once again the propensity for the parameters of the on-line autopilot to vary significantly in the presence of such disturbances.

6.6 Concluding Remarks

The previous chapters have discussed the design and implementation of a neural network architecture for tuning the parameter set of a fuzzy autopilot. Consequently, the resulting autopilots were purely fuzzy. The resulting autopilots in this chapter were considered to be of neuro-fuzzy type as the adaptation was performed at discrete sampling intervals during each simulation.

The novel application of the on-line hybrid rule of ANFIS to AUV autopilot design produced effective results and prompted research into the use of CANFIS for on-line simulations to control the yaw and roll simultaneously. Despite the ability of the presented novel on-line algorithm (Table 6.1) to control the AUV effectively, robustness experiments illustrated that the variation of the parameter set to adaptation during transient motion limited the robustness of the resulting autopilot. Thus the following chapter discusses autopilot simulations which continue in the vein of multivariable off-line fuzzy autopilot tuning as this technique proved more effective than on-line adaptation.

The work within this chapter represents a novel application of both the ANFIS and CANFIS structures, to the best of the present author's knowledge.

References

Corradini, M. L. and Orlando, G. (1997). A Discrete Adaptive Variable-Structure Controller for MIMO Systems and Its Application to an Underwater ROV. IEEE Transactions on Control Technology, Vol. 5, No. 3, pp349-359.

Ishii, K., Fujii, T. and Ura, T. (1993). An On-Line Adaptation Method in a Neural Network Based Control System for AUVs. IEEE Journal of Oceanic Engineering. Vol. 20, No. 3, pp221-228.

Jang, J.-S. R. (1992). Self-Learning Fuzzy Controller based on Temporal Back-Propagation. IEEE Transactions on Neural Networks, Vol. 3, pp714-723.

Juang, C.-F. and Lin, C.-T. (1998). An On-Line Self Constructing Neural Fuzzy Inference Network and Its Applications. . IEEE Transactions on Fuzzy Systems, Vol. 6, No. 1, pp12-32.

Kwiesielewicz, M., Craven, P. J. and Sutton, R. (1997). Modelling the Yaw Behaviour of a UUV using a Dynamic Neural Network. Proceedings of the 12th International Conference on Systems Engineering, Coventry, U.K., Vol. 2, pp410-413.

Mizutani, E. and Jang, J.-S. R. (1995). Co-Active Neuro-Fuzzy Modelling. Proceedings of the International Conference on Neural Networks, Perth, Western Australia, Australia, pp760-765.

Takagi, T. and Sugeno, M. (1985). Fuzzy Identification of Systems and its Applications to Modelling and Control. IEEE Transactions on Systems, Man and Cybernetics, Vol. 15, pp116-132.

Venugopal, K.P., Sudhakar, R. and Pandya, A.S. (1992). On-Line Learning Control of Autonomous Underwater Vehicles using Feedforward Neural Networks. IEEE Journal of Oceanic Engineering. Vol. 17, No. 4, pp308-319.

Yuh, J. (1990). A Neural Network Controller for Underwater Robotic Vehicles. IEEE Journal of Oceanic Engineering. Vol. 15, No. 3. pp161-166.

Chapter 7

Non-Linear Consequent Models for Fuzzy Autopilot Design

7.1 Introduction

The autopilot designs of the preceeding three chapters have shown the attractiveness of fusing fuzzy logic with neural network architectures in the design process. The resulting controllers are linguistically interpretable in comparison to the black-box structure of a typical neural network. Additionally, these designs retain their learning and adaptation capabilities which allow the topographic fitting of the non-linear function representing the process input-output behaviour. Hardy (1971) described the process of fitting a topographic surface to a given set of data points as

"given a set of discrete data on a topographic surface, reduce it to a satisfactory continuous function representing the topographic surface."

This empirical modelling is directly analogous to the process of tuning a fuzzy inference system (FIS) to reproduce a particular output, given specific inputs. When tuning the membership functions of a FIS, a set of representative training data is collected over a number of pre-chosen variables, and is subsequently presented to the encoded FIS architecture. The parameters of the fuzzy premise (input) and consequent (output) functions within this architecture then adapt in order that the future presentation of a particular set of input data pairs produces the required output. Clearly, this training data represents known points on the topographic surface that is to be modelled; the ability of the resulting network to interpolate between these training data points is then regarded as a measure of network generalization.

In recent years, the theory of radial basis function (RBF) approximations to topographic surfaces has been established and shown to provide an excellent framework for modelling smooth non-linear functions. When considered as a network architecture, RBF approximations employ a linear combination of non-linear basis functions, each of which is defined within a particular operating region. Thus the overall network output is a linear combination of local network responses. Consequently, such RBF networks can be referred to as non-linear gain-schedulers when employed in a control context.

The similarities between such an approach and the adaptive network-based fuzzy inference system (ANFIS) architecture of Chapter 4 are clear. Indeed, under certain conditions the two approaches become identical.

This chapter introduces a novel extension to the ANFIS regime, which employs the fuzzy concepts developed within ANFIS in conjunction with a RBF model for the inference system's consequent functions. The aim of such a fusion is to retain the linguistic interpretability of ANFIS whilst exploiting the functional non-linearity of such RBF local modelling techniques. The resulting fuzzy RBF network will be transparent in structure and easy to examine.

7.2 The ANFIS Approach to Function Modelling

More specifically the FISs employed until now have all been variants of the Takagi-Sugeno-Kang (TSK) (Takagi and Sugeno (1985)) model discussed in Chapter 4, where the i_{th} fuzzy rule is of the form (for two inputs):

$$If \;\; x_1 \;\; is \;\; A_i \;\; and \;\; x_2 \;\; is \;\; B_j \;\; then \;\; f_k = f(x_1, x_2) \tag{7.1}$$

for:

$i = 1,2,...,m$ (the number of input membership functions on x_1),

$j = 1,2,...,n$ (the number of input membership functions on x_2), and

$k = 1,2,...,mn$.

The f_k's are typically taken as a linear function of the input variables for a TSK FIS. These rule outputs perform linear interpolation of the function $f(\underline{x})$ describing the input-output behaviour of the underlying model and are written as

$$f = \sum_{k=1}^{mn} \sum_{j=1}^{n} a_{kj} x_j + b_k \tag{7.2}$$

where f is a vector of *dependent* variables, the x_j's *are the independent* variables and n represents the number of independent variables. In terms of a FIS, each linear rule can be envisaged as a moving singleton spike, its position in n-dimensional space being determined by the values of the input variables x_j.

Figure 7.1 provides an illustration of a FIS of this form, where one input variable is mapped onto one output variable via four fuzzy rules. Each individual linear rule is effective within a certain region. However, certain functions can prove difficult to model, and some mismatch between the model and the underlying function is not

unusual, especially at the points where rules cross-over. Indeed, this input-output function could be particularly irregular with sharp gradients.

When approximating a non-linear function, one method to diminish the modelling error between local models and the actual function may be to increase the number of fuzzy rules mapping the function from input to output domains. Consequently, the width of each interval is reduced, as is the error incurred at the interval cross-over points. However, this leads to an increase in the number of parameters and inevitably lengthens the training period when adjusting these parameters to make the fuzzy model fit the desired function more closely.

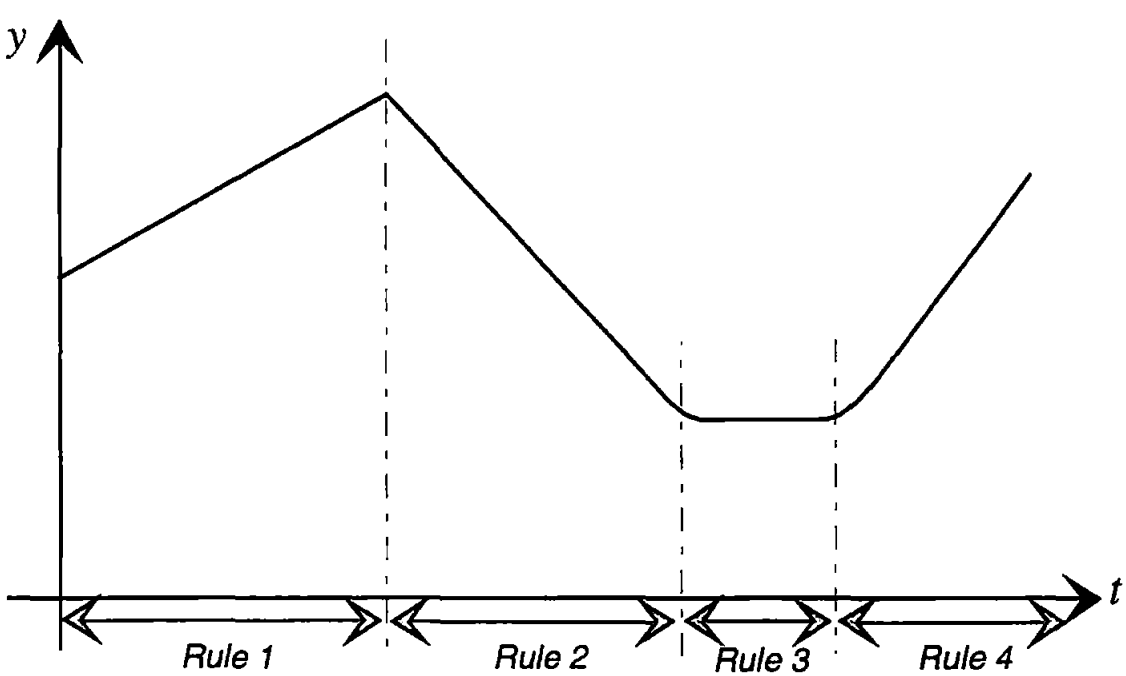


Figure 7.1: Piecewise interpolation of a non-linear function using linear rule outputs.

Alternatively, the linear functions employed in a typical TSK FIS output space can be replaced with non-linear functions that can better approximate the required input-output behaviour. Piecewise polynomials are often chosen for this purpose. This again implies that $\Re^n$ has to be divided into suitable regions for which a particular polynomial applies. Each polynomial must then be pieced together to provide continuity over the approximation, yet even polynomial function approximations are not always effective at modelling the somewhat sharp variations in many real topographic surfaces.

One particular type of function that has been extensively studied takes the form

$$f_j = \sum_{k=1}^{n} \eta_k \xi_k(\underline{x}) \tag{7.3}$$

where $\xi_k(\underline{x})$ is a non-linear function of the independent variables x_k, and the η_k's are unknown parameters. This type of model defines a non-linear relationship between the dependent variable f_j and the independent variables x_k, and defines a linear relationship between f_j and the unknown parameters η_k. One clear advantage of employing this type of non-linear model is that the unknown parameters η_k can still be computed by a linear technique, as used in the hybrid learning rule of ANFIS.

Jang et *al.* (1997) employed sigmoidal functions of the form

$$\xi_k(\underline{x}) = \frac{1}{1 + \exp\left[-\left(p_k x_1 + q_k x_2 + \ldots + v_k\right)\right]} \tag{7.4}$$

as their ANFIS consequents in contrast to the typically documented linear functions. This leads to a revised ANFIS structure that employs non-linear consequents of sigmoidal form and thus non-linear fuzzy rule outputs whose weights can still be found by linear algorithm approaches. The advantage of the chosen consequent functions is in

their similarity to the original consequents of a typical ANFIS model, producing a non-linear structure that requires only a simple modification of the original ANFIS modelling regime.

Conversely, White and Sofge (1992) state how the non-local nature of the commonly employed sigmoidal function can lead to neural networks that experience learning problems. In essence, each sigmoid does not relate to a specific region of the input space and thus incremental learning of the network parameters can yield conflicts between minimizing the network cost function and retaining the knowledge already stored within the network structure. That is, the use of the sigmoid neuron does not usually represent an overall network that can exhibit spatially localized learning properties. Various approaches have been documented to attempt to overcome these problems, such as local batch learning, very slow learning rates and distributed (uncorrelated) input sequences. However, these learning improvements were not documented or employed by Jang et *al.* (1997).

Clearly in control applications the local model should represent the actual system within the desired range of influence, and therefore be a good approximation to the system locally (Hunt and Johansen (1997)). Bossley (1997) highlights that *B*-spline membership functions, which have been used extensively in neurofuzzy modelling by Brown and Harris (1994), are not suited to locally modelling this relationship. A more suitable choice is often trapezoids, as in the work of Lin and Juang (1997) or Gaussian functions, which are constant for the majority of their response but represent the system over the desired operating region. This requires that these functions be positioned in the areas of the output space $\mathcal{R}^n$ which are most heavily populated by training data points.

Moreover, Poggio and Girosi (1990a) report the superior approximation power of neural network architectures containing RBFs over those using layers of sigmoidal functions. Indeed they provide a technical note on the different types of basis functions

that can be employed; a review of RBF approximations to non-linear modelling of topographic surfaces is provided in the following section.

These and other studies (Poggio and Girosi (1989), Jang, et *al.* (1997), Heiss and Kampl (1996)) have lead to the consideration of composite Gaussian functions as the consequent equations herein for non-linear rule implementation within an ANFIS type architecture. Subsequent sections within this chapter therefore lend themselves to an in-depth discussion of this proposal.

7.3 Topographic Approximations using Radial Basis Function Techniques

Hardy (1971) employed multiquadric functions, typically in conical form, to successfully model topographic surfaces. This modelling approach was compared to more traditional Fourier and polynomial series approximations, which rely on large amounts of data and are thus inefficient to implement. The multiquadric approximations also highlighted the deficiencies in the more traditional approaches, which resulted in oscillatory and inaccurate modelling. Hence, the paper concluded that the RBF method could accurately approximate the given contours with limited data points, a definite advantage over traditional techniques. Since this pioneering work, various applications of RBF techniques to function approximation have appeared in the literature. The main ones of interest are presented hereafter.

Powell (1992) more recently discussed the use of RBFs as an alternative means of achieving accurate approximations to topographic surfaces. By formally considering the extension to several variables of univariate spline functions, the paper highlights the intuitive nature of RBFs for such mappings and gives a list of functions that are often used. Needless to say, each function has inherent properties that are useful for specific

applications. A brief introduction to RBFs is now presented. The interpolation problem considered here can be written mathematically, as follows:

Given a set of m distinct data points in vector form, $\{\underline{x}_k ; k=1,2,...,m\}$ *in* $\Re^n$ *and m real numbers* $\{f_k ; k=1,2,...,m\}$ *in the form* $\{(x_k, f_k)\}$, *choose a function* $F: \Re^n \rightarrow \Re$ *which satisfies the interpolation conditions*

$$F(\underline{x}_k) = f_k; \qquad k=1,2,...,m \tag{7.5}$$

where the function F must pass through all data points.

The RBF technique solves the interpolation problem by forming a set of linear equations consisting of basis functions that are specific to certain regions of $\Re^n$. By employing an arbitrary distance measure (usually a Euclidean norm), the relative distances of known data points within the *n*-dimensional space $\Re^n$ produced by the given set of *independent* variables can be computed. This dependence is due to the use of RBFs of the form: $\xi(\|\underline{x} - \underline{c}_k\|)$, where $\underline{x}, \underline{c}_k \in \Re^n$ and $k=1,2,...,m$. The vectors $\underline{c}_k$ are the centres of the basis functions with dimension $\Re^n$. The RBF approach therefore suggests the interpolating functions to be of the form:

$$f(x) = \sum_{k=1}^{m} \eta_k \xi(\|\underline{x} - \underline{c}_k\|), \qquad \underline{x}, \underline{c}_k \in \Re^n, \quad k=1,2,...,m. \tag{7.6}$$

Substituting Eqn.(7.5) into Eqn.(7.6), yields the following set of linear equations for the coefficients η_k:

$$\begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_m \end{pmatrix} = \begin{pmatrix} a_{11}a_{12}\cdots a_{1m} \\ a_{21}a_{22}\cdots a_{2m} \\ \vdots \quad \vdots \quad \vdots \\ a_{m1}a_{m2}\cdots a_{mm} \end{pmatrix} \begin{pmatrix} \eta_1 \\ \eta_2 \\ \vdots \\ \eta_m \end{pmatrix} \tag{7.7}$$

where

$$a_{jk} = \xi(\underline{x}_k - \underline{c}_k) \qquad j,k=1,2,\ldots,m \tag{7.8}$$

which can be expressed more concisely as:

$$\underline{f} = A\underline{\eta} \tag{7.9}$$

Consequently, the solution of the RBF approach is obtained by solving the set of linear equations given by Eqn.(7.9). Clearly, the solution is given by $\underline{\eta} = A^{-1}\underline{f}$, where A^{-1} is the inverse matrix of A, and is directly calculable if the matrix A is square. However, if A is not square, the pseudo-inverse matrix can be formed as follows:

$$\underline{\eta} = \left(A^T A\right)^{-1} A^T \underline{f} \tag{7.10}$$

where $\left(A^T A\right)^{-1} A^T$ is the pseudo-inverse of A. Indeed, it has been proven that for all positive integers m, and for a large class of basis functions ξ, the matrix A is guaranteed to be non-singular if the data points are all distinct (Powell (1992)). This property is clearly very useful when employing linear optimization techniques to calculate the vector $\underline{\eta}$.

Gaussian functions are the most commonly used of the RBF family (Powell (1992)) and can be found extensively throughout the relevant literature. The general form of the Gaussian basis function is:

$$\xi(r)=\exp\left[\frac{-r^2}{2\sigma^2}\right] \tag{7.11}$$

where σ denotes the half-width of the basis function. Certain properties of this equation make it very appealing for use within an ANFIS type controller structure. Firstly, it is highly non-linear and provides good locality as a RBF indicating its suitability within a gain-scheduling type structure. Additionally, Gaussian basis functions are the only functions within the radial family that can be factorized (Poggio and Girosi (1990b), Jang et *al.* (1997)). Therefore a multi-dimensional Gaussian function can be represented as the products of lower dimensional Gaussian functions. Poggio and Girosi (1990b) argue that this composition property reflects similarities between this type of function and the neurons within the brain and is therefore probably the most realistic neuron function.

The multiquadric function as proposed by Hardy (1971) was originally employed for topographical mappings and surface re-constructions. The general form of a multiquadric function is:

$$\xi(r)=\sqrt{r^2+\gamma^2} \tag{7.12}$$

where γ is a positive constant. Although Hardy (1971) discussed the appropriateness of this function for modelling and surface approximation there is a lack of theoretical support and rigorous analysis concerning the function as compared to Gaussian functions.

The inverse multiquadric function was also proposed by Hardy (1971):

$$\xi(r) = \frac{1}{\sqrt{r^2 + \gamma^2}}. \tag{7.13}$$

The shape of the inverse multiquadric function has some similarities to the shape of the Gaussian function but again does not possess the same amount of rigorous theoretical support.

The thin plate spline function is written as:

$$\xi(r) = r^2 \log r \tag{7.14}$$

and was derived to produce a solution to minimize the amount of bending energy within a thin plate. An obvious advantage of this model is that the equation does not depend on any user specified constant as with the multiquadric equation.

The cubic equation has the general form:

$$\xi(r) = r^3 \tag{7.15}$$

and is one of the least commonly used basis functions for RBF modelling.

The linear equation, which is employed in typical ANFIS approximations, takes the general form:

$$\xi(r) = r. \tag{7.16}$$

Clearly a wide choice of basis functions exist for use within modelling architectures to approximate topographic surfaces and functions. Poggio and Girosi (1990b) formulate this approximation/interpolation problem within the framework of regularization theory as detailed below.

7.3.1 The Approximation Problem. Poggio and Gorosi (1990b) defined this problem as:

Given a set of data $S=\{(x_i, f_i)\}_{i=1}^{N}$ *which is obtained by sampling an unknown function F in the presence of noise, the approximation problem is to recover the function F (or an estimate of it) from the sampled data set S.*

This type of problem is referred to as 'ill-posed' since there exists an infinite number of possible solutions. In order to facilitate the approximation, some *a priori* assumptions are made about the unknown function. The function may be constrained to take a specific form $F(\underline{\eta},x)$ which is dependent on an unknown parameter vector $\underline{\eta}$. Subsequently, the problem is transformed into one of regression. However, the resulting solution depends highly on the appropriateness of this *a priori* assumption or assumed functional form. Consequently, it is more usual to make the assumption that the unknown function $F(\underline{\eta},\underline{x})$ is smooth such that similar inputs produce similar outputs.

The regularization approach to the 'ill-posed' approximation problem determines the approximating function f that minimizes a cost function of the form:

$$H[f]=\sum_{i=1}^{N}(f(x_i)-y_i)^2+\lambda\xi[f] \tag{7.17}$$

where λ is a positive constant generally known as the regularization parameter. The cost function is clearly composed of two terms. The first term on the right hand side of Eqn.(7.17) minimizes the difference between the actual function being approximated and the approximation itself. The second term is included to encourage smoothness within the approximating function f, while the regularization parameter λ controls the weighting between the two components. Function smoothness is incorporated by defining a smoothness functional $\xi[f]$ in such a way that the lower values of the functional correspond to smoother functions. The regularization parameter λ can also be viewed as an indicator of the sufficiency of the given data set as examples that specify the solution $f(x)$. In particular, the limiting value $\lambda \rightarrow 0$, implies that the problem is unconstrained, with the solution $f(x)$ being completely determined by the examples. The other limiting value $\lambda \rightarrow \infty$ implies that the a' priori smoothness constraint is by itself sufficient to specify the solution function $f(x)$, which also means that the data is irrelevant to the solution chosen by the smoothness constraint. In practice, the regularization parameter λ is assigned a value between these two extremes, so that both the sample data and the smoothness constraint contribute to the solution $f(x)$. Commonly the regularization parameter λ is chosen according to cross-validation techniques.

It has been shown in Poggio and Girosi (1990b) for a wide class of functional forms ξ that the minimization of Eqn.(7.17) (and thus the solution of the regularization problem) yields a solution of the form:

$$f(x) = \frac{1}{\lambda}\sum_{i=1}^{N}[y_i - f(x_i)]\xi(\underline{x}, \underline{x}_i) \tag{7.18}$$

where $\xi(x, x_i)$ is a basis function centred at point x_i. Eqn.(7.18) states that the solution to the regularization problem is a weighted sum of N basis functions centred at the sampled data points. Let $\eta_i = [y_i - f(x_i)]/\lambda$, then the solution to the regularization problem

lies in an N-dimensional subspace of the space of smooth functions, and the set of basis functions $\{\xi(x, x_i)\}$ centred at the data points constitutes a basis for this space. If the basis function ξ is chosen from a set of rotationally and translationally invariant functions, then ξ becomes a radial symmetric function denoted by $\xi=\xi(\|x-x_i\|)$. Using η_i instead of $[y_i-f(X_i)]/\lambda$, the regularization solution is given in the form:

$$f(\underline{x}) = \sum_{i=1}^{N} \eta_i \xi(\|\underline{x} - \underline{x}_i\|). \tag{7.19}$$

Note that Eqn.(7.19) is almost identical to Eqn.(7.3), except that the basis functions are centred at data points x_i. Thus the method of RBFs with basis function centred at data points can be derived from regularization theory. Alternatively, regularization theory provides a firm theoretical background to the method of RBFs.

An approximating solution to the regularization solution can be formed by considering the basis functions to have moving centres. The regularization approach specifies the requirement for each training data example pair to be represented by a specific basis function. Obviously, this becomes computationally inefficient when the number of samples is large. Instead of using one function centre for each data point, a smaller number of function centres $\{\underline{\eta}_i$, $i=1,2,...,m$, $m \leq N\}$ could be used to replace $\underline{x}_i$ in Eqn.(7.19). It has been shown (Poggio and Girosi (1990b)) theoretically that this approach constructs an approximation to the regularization solution. However, the number and location of the centres will strongly effect the outcome of the approximation. This leads to an approximating function of the form:

$$f(\underline{x}) = \sum_{\alpha=1}^{m} \eta_\alpha \xi(\underline{x}; \underline{x}_\alpha) \tag{7.20}$$

where α is the number of linearly independent functions (which are now fewer than the number of training data pairs), and the parameters x_α are called the centres of the basis functions denoted by ξ.

Each radial function is combined with a multiplicative weight η_α which determines the algebraic sign and gradient of the basis functions' slope, as shown in Figure 7.2 for a two input (dimensional) problem. Thus when the complete set of coefficients are substituted into the consequent functions and these functions are summed together, the overall output surface should fit the discrete data points exactly, providing logical interpolation for intermediary points. Such a surface is depicted in Figure 7.3.

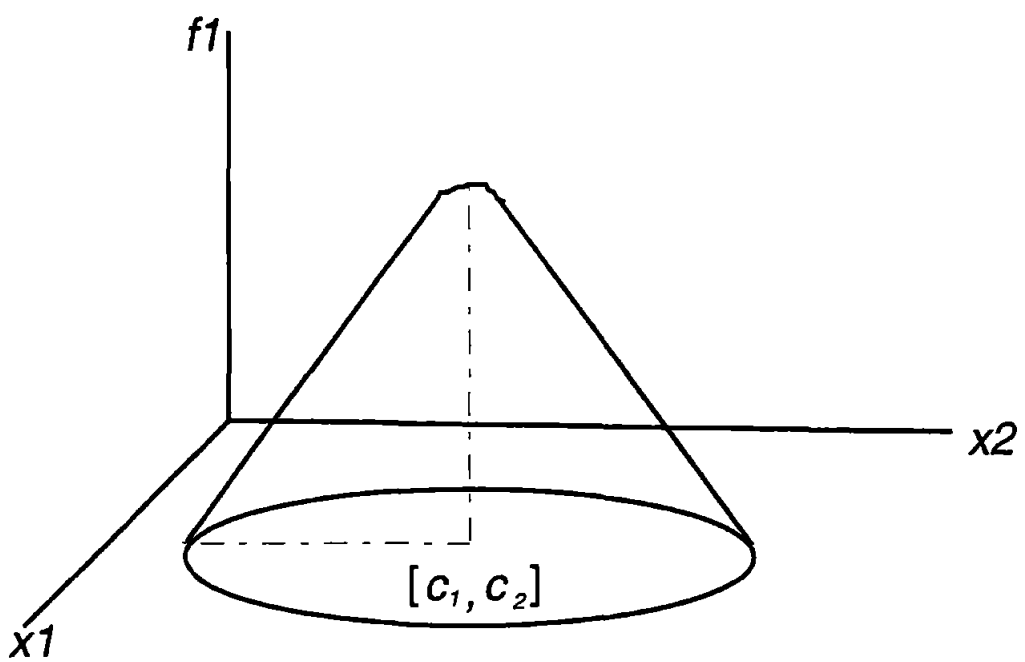


Figure 7.2: A two dimensional representation of a Gaussian radial basis function.

Additionally, the shape of the overall surface at any particular point is a function of the summation of all the individual surfaces at significant points. This means that sub-surfaces (as depicted above in Figure 7.2) with particularly sharp gradients, which are not in close proximity to the point of interest can still have a direct influence on regions containing slower sloped sub-surfaces. This behaviour is an important feature of this method because it suggests the ability of the resulting approximation to provide smooth transition over the output space.

7.3.2 A Review of Radial Basis Function Approaches to Control System Design

The current literature provides examples of RBF approaches to control system designs; of the existing literature, few are dedicated to fusing the approximation power of RBF networks with fuzzy logic. Indeed those few do not address the problem of rulebase interpretability and all employ neural network architectures consisting of three layers. As one of the main strengths of fuzzy logic is the availability of a clear rulebase for verification, the present work is driven towards producing a linguistically interpretable tuning structure fusing fuzzy logic with the locality properties of Gaussian basis functions.

McDowell et *al.* (1997) implemented a RBF network as a multi-input multi-output (MIMO) bank-to-turn autopilot for a surface-to-air missile. This application was based upon Gaussian basis functions, which were trained to adapt and compensate for roll induced cross-coupling. An excellent evaluation of the resulting neural autopilot was provided in 3-dimensions (6 degrees of freedom) using a command to line of sight algorithm. Results obtained are employed in conjunction with a gain-scheduling autopilot design, in the presence of time varying aerodynamic derivatives and control surface saturation constraints. The performance of the overall autopilot system is commended, however no verification tests appear to have been performed on the neural structure. Due to the lack of linguistic transparency of the resulting neural autopilot it is difficult to employ an expert to verify the resulting control rules.

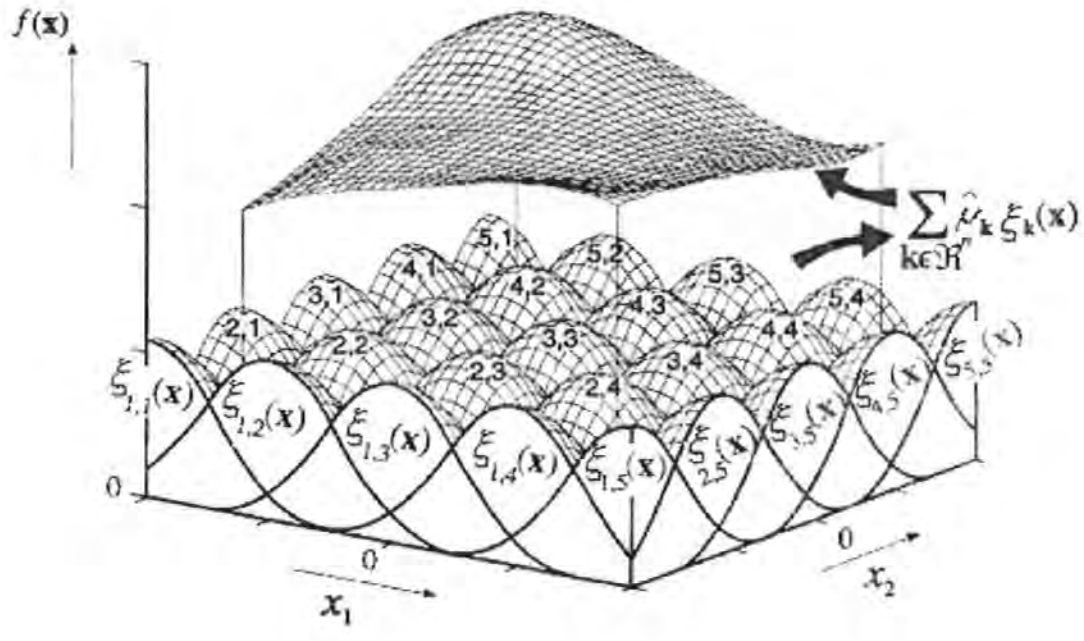


Figure 7.3: Gaussian basis function modelling of a smooth surface (after Heiss and Kampl (1996)).

Horikawa et *al.* (1992) highlight this requirement, stating that five layered network architectures are preferable in order that the transparency of the control rules be preserved during learning. However, their consequent functions do not possess the attractive properties of the multi-dimensional radially symmetric functions used by McDowell et *al.* (1997).

Choi and Hwang (1997) present the application of a MIMO fuzzy RBF network for control of an autonomous submersible. The results of the fuzzy RBF network autopilot simulations appear promising, yet the network again only employs a three layered architecture and is thus difficult to verify, or document as a fuzzy rulebase.

Due to the limited availability of formal design methods and robustness verification techniques concerning fuzzy controllers, it is considered important by the present

author to have access to the tuned rulebase as a means of control rule verification and analysis. Three layered representations of fuzzy RBF networks do not possess this transparency, and the applications within the literature typically ignore the important verification of the resulting structure.

7.4 Non-Linear Consequent Functions of *n*-dimensional Form

By employing composite Gaussian functions in place of the linear equations of the consequent layer of an ANFIS architecture, each fuzzy rule output becomes a non-linear function of the network inputs (Eqn.(7.20)). Rule outputs are then dependent on the width of the Gaussian basis function, its centre position in n-dimensional space and the multiplicative weight η_α. The overall effect of the architecture becomes that of a non-linear gain-scheduling controller, which is dependent on locally receptive fields. The i_{th} fuzzy rule is thus of the form (for 2 inputs and 1 functional output):

$$If \quad x_1 \quad is \quad A_i \quad and \quad x_2 \quad is \quad B_j \quad then \quad f_k = \eta_k \exp\left(\frac{-\left\|\underline{x}_k - \underline{c}_k\right\|_2^2}{2\sigma_k^2}\right) \qquad (7.21)$$

for:

$i = 1,2,...,m$ (the number of input membership functions on x_1),

$j = 1,2,...,n$ (the number of input membership functions on x_2), and

$k = 1,2,...,mn$.

Re-writing Eqn.(7.21) as the sum of composite Gaussian basis functions, yields the following system of n simultaneous linear equations for the unknown coefficients η_k :

$$\left.\begin{aligned} f_1 &= \eta_1 \exp\left[\frac{-\left\|\underline{x}_1 - \underline{c}_1\right\|^2}{2(\sigma_1)^2}\right] + \ldots\ldots + \eta_n \exp\left[\frac{-\left\|\underline{x}_1 - \underline{c}_n\right\|^2}{2(\sigma_n)^2}\right], \\ f_2 &= \eta_1 \exp\left[\frac{-\left\|\underline{x}_2 - \underline{c}_1\right\|^2}{2(\sigma_1)^2}\right] + \ldots\ldots + \eta_n \exp\left[\frac{-\left\|\underline{x}_2 - \underline{c}_n\right\|^2}{2(\sigma_n)^2}\right], \\ &\vdots \\ f_n &= \eta_1 \exp\left[\frac{-\left\|\underline{x}_n - \underline{c}_1\right\|^2}{2(\sigma_1)^2}\right] + \ldots\ldots + \eta_n \exp\left[\frac{-\left\|\underline{x}_n - \underline{c}_n\right\|^2}{2(\sigma_n)^2}\right] \end{aligned}\right\} \quad (7.22)$$

which when written in matrix notation yields once more

$$\underline{f} = A\underline{\eta} \quad (7.23)$$

where A is a matrix of Gaussian basis functions. Typically there will be fewer basis functions than available training samples, and consequently an initial estimate for the centre positions of each Gaussian is required. In this instance, the matrix A is not square and a sequential least squares estimate to the parameter vector $\underline{\eta}$ is often sought.

7.4.1 The Flexibility of Gaussian Radial Basis Functions

Gaussian functions are endowed with very attractive properties with respect to this application:

- Firstly, they are invariant under multiplication and differentiation. Therefore the calculation of gradient information concerning the hybrid learning rule for parameter adjustment requires only the pre-scaling of the Gaussian functions which have already been calculated.

- An n-dimensional Gaussian consequent membership function can be constructed by considering components from each network input within the $L2$-norm calculation. Such a construction requires only one evaluation of the exponential term of the Gaussian function due to the property that Gaussians are invariant under multiplication by other Gaussian functions.

- More generally, if the interpolation points $\{\underline{x}_k, k=1,2,\ldots,n\}$ are in general position and if the orders of the polynomials are at least quadratic, then it is often difficult to ensure that there are enough independent coefficients to satisfy the interpolation conditions. These problems become more severe when the dimension n is increased for piecewise polynomial approximations, but it is seen that the dimension has no effect on the convenience of the approximation suggested.

- Guaranteed non-singularity of the symmetric covariance matrix $A^T A$ under very mild or no restrictions for Gaussian RBFs has been proved. Consequently, provided that

$$rank\left(A^T A\right) > 1 \tag{7.24}$$

the determinant of $A^T A$ is defined and consequently the non-unique matrix inverse of $A^T A$ can be found. This provides a strong choice for the use of RBF approximations over many other types of approximating function. (NB: Even if the rank of $A^T A$ is equal to 1, the Moore-Penrose pseudo-inverse (Golub and Van Loan (1989)) could be formed).

7.4.2 The Modified Fuzzy Inference Mechanism

As a result of employing this new form of consequent function, the fuzzy inference procedure changes. Figures 7.4 and 7.5 illustrate respectively the more typical Mamdani and TSK styles of fuzzy inference. In comparison, Figure 7.6 illustrates the proposed composite Gaussian fuzzy inference mechanism. To elicit diagrammatic representation of the new output space, FISs considering two inputs and one output (with only two rules) are considered.

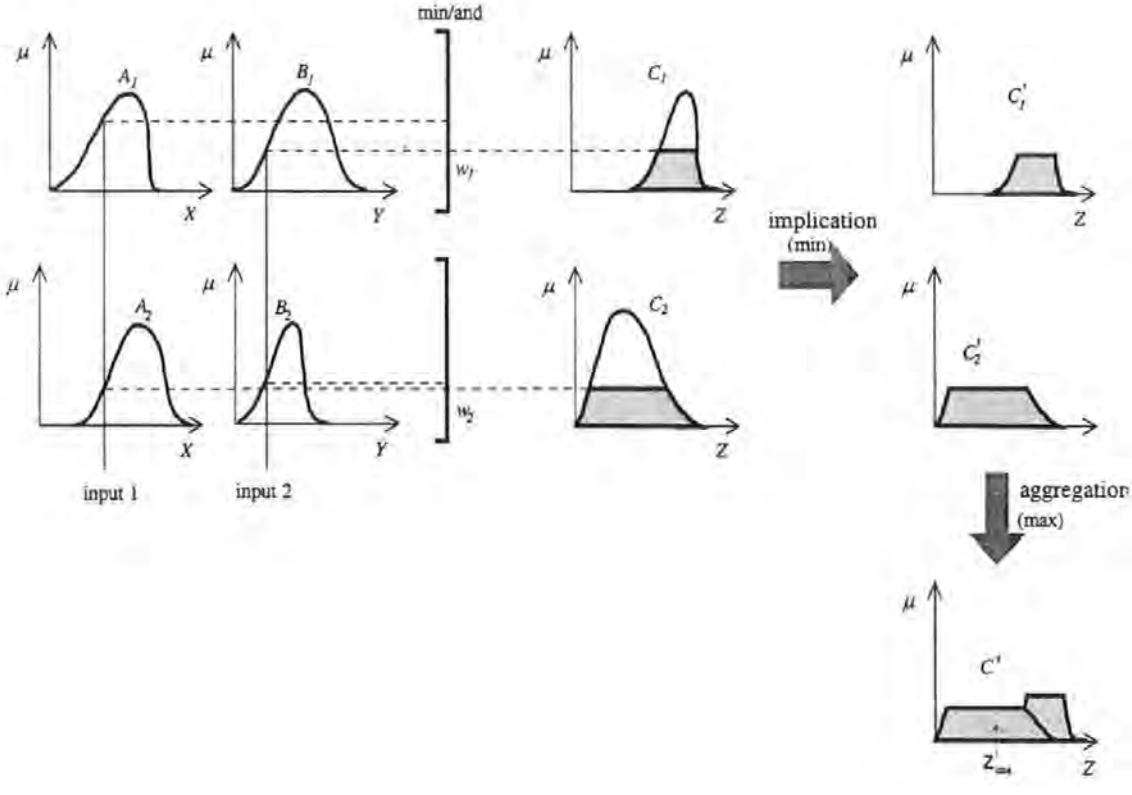


Figure 7.4: Mamdani fuzzy inference diagram.

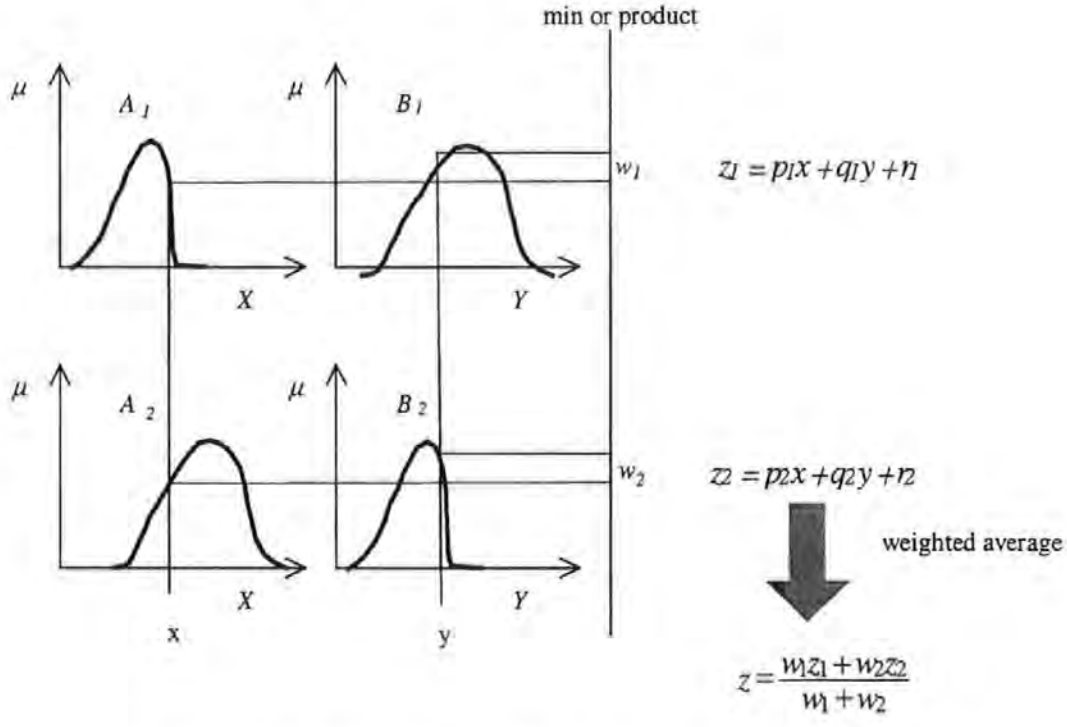


Figure 7.5: Takagi-Sugeno-Kang fuzzy inference diagram.

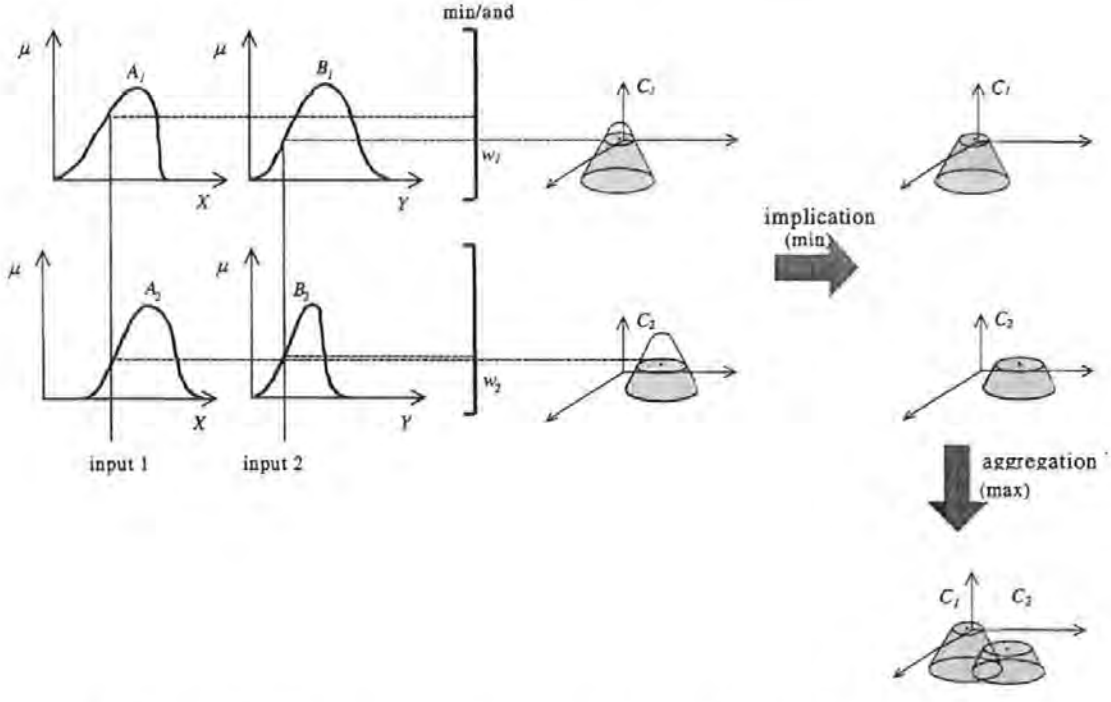


Figure 7.6: The proposed composite Gaussian fuzzy inference diagram.

7.5 Results and Discussion - Course Changing

Based upon the proposed method of composite Gaussian fuzzy inference a 9 rule fuzzy autopilot was developed (2 inputs and one functional output). This structure was chosen to elicit comparison with the previous MISO autopilot designs. The architecture of this autopilot is shown in the schematic of Figure 7.7 in which each function within layer 4 is a composite Gaussian of dimension 2 (as illustrated for the first consequent function). The output domain of this new autopilot was represented graphically in Figure 7.3 and it is thus apparent that the two-dimensional consequent functions are hill shaped in nature. By varying the parameter set of the proposed consequent functions, any smooth function can be approximated, assuming that the centres are positioned correctly (Heiss and Kampl (1996)).

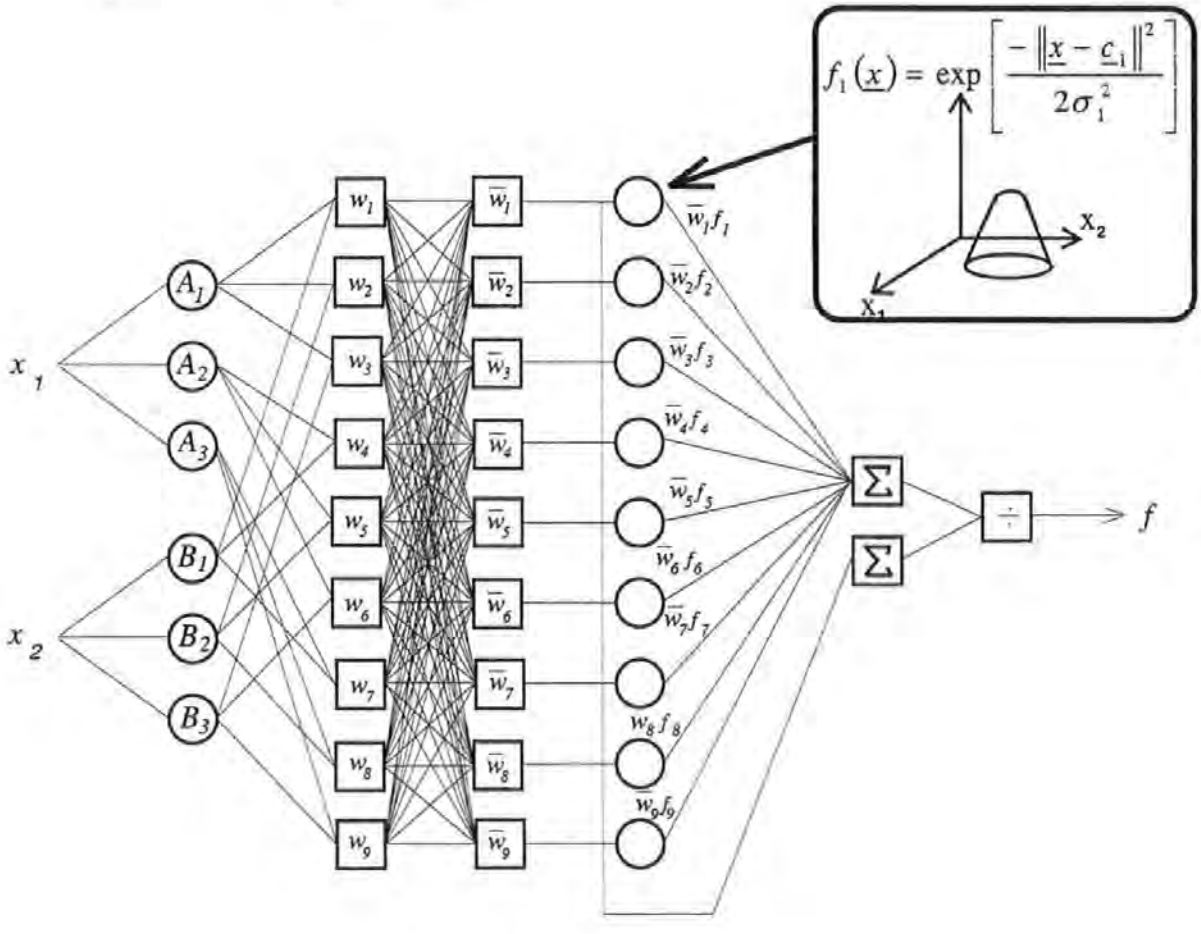


Figure 7.7: The proposed Gaussian FIS for two inputs and one functional output.

The response of the AUV when employing this autopilot for a course-changing manoeuvre of 40° (prior to the application of tuning algorithms) is depicted in Figure 7.8; the corresponding low canard rudder response is given in Figure 7.9.

This autopilot produced a sluggish course-changing response from the AUV in comparison to the results achieved by employing the hybrid rule tuned autopilot of Chapter 4, but compares similarly to the pre-tuned 9 rule TSK autopilot. In order to improve upon these results, the autopilot parameter set required some modification in the form of tuning. The ensuing sections discuss the results obtained after the application of particular tuning algorithms.

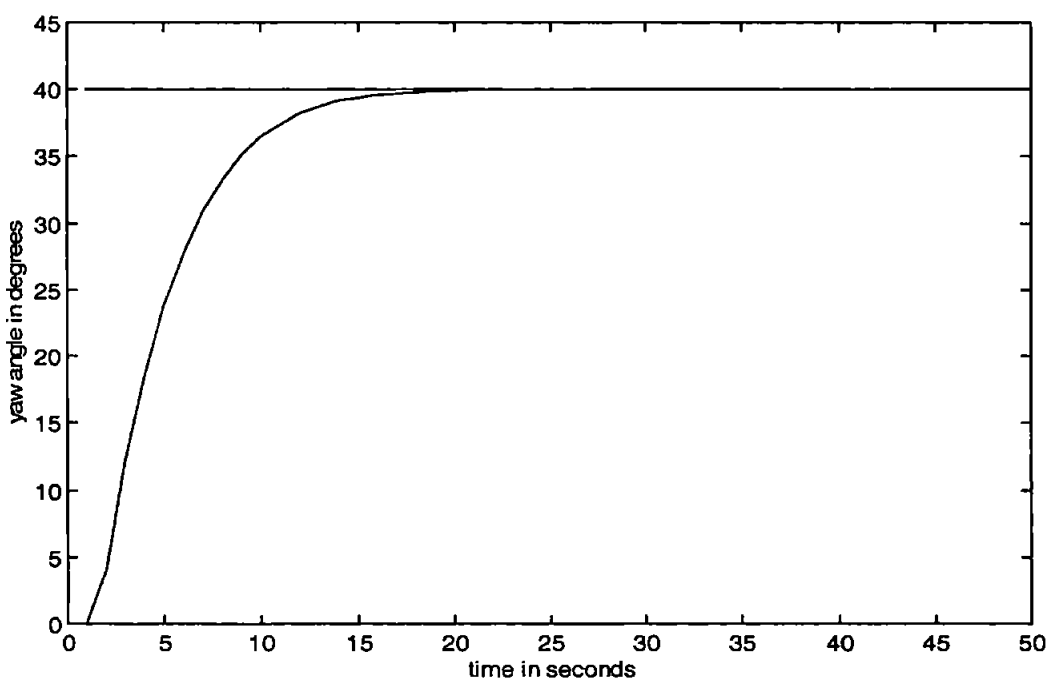


Figure 7.8: The yaw response of the AUV when employing the pre-tuned Gaussian autopilot for a 40° course-changing manoeuvre.

7.5.1 Applying the Hybrid Learning Rule

The linear dependence of the proposed consequent functions on their multiplicative weight coefficients (η_k's of Eqn.(7.6)), elicits direct application of the hybrid learning

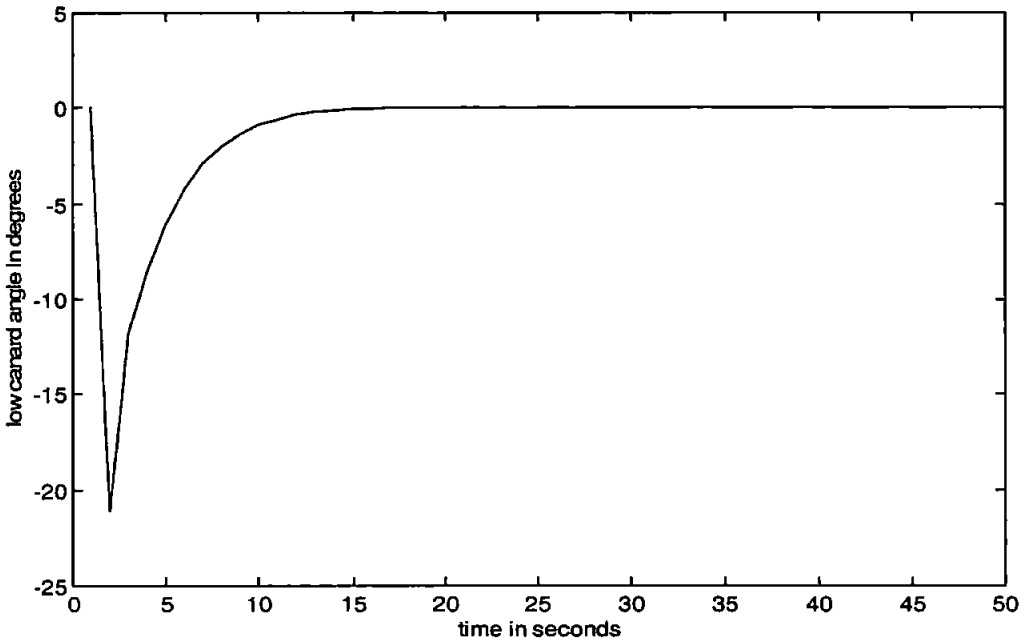


Figure 7.9: Low canard rudder response of the AUV when employing the Gaussian autopilot for a 40° course-changing manoeuvre.

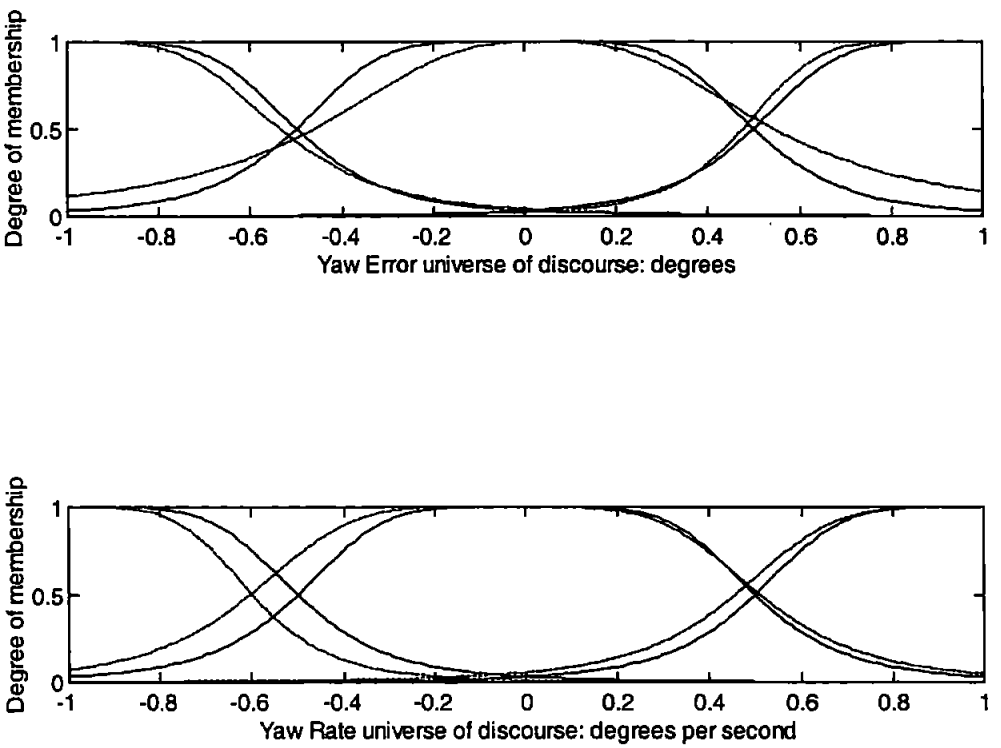


Figure 7.10: The input fuzzy sets before and after 300 epochs of tuning with the hybrid learning algorithm for the Gaussian FIS. (Dashed lined show tuned positions).

rule for backpropagation tuning of the premise parameters and least squares tuning of the consequent weight coefficients η_k. However, this assumes that the non-linear parameters ($\underline{c}_k$ and $\underline{\sigma}_k$) of the Gaussian consequent functions remain fixed.

Upon completion of 300.5 epochs of tuning, the fuzzy sets of the hybrid rule tuned Gaussian autopilot were taken as depicted in Figure 7.10. Clearly some adaptation of the autopilots input parameter set has taken place.

The rule base of the hybrid rule tuned Gaussian autopilot was taken as Eqn.(7.25), where the coefficients η_k have been tuned using the least-squares algorithm:

$$\text{If } \psi_\varepsilon \text{ is N and } \dot{\psi} \text{ is N then } \delta = 1.499 \cdot \exp\left[\frac{-(\psi_\varepsilon - 1)^2 - \left(\dot{\psi} - 1\right)^2}{2(0.4)^2}\right]$$

$$\text{If } \psi_\varepsilon \text{ is N and } \dot{\psi} \text{ is Z then } \delta = 0.368 \cdot \exp\left[\frac{-(\psi_\varepsilon - 0.75)^2 - \left(\dot{\psi} - 0.75\right)^2}{2(0.4)^2}\right]$$

$$\text{If } \psi_\varepsilon \text{ is N and } \dot{\psi} \text{ is P then } \delta = 2.091 \cdot \exp\left[\frac{-(\psi_\varepsilon - 0.5)^2 - \left(\dot{\psi} - 0.5\right)^2}{2(0.4)^2}\right]$$

$$\text{If } \psi_\varepsilon \text{ is Z and } \dot{\psi} \text{ is N then } \delta = 0.390 \cdot \exp\left[\frac{-(\psi_\varepsilon - 0.25)^2 - \left(\dot{\psi} - 0.25\right)^2}{2(0.4)^2}\right]$$

$$\text{If } \psi_\varepsilon \text{ is Z and } \dot{\psi} \text{ is Z then } \delta = 0.897 \cdot \exp\left[\frac{-(\psi_\varepsilon - 0)^2 - \left(\dot{\psi} - 0\right)^2}{2(0.4)^2}\right] \qquad (7.25)$$

$$\text{If } \psi_\varepsilon \text{ is Z and } \dot{\psi} \text{ is P then } \delta = 1.350 \,.\exp\left[\frac{-(\psi_\varepsilon + 0.25)^2 - \left(\dot{\psi} + 0.25\right)^2}{2(0.4)^2}\right]$$

$$\text{If } \psi_\varepsilon \text{ is P and } \dot{\psi} \text{ is N then } \delta = 1.012 \,.\exp\left[\frac{-(\psi_\varepsilon + 0.5)^2 - \left(\dot{\psi} + 0.5\right)^2}{2(0.4)^2}\right]$$

$$\text{If } \psi_\varepsilon \text{ is P and } \dot{\psi} \text{ is Z then } \delta = 0.576 \,.\exp\left[\frac{-(\psi_\varepsilon + 0.75)^2 - \left(\dot{\psi} + 0.75\right)^2}{2(0.4)^2}\right]$$

$$\text{If } \psi_\varepsilon \text{ is P and } \dot{\psi} \text{ is P then } \delta = -0.601 \,.\exp\left[\frac{-(\psi_\varepsilon + 1)^2 - \left(\dot{\psi} + 1\right)^2}{2(0.4)^2}\right]$$

The course-changing response of the AUV, when employing this autopilot in the forward path is depicted in Figure 7.11; the corresponding low canard rudder response is given in Figure 7.12. Application of the hybrid learning rule for selective tuning of the Gaussian autopilots parameter set has improved the course-changing responsiveness of the AUV by 12.3%, whilst the canard control effort has increased by 7.8%. However, the maximum canard angle incurred resides well below the saturation level of 25.2°. An evaluation of the current autopilots performance improvement with respect to the pre-tuned Gaussian autopilot is provided in Table 7.1.

The initial positions of the non-linear consequent function centres ($\underline{c_k}$) remained fixed throughout the tuning process; the widths of the consequents ($\underline{\sigma_k}$) remain fixed also. As detailed in section 7.3.1, the rule base considers a depleted number of basis functions compared to typical RBF control structures, and therefore the correct positioning of the function centres is of paramount importance in achieving suitable approximations to the

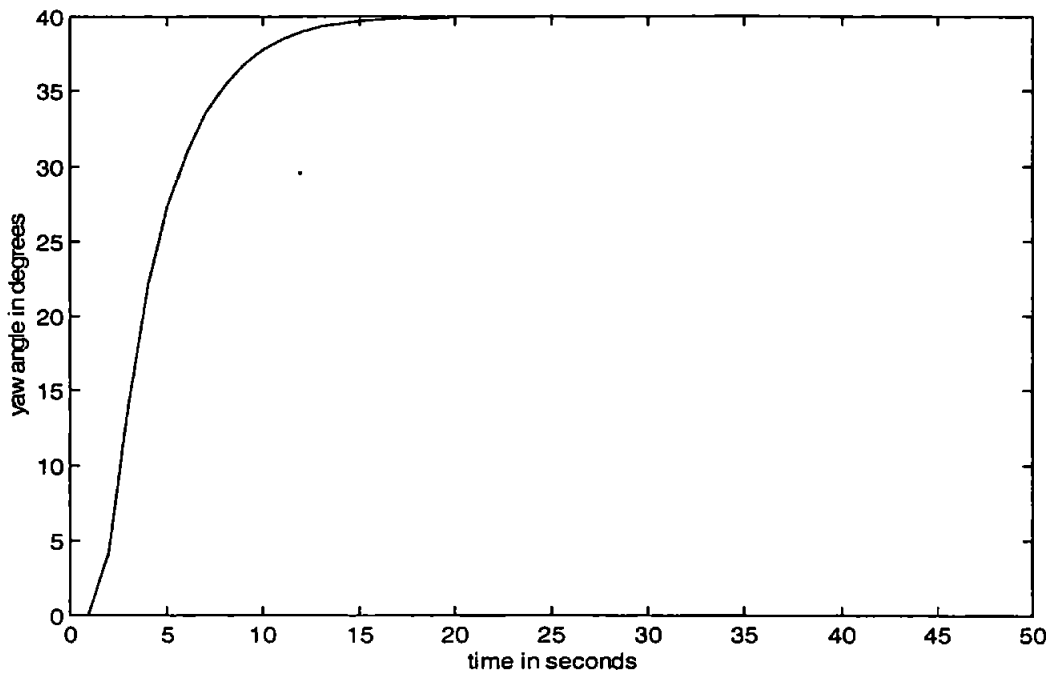


Figure 7.11: The yaw response of the AUV when employing the hybrid tuned Gaussian autopilot for a 40° course-changing manoeuvre.

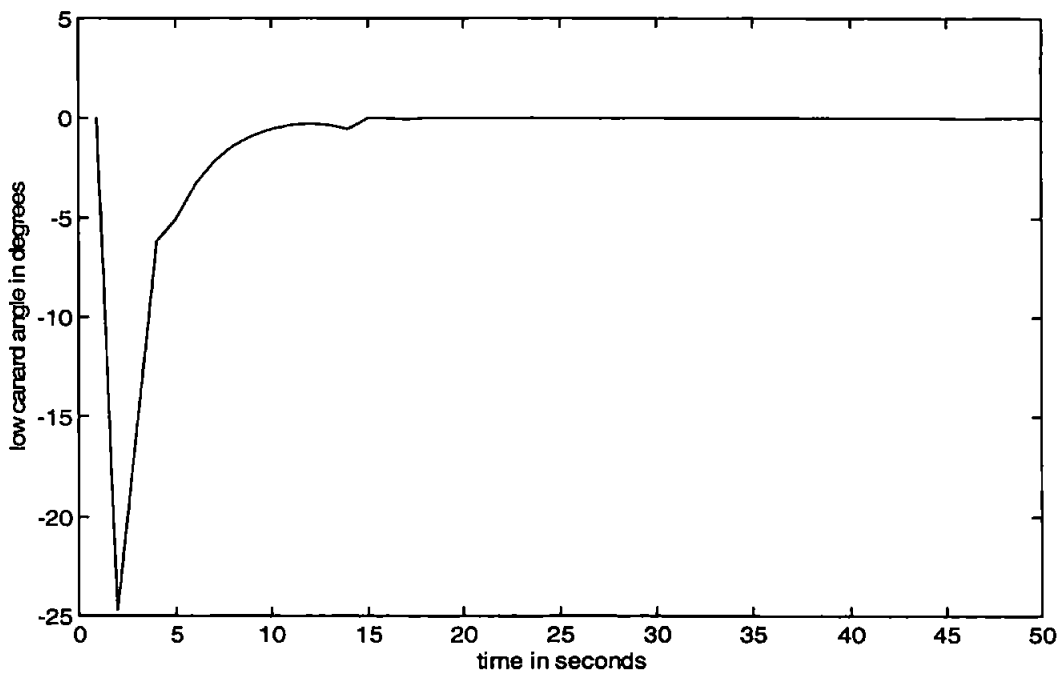


Figure 7.12: Low canard rudder response of the AUV when employing the hybrid tuned Gaussian autopilot for a 40° course-changing manoeuvre.

under-lying models behaviour. Therefore to further improve the course-changing effectiveness of this autopilot it is important that the centres of the consequents be positioned correctly.

AUV model	Pre-tuned Gaussian autopilot					Hybrid tuned Gaussian autopilot				
	ψ_ε $(^\circ)^2$	δ_ε $(^\circ)^2$	T_R sec	$M_p(t)$ %	sse %	ψ_ε $(^\circ)^2$	δ_ε $(^\circ)^2$	T_R sec	$M_p(t)$ %	sse %
7.5 knots	3908.8	847.1	15.86	0.02	0	3428.0	913.2	14.31	0	0

Table 7.1: Performance comparisons between hybrid rule tuned and pre-tuned Gaussian autopilots for a course-change of 40° at 7.5-knots.

Appendix J details the parameters of the hybrid rule tuned Gaussian autopilot. Although the proposed Gaussian FIS approach shows some promise as an autopilot structure, the computational cost of implementation is greater than a typical ANFIS type architecture. The following section discusses a more computationally efficient method for implementing the composite Gaussian functions to alleviate the computational burden involved in calculating the model output.

7.5.2 Towards Computationally Efficient Gaussian Implementation

The use of Gaussian consequent functions within the network architecture is shown to be suitable for modelling highly non-linear functions in the form of autopilot solutions for AUV control. However, this requires the multiplication of each basis function with its corresponding coefficient η_i.

Heiss and Kampl (1996) discuss the implementation of a multiplication-free RBF network, whereby Gaussian basis functions are employed in a computationally efficient additive form. Because matrix additions require fewer floating point operations (flops)

as opposed to matrix multiplications, the Gaussians within a RBF type neural network can be encoded as a series of additions rather than multiplications.

This methodology is adopted herein for the consequent layer of the proposed Gaussian FIS such that each composite Gaussian function is encoded as a more computationally efficient set of additions, equivalent to that of its original multiplicative form.

In a first step to increase computational efficiency, the input space ($\mathfrak{R}^n$) is scaled such that the width of each basis function is pre-defined as $1/\sqrt{2\ln(2)}$, which from Eqn.(7.3) yields:

$$\exp\left[\frac{-\|\underline{x}-\underline{c}\|^2}{2(\sigma)^2}\right]_{\sigma=\frac{1}{\sqrt{2\ln(2)}}}$$

$$= \exp\left[-\|\underline{x}-\underline{c}\|^2.\ln(2)\right]$$

$$= 2^{-\|\underline{x}-\underline{c}\|^2} \tag{7.26}$$

Indeed, Hu (1997) clearly states that the width of Gaussian basis functions contributes to the fine tuning of such an approach, the modelling capability being more dependent upon the partitioning of the input-output product space. Eqn.(7.21) can be re-written as:

$$f(\underline{x}) = \sum_{i=1}^{N} \eta_i . 2^{-\|\underline{x}-\underline{c}\|^2} \tag{7.27}$$

which can be further simplified as follows:

$$\eta_i .2^{-\|\underline{x}-\underline{c}\|^2} = 2^{\log_2(\eta_i)} .2^{-\|\underline{x}-\underline{c}\|^2} = 2^{\log_2(\eta_i)-\|\underline{x}-\underline{c}\|^2} \tag{7.28}$$

where $log_2(\bullet)$ is the logarithm to the base 2. Thus Eqn.(7.27) can be expressed as:

$$f(\underline{x}) = \sum_{i=1}^{N} 2^{\log_2(\eta_i)-\|\underline{x}-\underline{c}\|^2} \tag{7.29}$$

Eqn.(7.29) has no practical importance except that no multiplication is required to evaluate the Gaussian element of the consequent functions of layer 4 when employing this model. However, this model formulation is somewhat restrictive in that only positive functions f and positive weights η_i can be considered. Also it would be futile to save n multiplications and replace them by n logarithms, especially as the model is non-linear in the parameters of $\log_2(\eta)$. To alleviate these drawbacks, consider a model of the form suggested by Heiss and Kampl (1996):

$$f(x) = \log_e\left[2^{\hat{a}_i-\|\underline{x}-\underline{c}\|^2}\right] \tag{7.30}$$

where no multiplication is necessary to evaluate Eqn.(7.30) and the logarithm need only be computed once for each model evaluation. The logarithm of Eqn.(7.30) is included to compensate for the non-linearity of the underlying model, and can be an approximation to the logarithm to the base e. Both the function and its parameters are under no restrictions, the model being linear with respect to these parameters.

7.5.2.1 Implementing the Theory within an Autopilot

To examine the improved efficiency of such an approach, comparisons were made with the autopilot design of section 7.5.1. As the fuzzy controllers are all implemented as MATLAB Executable (MEX) files, to elicit direct comparison it was essential that the

structure and inference of each autopilot was identical. The consequent functions within each autopilot were written as MATLAB M-Files and called externally from the MEX routine. Thus the number of floating point operations could be compared directly.

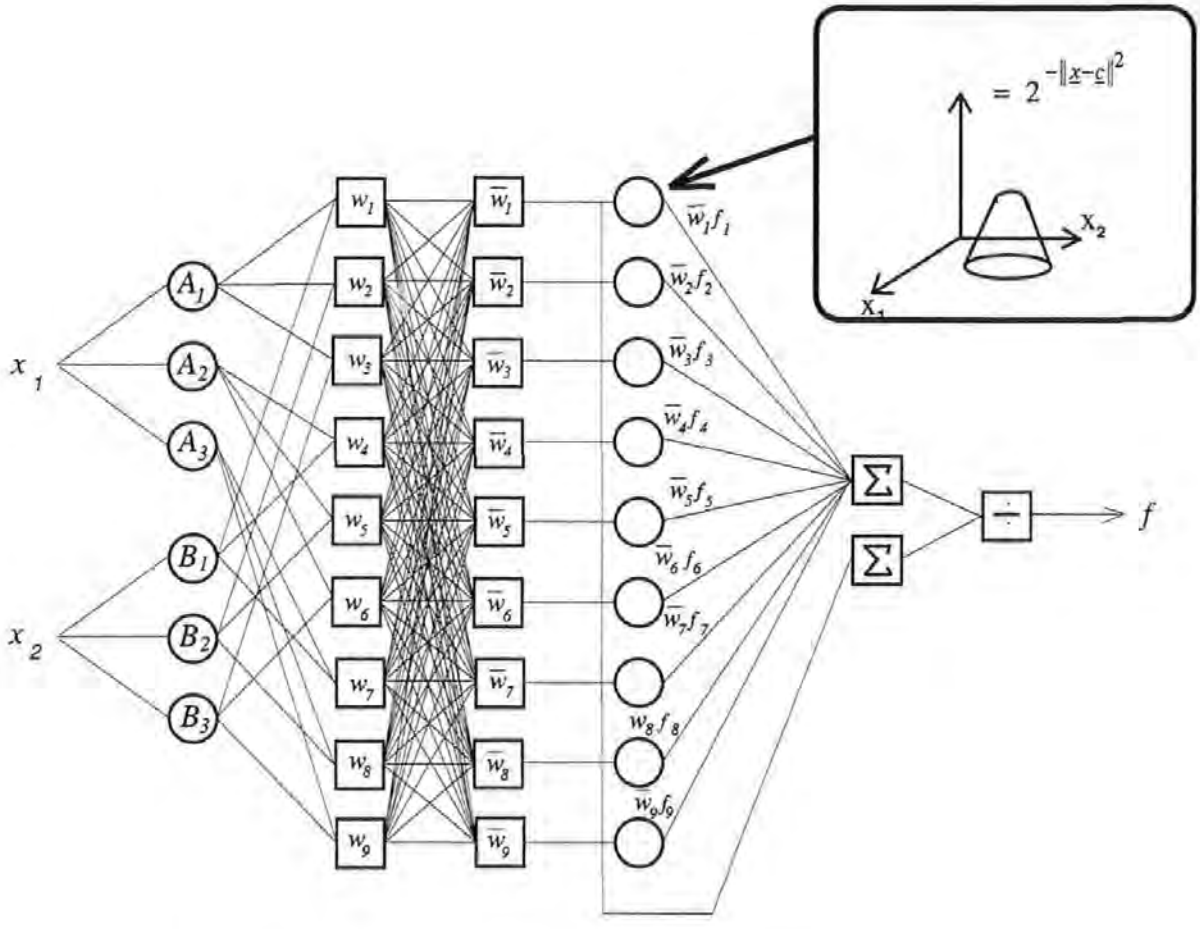


Figure 7.13: The computationally efficient Gaussian autopilot structure.

7.5.2.2 Results and Discussion

The hybrid rule tuned Gaussian MISO autopilot was simulated at 7.5 knots whilst employing the proposed computationally efficient Gaussian functions of Eqn.(7.26), as shown in Figure 7.13.

As the responses obtained from employing each autopilot are identical, they are not reproduced here. Conversely, the number of flops involved in simulating the AUV over a 40° course change at 7.5-knots for 50 seconds are given in Table 7.2

AUV model	Original Gaussian Model	New Gaussian Model
7.5-knots	758,559 flops	754,851 flops

Table 7.2: Comparative assessment of the number of floating point operations arising from each autopilot strategy.

The implementation of the computationally efficient Gaussian consequent functions diminishes the number of floating point operations used by 3708 (0.49%) in this instance, when employing the MISO Gaussian autopilot. Indeed, extended simulations did not sufficiently improve this poor computational saving. Clearly, this approach was almost fruitless here and was not considered further.

The following section documents a novel algorithm, based upon an extension of the hybrid learning rule, to tune the Gaussian FIS non-linear consequent parameters.

7.5.3 A Natural Extension to the Hybrid Learning Rule

To elicit full adaptation of the consequent functions it is necessary to employ a non-linear tuning algorithm. The backpropagation training element of the hybrid learning rule is clearly a convenient way to facilitate this. By extending the hybrid rule to adapt the non-linear parameters of the Gaussian consequent functions in the backward pass of each epoch, the whole non-linear parameter set can be locally tuned.

Applying this approach to the pre-tuned Gaussian autopilot, produced the premise fuzzy sets of Figure 7.14. The amount by which these sets have adapted to approximate the underlying model's behaviour is clearly less than in the case of the autopilot tuned

using the original hybrid learning algorithm. This may be due to the fitting of the non-linear consequent functions parameters to the underlying model, which would alleviate the requirement for the non-linear parameter set of the input space to vary as significantly.

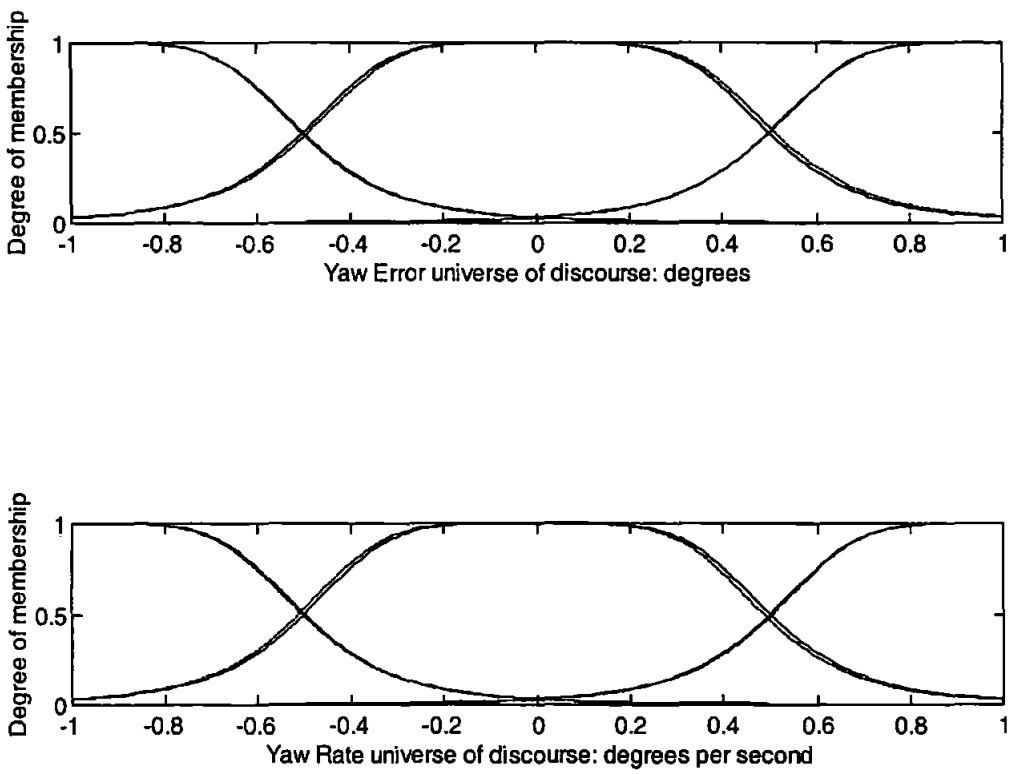


Figure 7.14: The input fuzzy sets before and after tuning with the extended hybrid learning algorithm. (Dashed lines show the tuned positions).

The rule base of the Gaussian autopilot after completion of 300.5 epochs of tuning via the extended hybrid learning rule was taken as follows (Eqn.(7.31)):

$$\text{If } \psi_\varepsilon \text{ is N and } \dot{\psi} \text{ is N then } \delta = 4.894 \cdot \exp\left[\frac{-(\psi_\varepsilon - 0.987)^2 - \left(\dot{\psi} - 0.998\right)^2}{2(0.394)^2}\right]$$

$$\text{If } \psi_\varepsilon \text{ is N and } \dot{\psi} \text{ is Z then } \delta = 1.471 \,.\exp\left[\frac{-(\psi_\varepsilon - 0.735)^2 - (\dot{\psi} - 0.746)^2}{2(0.401)^2}\right]$$

$$\text{If } \psi_\varepsilon \text{ is N and } \dot{\psi} \text{ is P then } \delta = -0.788 \,.\exp\left[\frac{-(\psi_\varepsilon - 0.489)^2 - (\dot{\psi} - 0.504)^2}{2(0.395)^2}\right]$$

$$\text{If } \psi_\varepsilon \text{ is Z and } \dot{\psi} \text{ is N then } \delta = -0.263 \,.\exp\left[\frac{-(\psi_\varepsilon - 0.251)^2 - (\dot{\psi} - 0.241)^2}{2(0.401)^2}\right]$$

$$\text{If } \psi_\varepsilon \text{ is Z and } \dot{\psi} \text{ is Z then } \delta = 0.569 \,.\exp\left[\frac{-(\psi_\varepsilon + 0.004)^2 - (\dot{\psi} - 0.003)^2}{2(0.394)^2}\right] \quad (7.31)$$

$$\text{If } \psi_\varepsilon \text{ is Z and } \dot{\psi} \text{ is P then } \delta = 0.395 \,.\exp\left[\frac{-(\psi_\varepsilon + 0.261)^2 - (\dot{\psi} + 0.239)^2}{2(0.395)^2}\right]$$

$$\text{If } \psi_\varepsilon \text{ is P and } \dot{\psi} \text{ is N then } \delta = 0.397 \,.\exp\left[\frac{-(\psi_\varepsilon + 0.485)^2 - (\dot{\psi} + 0.496)^2}{2(0.397)^2}\right]$$

$$\text{If } \psi_\varepsilon \text{ is P and } \dot{\psi} \text{ is Z then } \delta = 0.399 \,.\exp\left[\frac{-(\psi_\varepsilon + 0.773)^2 - (\dot{\psi} + 0.770)^2}{2(0.399)^2}\right]$$

$$\text{If } \psi_\varepsilon \text{ is P and } \dot{\psi} \text{ is P then } \delta = 0.403 \,.\exp\left[\frac{-(\psi_\varepsilon + 0.994)^2 - (\dot{\psi} + 1.003)^2}{2(0.403)^2}\right]$$

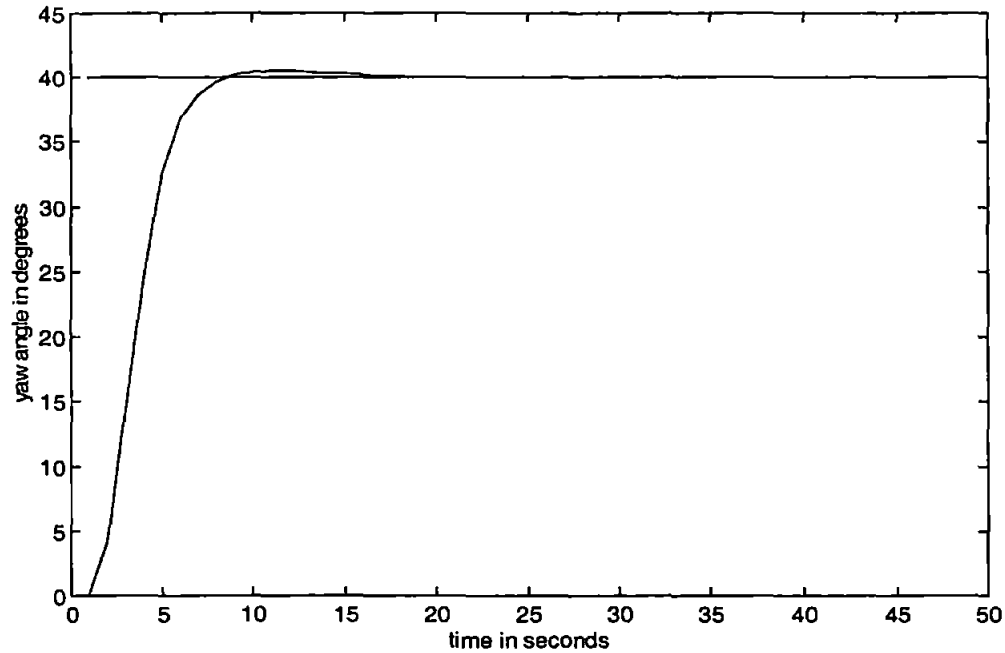


Figure 7.15: The yaw response of the AUV when employing the extended hybrid tuned Gaussian autopilot for a 40° course-changing manoeuvre.

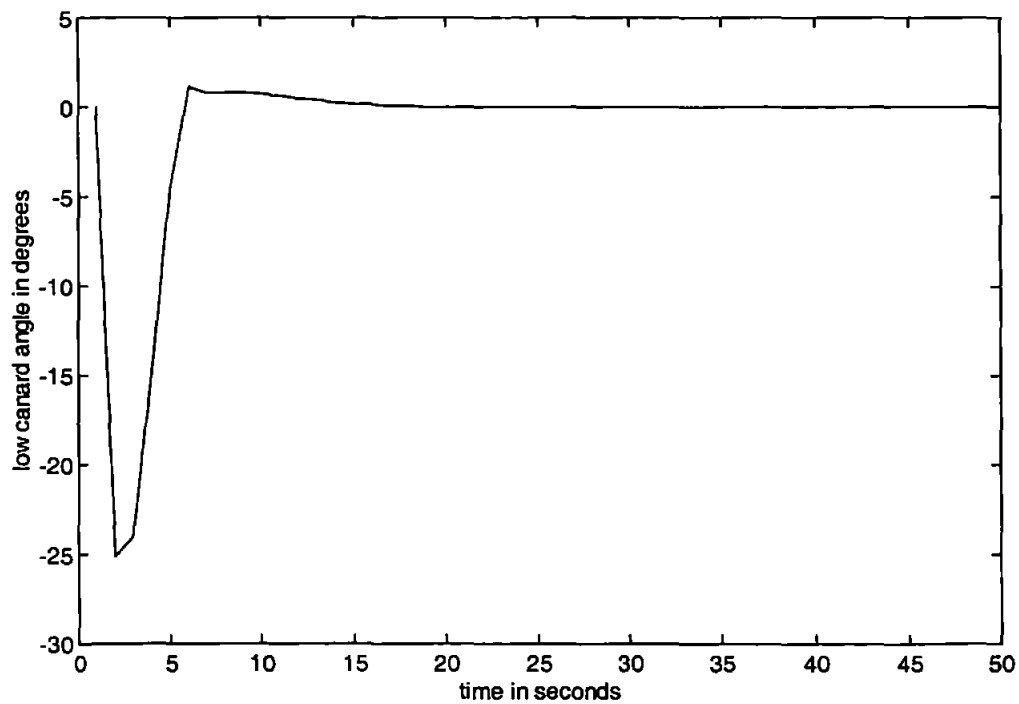


Figure 7.16: Low canard rudder response of the AUV when employing the extended hybrid tuned Gaussian autopilot for a 40° course-changing manoeuvre.

The application of this new algorithm produces a course-changing autopilot that incurs a fast accurate response when employed within the AUV model (Figure 7.15 and Figure 7.16). Indeed, this autopilot is comparable if not superior to the ANFIS style autopilot developed within Chapter 4. Table 7.3 illustrates the comparative performances of these two autopilot strategies at 7.5-knots.

The results of Appendix K illustrate that a Gaussian type FIS approach can provide superior modelling accuracy to the ANFIS technique, when required to approximate a highly non-linear function. Simulating the AUV at 5 and 10-knots as a form of robustness test yielded the responses of Figures 7.17 and 7.18. It is evident that the speed variations provide a suitable measure of the Gaussian autopilots course-changing generalization ability.

AUV model	Hybrid rule tuned ANFIS autopilot					Extended hybrid rule tuned Gaussian autopilot				
	ψ_ε $(^\circ)^2$	δ_ε $(^\circ)^2$	T_R sec	$M_p(t)$ %	sse %	ψ_ε $(^\circ)^2$	δ_ε $(^\circ)^2$	T_R sec	$M_p(t)$ %	sse %
7.5 knots	3057.4	1451.7	7.65	1.75	0	3033.2	1459.9	7.37	1.76	0

Table 7.3: Performance comparisons between hybrid rule tuned ANFIS autopilot and extended hybrid rule tuned Gaussian autopilot for a course-change of 40° at 7.5-knots.

The results for course-changing control of the AUV prove effective over the extremities of the speed envelope. Additionally, the resulting autopilot compared favourably with the ANFIS style autopilot of Chapter 4. Encouraged by the accuracy of these results, the technique was applied to the design of a multivariable course-changing and roll-minimizing autopilot. Results pertaining to this work are detailed in the following section.

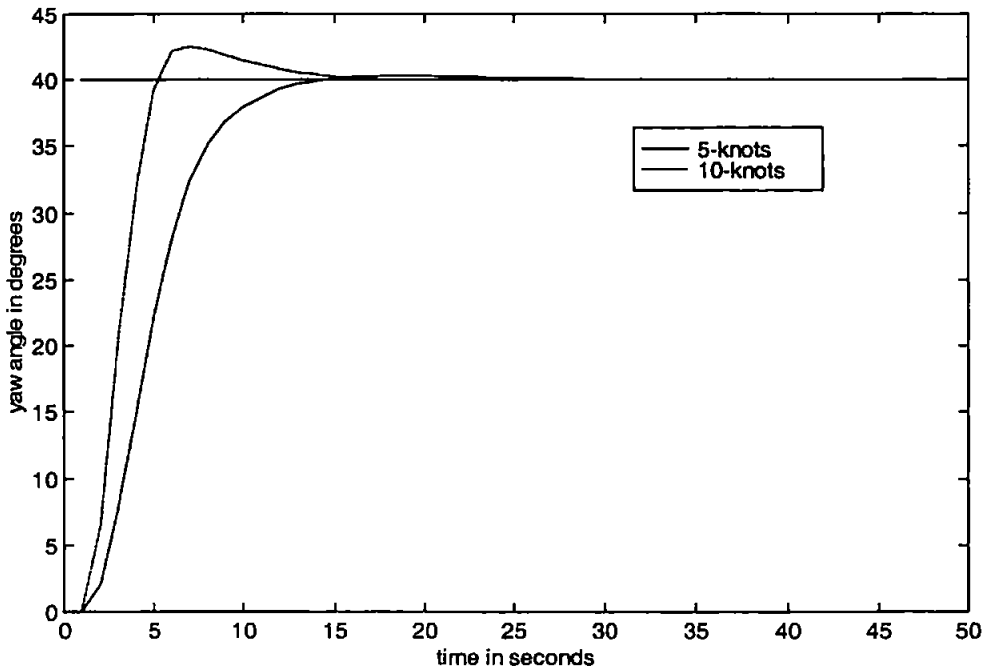


Figure 7.17: The yaw response of the AUV when employing the extended hybrid tuned Gaussian autopilot for a 40° course-changing manoeuvre – 5 and 10-knots.

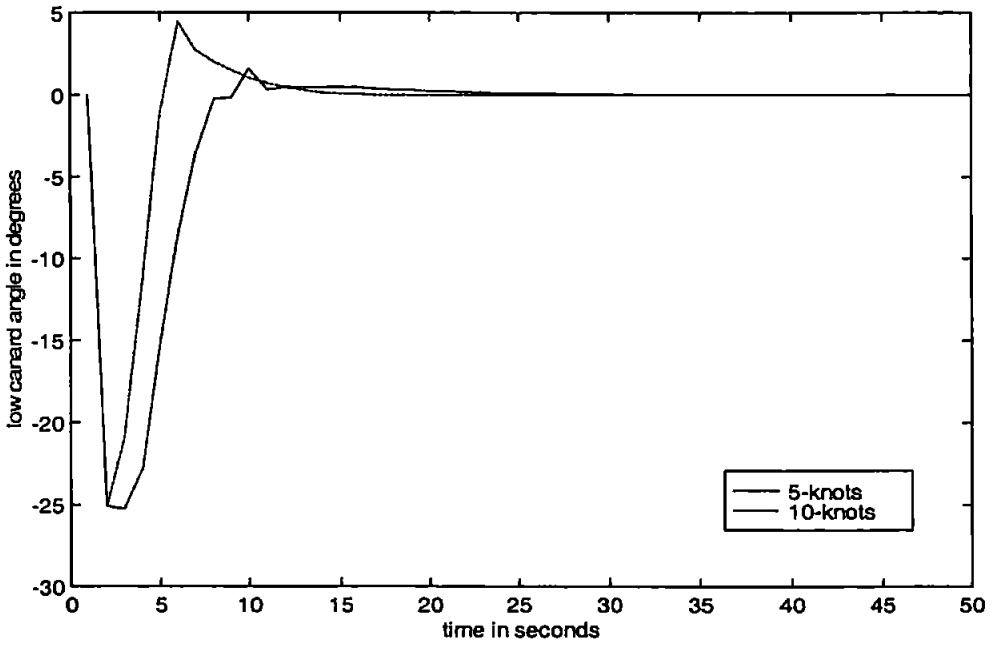


Figure 7.18: Low canard rudder response of the AUV when employing the extended hybrid tuned Gaussian autopilot for a 40° course-changing manoeuvre – 5 and 10-knots.

7.6 Results and Discussion - Multivariable Control

Obviously, the design approach of the previous sections is equally applicable to multivariable autopilot configurations. More specifically, if the function to be approximated F is multi-dimensional with respect to the output space, the interpolation conditions of Eqn.(7.5) can be extended to include a subscript i, yielding:

$$F_k(\underline{x}_i) = f_{ik}; \qquad i=1, 2, ..., n, \; k=1, 2, ..., m \tag{7.32}$$

and the interpolation functions of Eqn.(7.6) are now given as:

$$f_k(x) = \sum_{j=1}^{m} \eta_{jk} \xi\left(\left\|x - c_j\right\|\right) \qquad \underline{x}, \underline{c}_i \in \Re^n, \; k=1,2, ...,m \tag{7.33}$$

where the coefficients η_{jk} of Eqn.(7.33) are obtained using the inverse of matrix A defined in Eqn.(7.23).

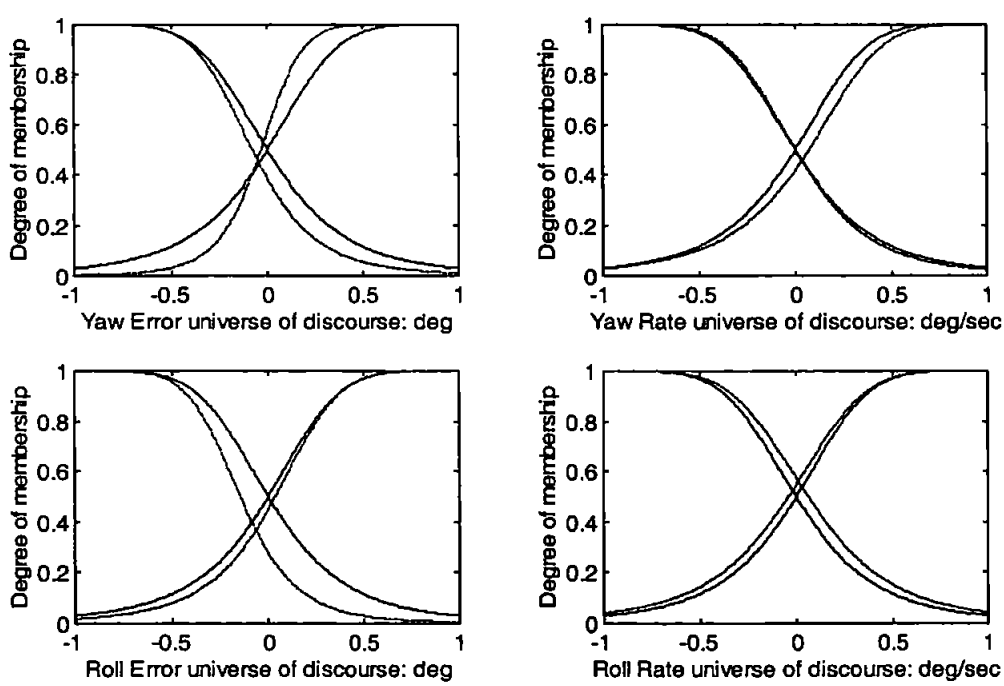


Figure 7.19: The original (—) and tuned (—. .) fuzzy sets for the 16 rule Gaussian inference multivariable autopilot.

7.6.1 Course Changing and Roll Minimizing Autopilot

Applying the theory of section 7.4 to the design of a multivariable yaw and roll autopilot implies Gaussian consequent functions of four-dimensional form within each output space. The following results were achieved by employing the extended tuning paradigm of section 7.5.2 to a 4 input-2 output Gaussian inference architecture. This architecture was chosen to elicit comparisons with the 16 rule co-active ANFIS (CANFIS) autopilot designed within Chapter 5.

Upon completion of 300.5 epochs of tuning the fuzzy sets were as depicted in Figure 7.19. The rule base of this 16 rule multivariable Gaussian autopilot was taken in abbreviated form as Eqn.(7.34):

$$\text{If } \psi_\varepsilon \text{ is N and } \dot{\psi} \text{ is N and } \phi_\varepsilon \text{ is N and } \dot{\phi} \text{ is N}$$

$$\text{then } \delta_\psi = 1.004.\exp\left[\frac{-(\psi_\varepsilon + 0.5904)^2 - \left(\dot{\psi} + 0.7906\right)^2 - (\phi_\varepsilon + 1.407)^2 - \left(\dot{\phi} + 2.106\right)^2}{2(0.3764)^2}\right]$$

$$\text{and } \delta_\phi = 0.972.\exp\left[\frac{-(\psi_\varepsilon + 1.917)^2 - \left(\dot{\psi} - 0.2177\right)^2 - (\phi_\varepsilon - 0.1113)^2 - \left(\dot{\phi} + 2.331\right)^2}{2(0.4065)^2}\right]$$

If ψ_ε is N and $\dot{\psi}$ is N and ϕ_ε is N and $\dot{\phi}$ is P

$$\text{then } \delta_\psi = 0.9937 .\exp\left[\frac{-(\psi_\varepsilon - 0.4207)^2 - \left(\dot{\psi} + 0.4184\right)^2 - (\phi_\varepsilon + 0.5837)^2 - \left(\dot{\phi} - 1.472\right)^2}{2(0.3965)^2}\right]$$

$$\text{and } \delta_\phi = 0.999 .\exp\left[\frac{-(\psi_\varepsilon + 0.0595)^2 - \left(\dot{\psi} + 0.4885\right)^2 - (\phi_\varepsilon + 0.3987)^2 - \left(\dot{\phi} - 0.7263\right)^2}{2(0.414)^2}\right]$$

. . . . . (7.34)

If ψ_ε is P and $\dot{\psi}$ is P and ϕ_ε is P and $\dot{\phi}$ is N

$$\text{then } \delta_\psi = 0.9392 .\exp\left[\frac{-(\psi_\varepsilon + 0.5992)^2 - \left(\dot{\psi} + 0.1073\right)^2 - (\phi_\varepsilon + 0.361)^2 - \left(\dot{\phi} - 0.171\right)^2}{2(0.4109)^2}\right]$$

$$\text{and } \delta_\phi = 1.016 .\exp\left[\frac{-(\psi_\varepsilon - 0.0029)^2 - \left(\dot{\psi} - 0.6548\right)^2 - (\phi_\varepsilon - 0.0354)^2 - \left(\dot{\phi} + 0.4872\right)^2}{2(0.3936)^2}\right]$$

If ψ_ε is P and $\dot{\psi}$ is P and ϕ_ε is P and $\dot{\phi}$ is P

$$\text{then } \delta_\psi = 1.353 .\exp\left[\frac{-(\psi_\varepsilon - 0.628)^2 - \left(\dot{\psi} - 0.9703\right)^2 - (\phi_\varepsilon - 1.305)^2 - \left(\dot{\phi} - 1.386\right)^2}{2(0.3806)^2}\right]$$

$$\text{and } \delta_\phi = 1.088 .\exp\left[\frac{-(\psi_\varepsilon - 2.058)^2 - \left(\dot{\psi} + 0.1135\right)^2 - (\phi_\varepsilon - 0.4532)^2 - \left(\dot{\phi} - 0.9786\right)^2}{2(0.3898)^2}\right]$$

Appendix L details the full rule base of the multivariable Gaussian autopilot. The yaw and roll responses of the AUV are depicted in Figure 7.20, when employing this autopilot over a 40° course-changing manoeuvre at 7.5-knots, during which 0° roll cross-coupling was demanded. Additionally, Figure 7.21 illustrates the associated low canard and stern hydroplane responses of the AUV.

Clearly, the proposed autopilot structure performs well yielding a maximum roll angle of 5° with respect to the 40° course-changing demand. Additionally, the control effort employed during this manoeuvre is reduced as compared to the results achieved when employing the CANFIS yaw and roll autopilot of Chapter 5. To illustrate this, Table 7.4 includes the results of each strategy at 7.5 knots. The response of the AUV when employing the Gaussian autopilot at both 5 and 10-knots is depicted in Figures 7.22 and 7.23.

Quantitatively, the multivariable Gaussian autopilot quite clearly yields a superior performance to that of the CANFIS autopilot. The yaw error induced during the course change is almost identical in terms of integral of time square error, but the roll error incurred is reduced by 37.28%. The canard rudder experiences a more sustained period at its maximum induced angle than under the CANFIS autopilot, yet induces a lower overall angle. Thus the yaw induced roll response is less pronounced.

	Performance Indices – Yaw and Roll Autopilots							
Autopilot strategy	ψ_ε $(^\circ)^2$	δ_ε $(^\circ)^2$	ϕ_ε $(^\circ)^2$	δ_s $(^\circ)^2$	T_R secs	$M_p(t)$ %	sse %	Max ϕ
CANFIS 16 rules	3637.50	1093.40	192.15	100.80	8.5	1.69	0.0	6.8
Gaussian 16 rules	3651.61	996.21	120.52	108.58	8.7	1.55	0.0	5.1

Table 7.4: Performance comparisons between the CANFIS and Gaussian yaw and roll autopilot strategies at 7.5-knots.

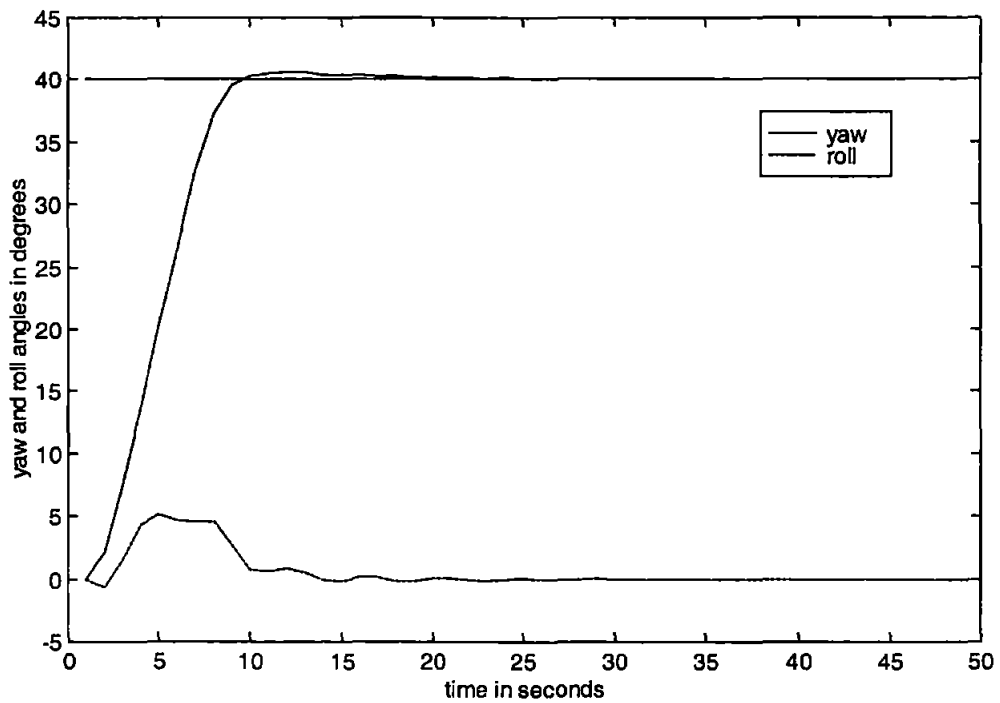


Figure 7.20: Yaw and roll responses when employing the extended hybrid rule tuned multivariable Gaussian autopilot.

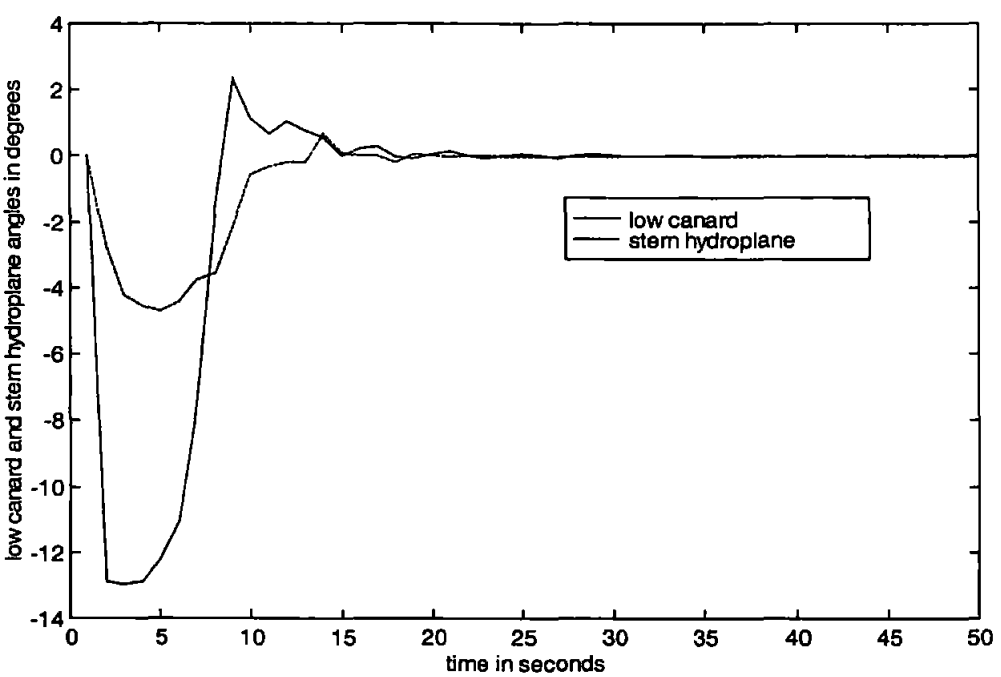


Figure 7.21: Low canard rudder and stern hydroplane responses when employing the extended hybrid rule tuned multivariable Gaussian autopilot.

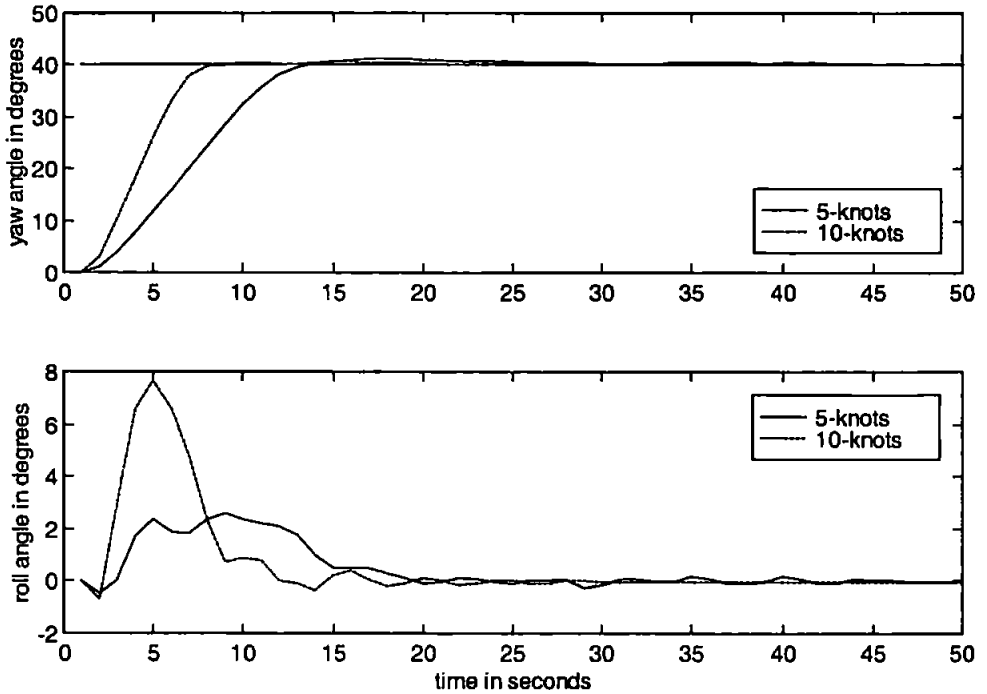


Figure 7.22: Yaw and roll responses when employing the extended hybrid rule tuned multivariable Gaussian autopilot – 5 and 10-knots.

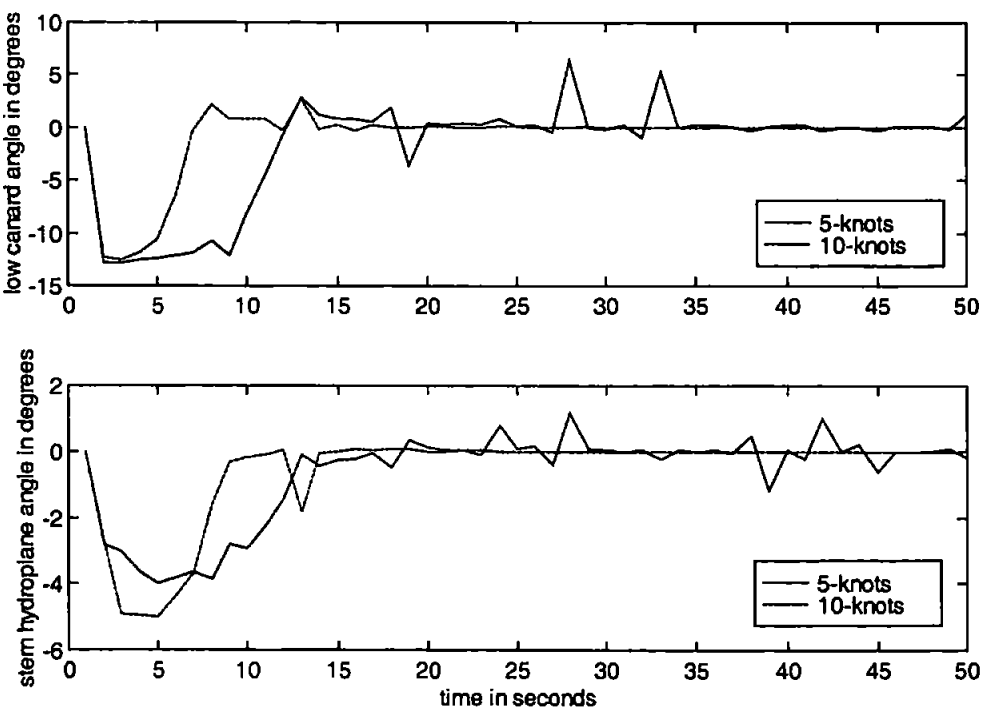


Figure 7.23: Low canard rudder and stern hydroplane responses when employing the extended hybrid rule tuned multivariable Gaussian autopilot – 5 and 10-knots.

7.7 Gaussian Autopilot Robustness

The effectiveness of the Gaussian multivariable autopilot structure is highlighted in section 7.6. To fully establish this autopilots superiority over the CANFIS multivariable autopilot of Chapter 5, it is necessary to examine the comparative robustness of each strategy. These experiments will take the form of vehicle hydrodynamic coefficient variations, sea current disturbances and noise rejection testing.

7.7.1 Vehicle Coefficient Variations

As an initial measure of the Gaussian autopilots robustness to vehicle coefficient variations, the mass of the AUV was varied; the original mass of the AUV was increased by 75% to 6300 kilograms to simulate the effects of a varying payload. Figure 7.24 and 7.25 illustrate the responses of the AUV under these robustness experiments.

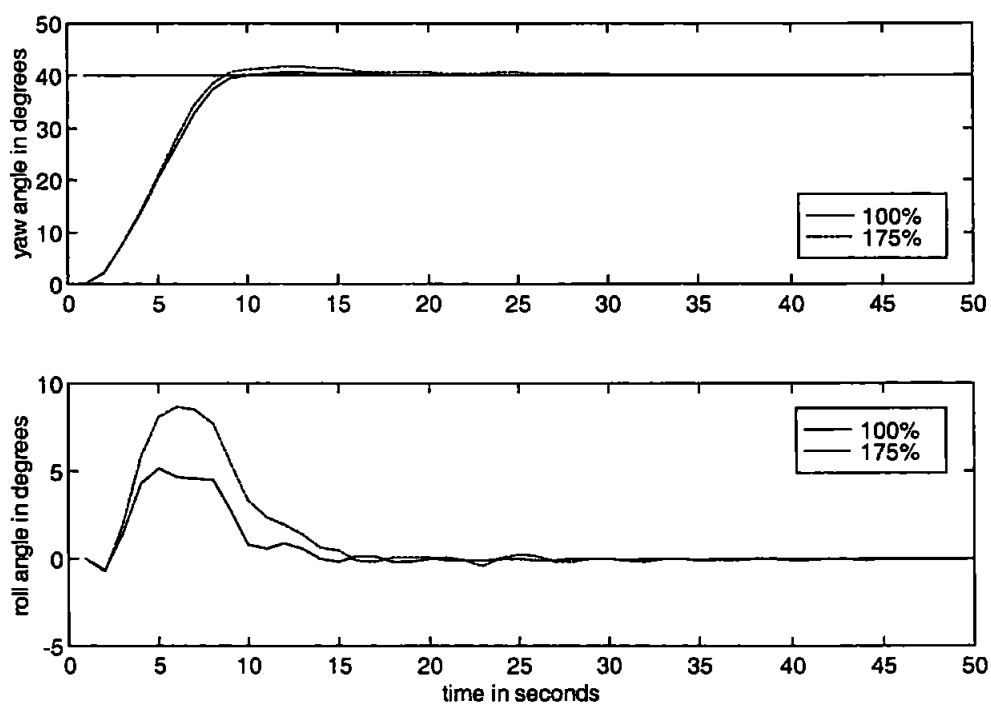


Figure 7.24: Yaw and roll responses when employing the extended hybrid rule tuned multivariable Gaussian autopilot – mass variation.

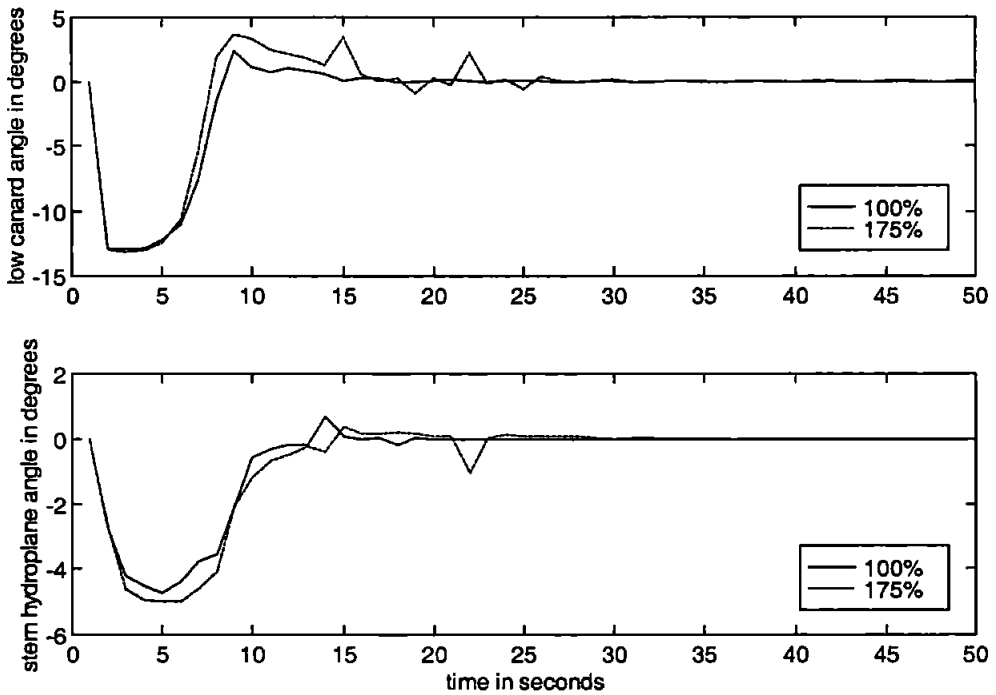


Figure 7.25: Low canard rudder and stern hydroplane responses when employing the extended hybrid rule tuned multivariable Gaussian autopilot – mass variation.

Evidently, the new Gaussian multivariable autopilot performs well in light of the increased AUV payload; comparison with Figures 5.18 and 5.19 highlights the improved roll damping achieved by using this autopilot over the CANFIS multivariable autopilot. Additionally, the following perturbations in the hydrodynamic coefficients are to be considered:

- $\pm$ 20% variation in $Y_{UV,}$ the sway damping coefficient
- $\pm$ 20% variation in $Y_{UR,}$ the yaw into sway coefficient
- $\pm$ 20% variation in $K_{UV,}$ the sway into roll coefficient

- ± 20% variation in K_{UR}, the yaw into sway coefficient
- ± 20% variation in K_{UP}, the roll damping coefficient
- ± 20% variation in N_{UV}, the sway into yaw coefficient
- ± 20% variation in N_{VR}, the roll into yaw coefficient
- ± 20% variation in N_{UR}, the yaw damping coefficient

The hydrodynamic coefficients are again considered as a function of the total velocity squared (Eqn.(4.42)) and are assumed to vary over a mission scenario. The responses of the AUV to a 40° course-changing manoeuvre when employing the Gaussian multivariable autopilot under these coefficient variations are illustrated in Figures 7.26 to 7.33.

The autopilot clearly accommodates the hydrodynamic coefficient variations in a robust manner; the responses pertaining to these simulations compare favorably with those of section 5.5.1, illustrating the superiority of the Gaussian inference for modelling the underlying AUV dynamics.

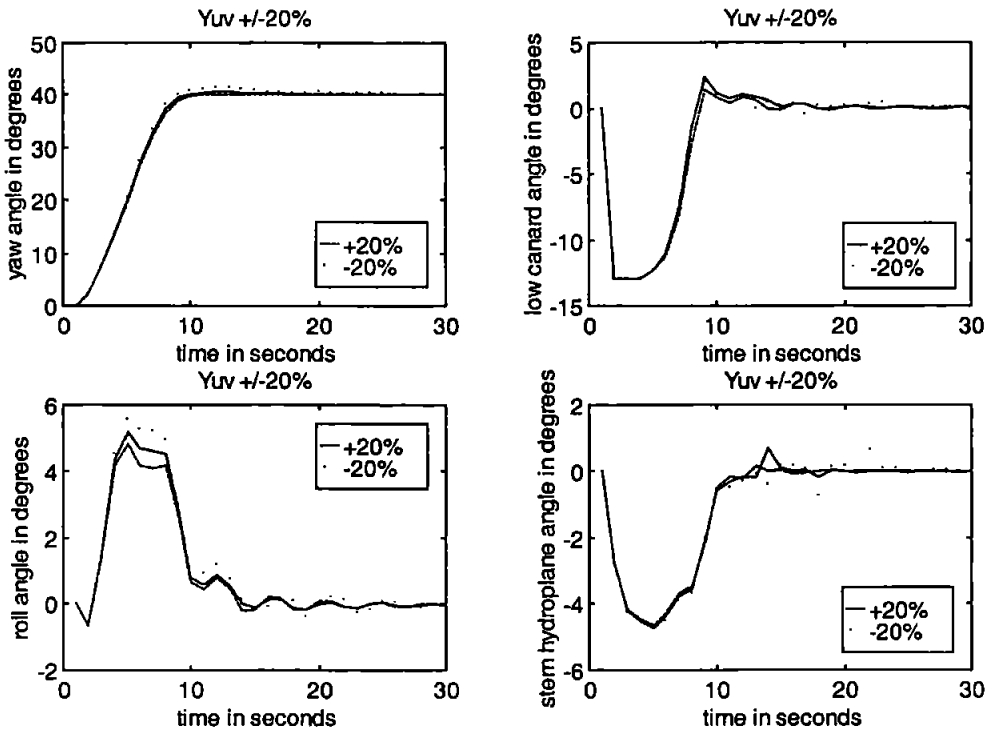


Figure 7.26: Varying Y_{uv} hydrodynamic coefficient during a 40° course-change when employing the multivariable Gaussian autopilot.

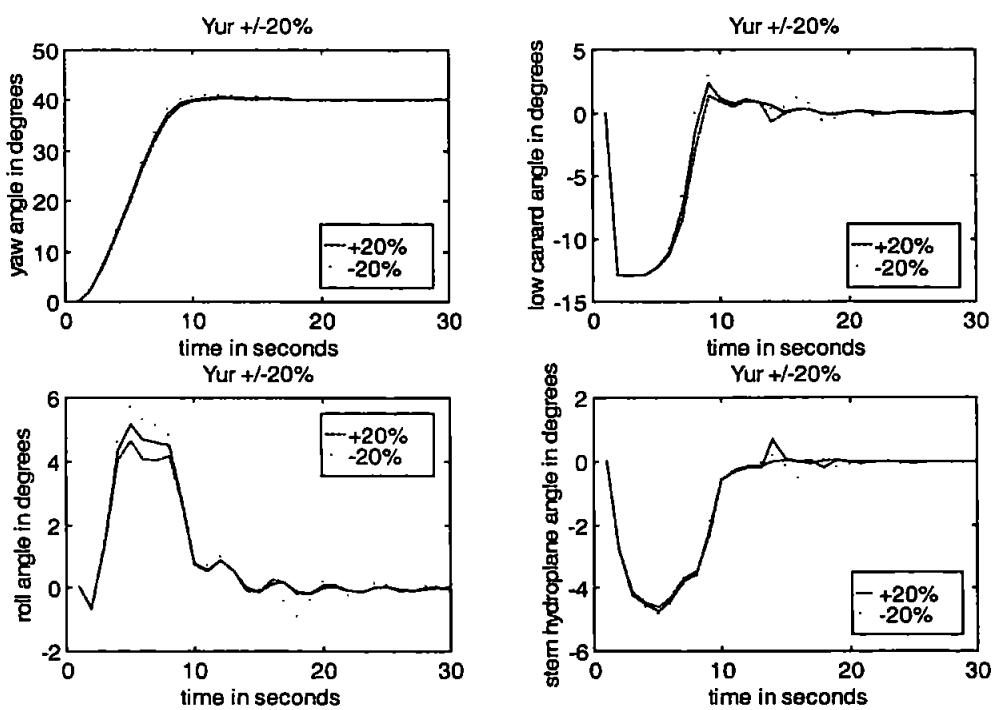


Figure 7.27: Varying Y_{ur} hydrodynamic coefficient during a 40° course-change when employing the multivariable Gaussian autopilot.

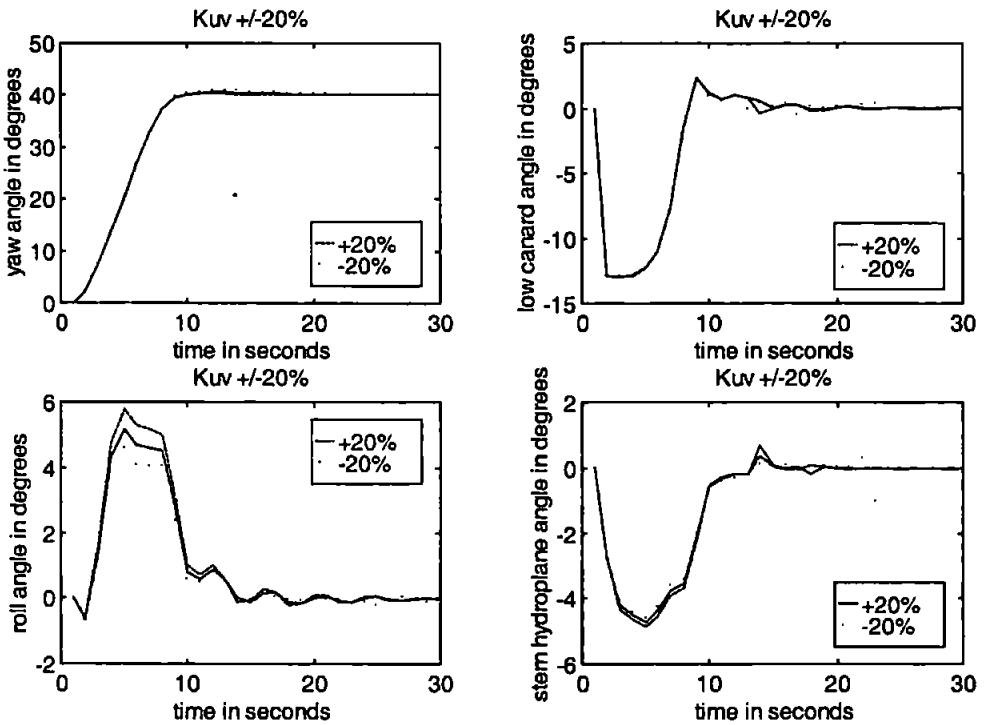


Figure 7.28: Varying K_{uv} hydrodynamic coefficient during a 40° course-change when employing the multivariable Gaussian autopilot.

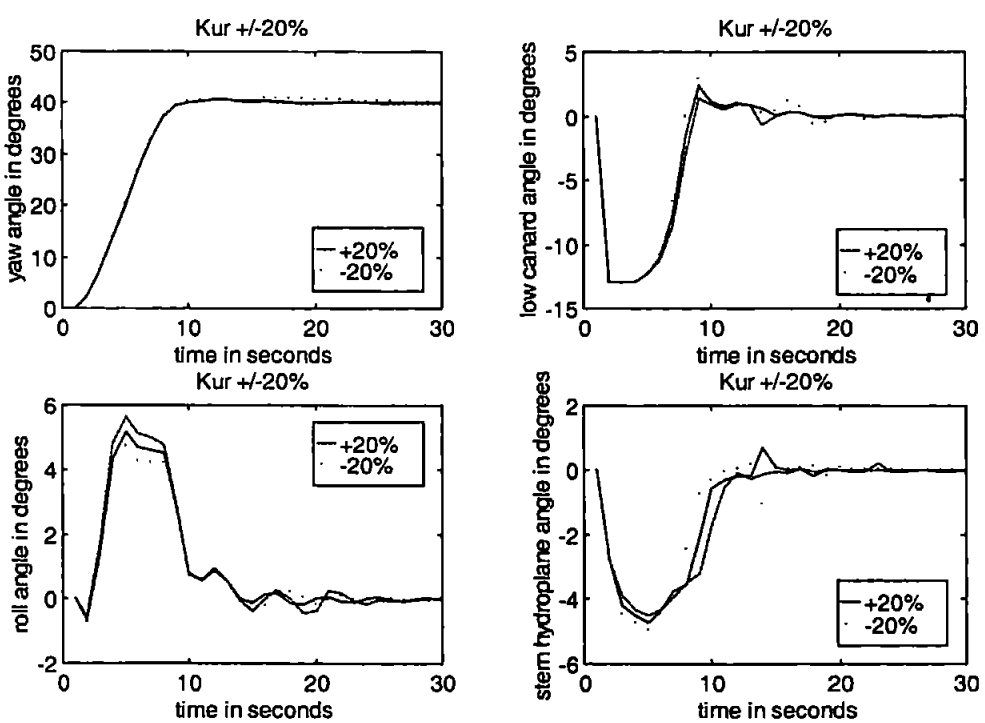


Figure 7.29: Varying K_{ur} hydrodynamic coefficient during a 40° course-change when employing the multivariable Gaussian autopilot.

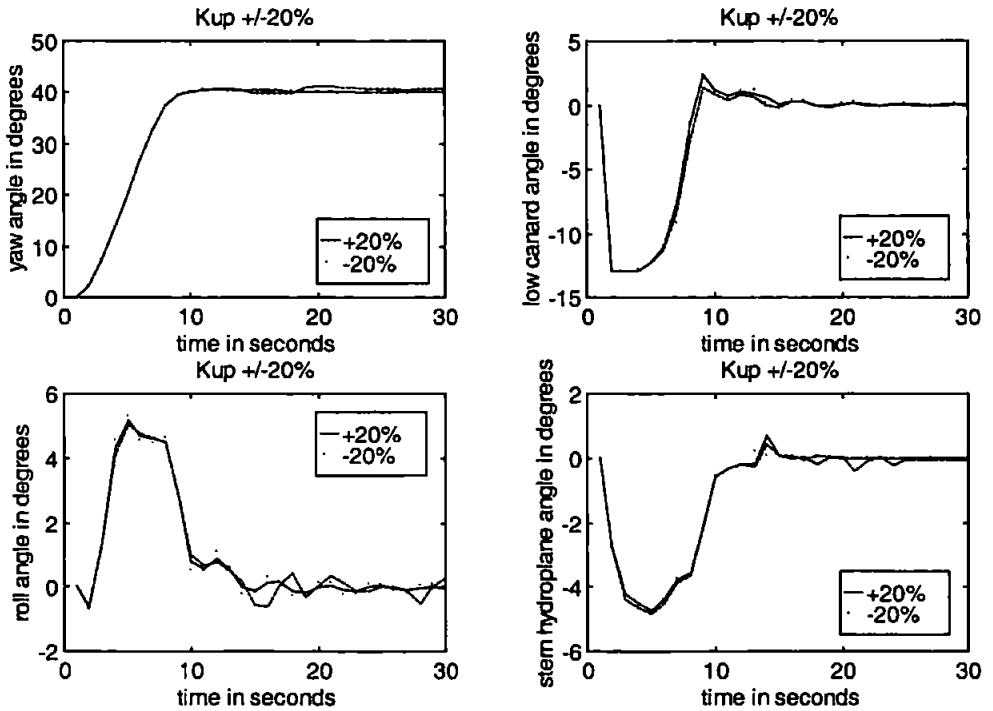


Figure 7.30: Varying K_{up} hydrodynamic coefficient during a 40° course-change when employing the multivariable Gaussian autopilot.

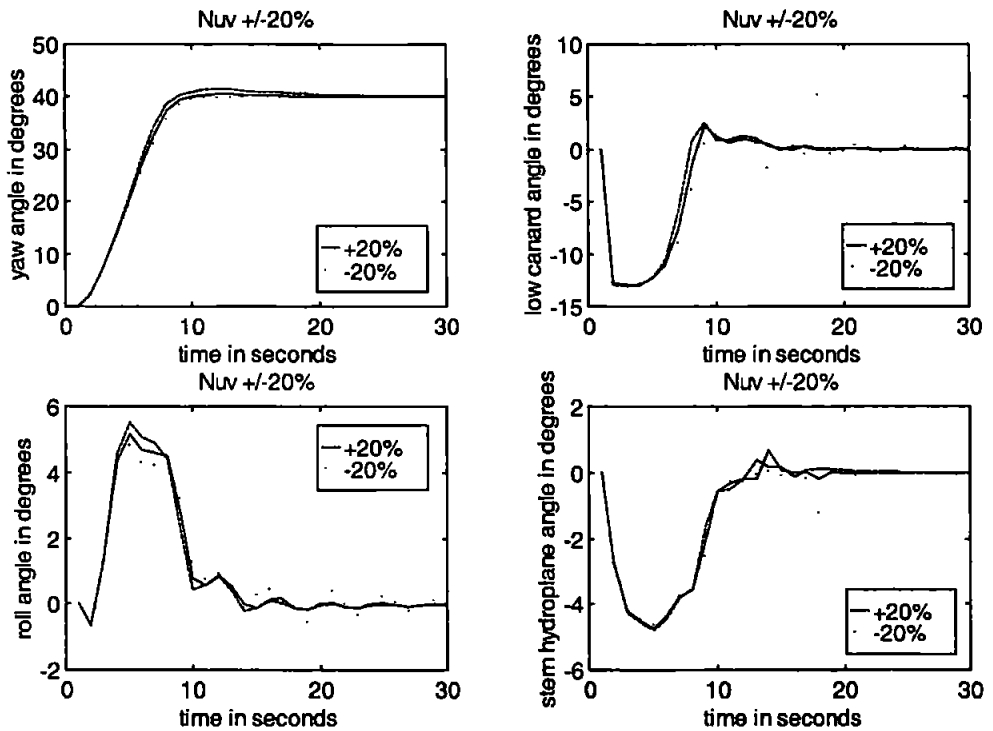


Figure 7.31: Varying N_{uv} hydrodynamic coefficient during a 40° course-change when employing the multivariable Gaussian autopilot.

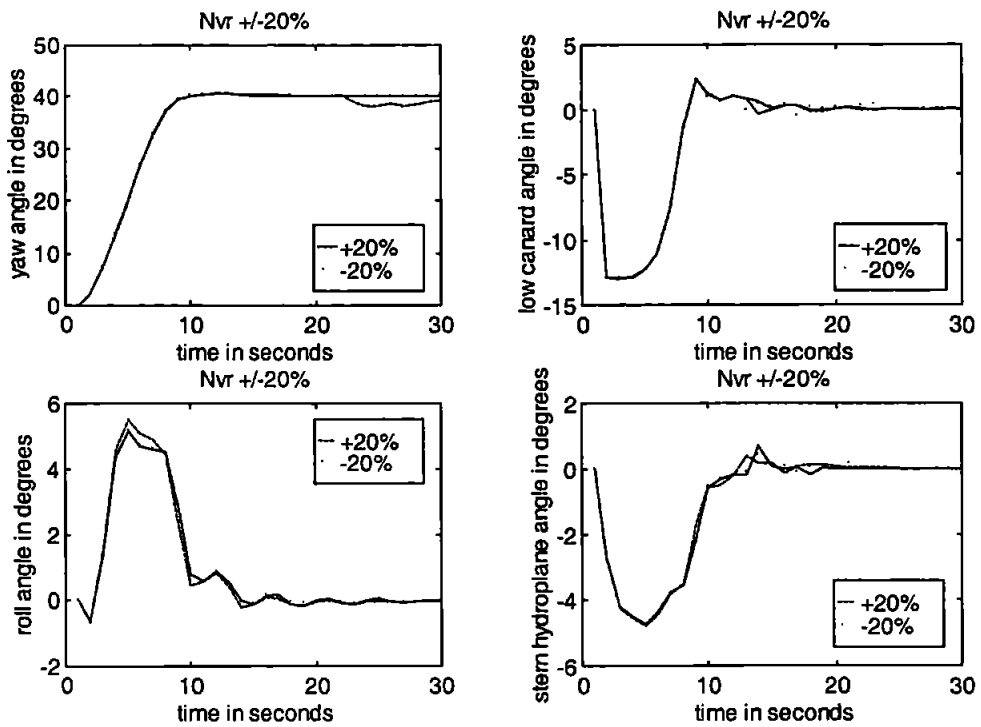


Figure 7.32: Varying N_{vr} hydrodynamic coefficient during a 40° course-change when employing the multivariable Gaussian autopilot.

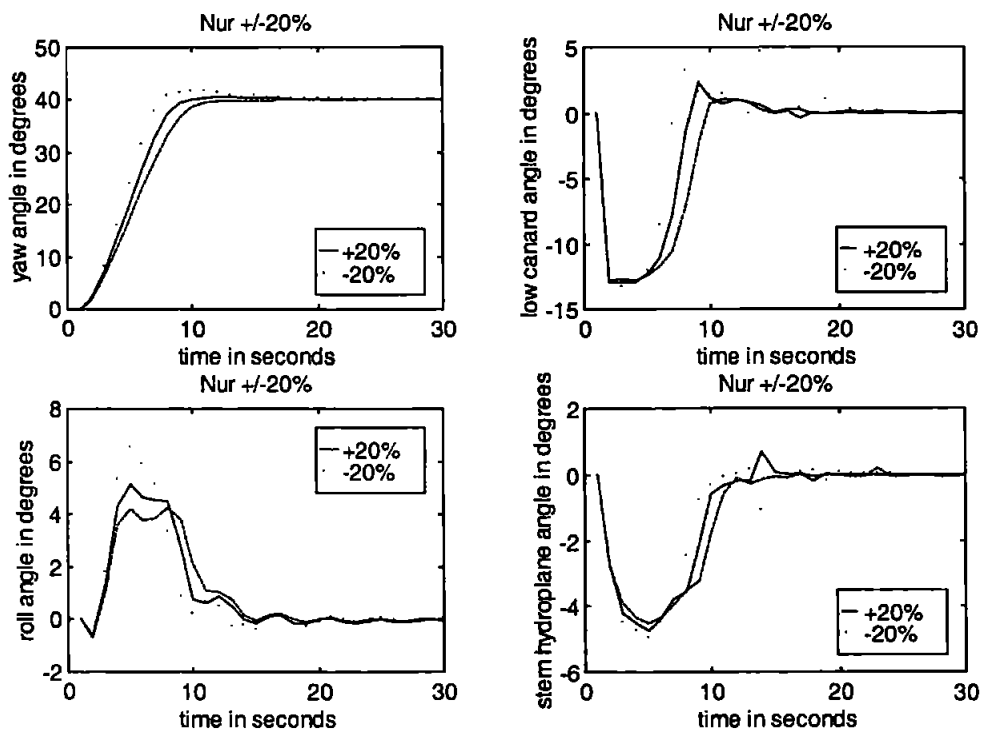


Figure 7.33: Varying N_{ur} hydrodynamic coefficient during a 40° course-change when employing the multivariable Gaussian autopilot.

7.7.2 Measurement Noise

An assessment of the effectiveness of the proposed Gaussian inference control scheme in the presence of measurement noise is provided within this section; comparisons are made with the CANFIS autopilot of Chapter 5.

The following results illustrate the course-changing and roll-minimizing ability of the Gaussian inference and CANFIS autopilots in the presence of 1%, 5% and 10% signal to noise ratios (SNR); these values represent a peak noise level of 0.25°, 1.25° and 2.5° with respect to the maximum canard rudder angle of 25.2°. These noise levels were shown previously in section 6.5.3, Figure 6.36.

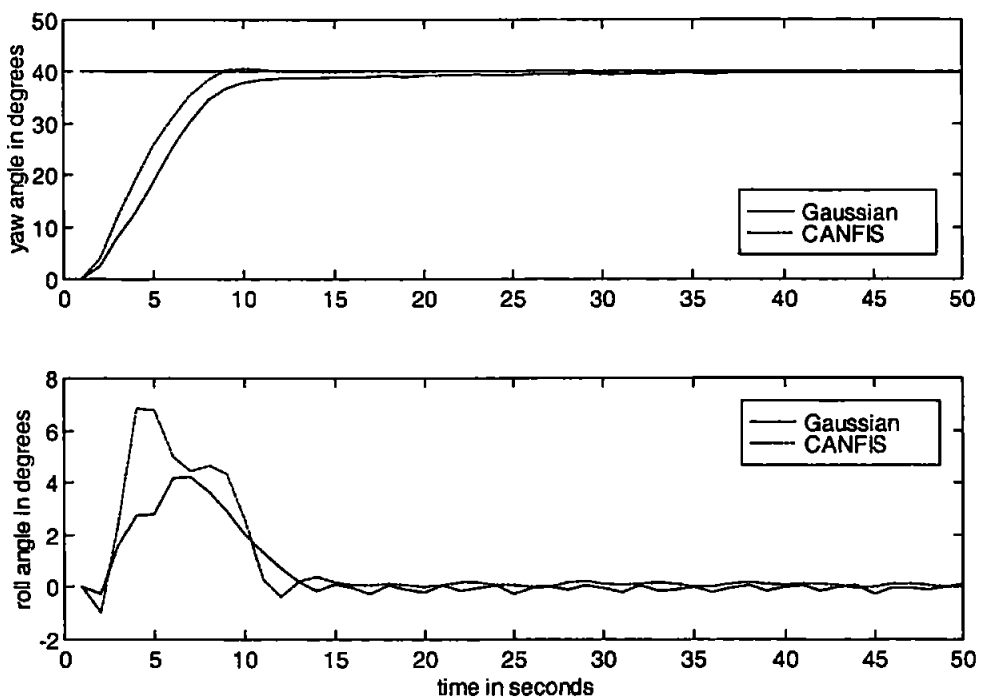


Figure 7.34: Yaw and Roll responses for the Gaussian and CANFIS autopilots in the presence of a 1% SNR.

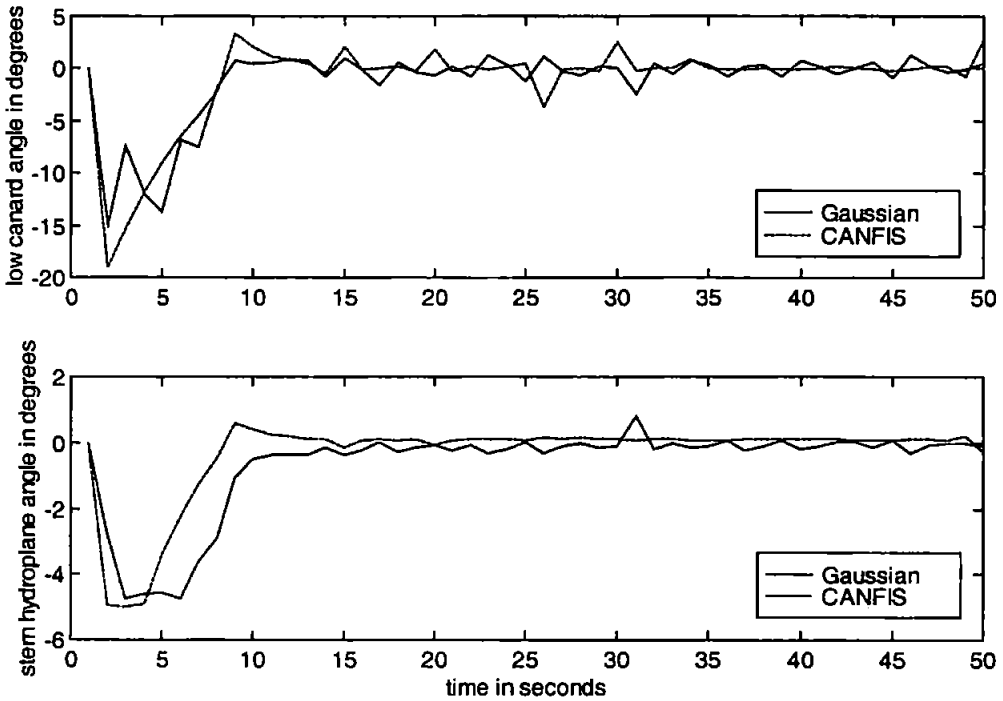


Figure 7.35: Low canard and stern hydroplane responses for the Gaussian and CANFIS autopilots in the presence of a 1% SNR.

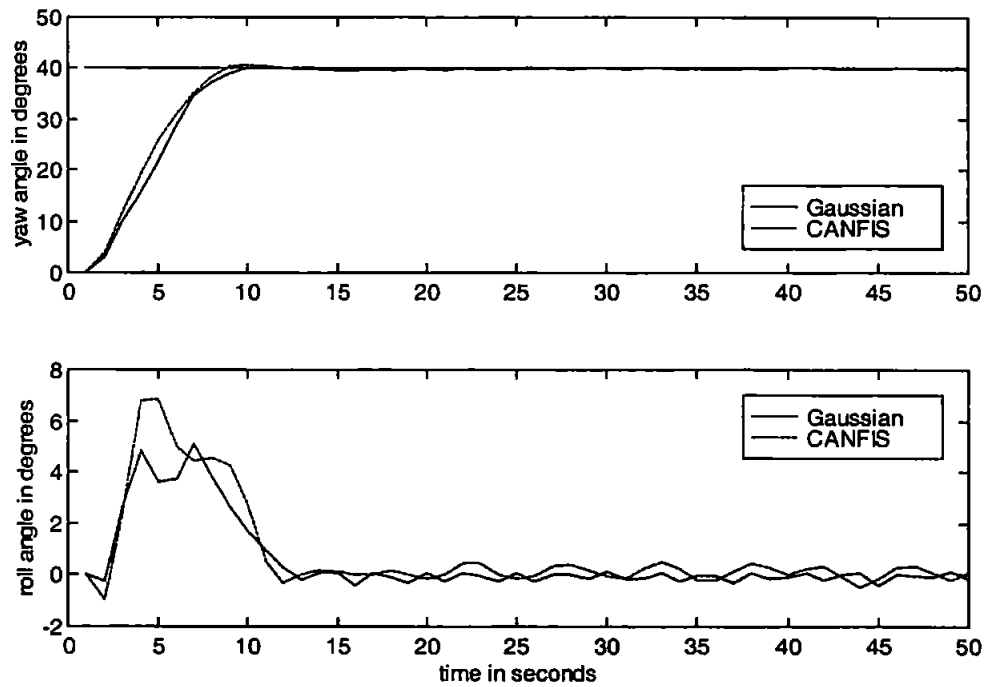


Figure 7.36: Yaw and Roll responses for the Gaussian and CANFIS autopilots in the presence of a 5% SNR.

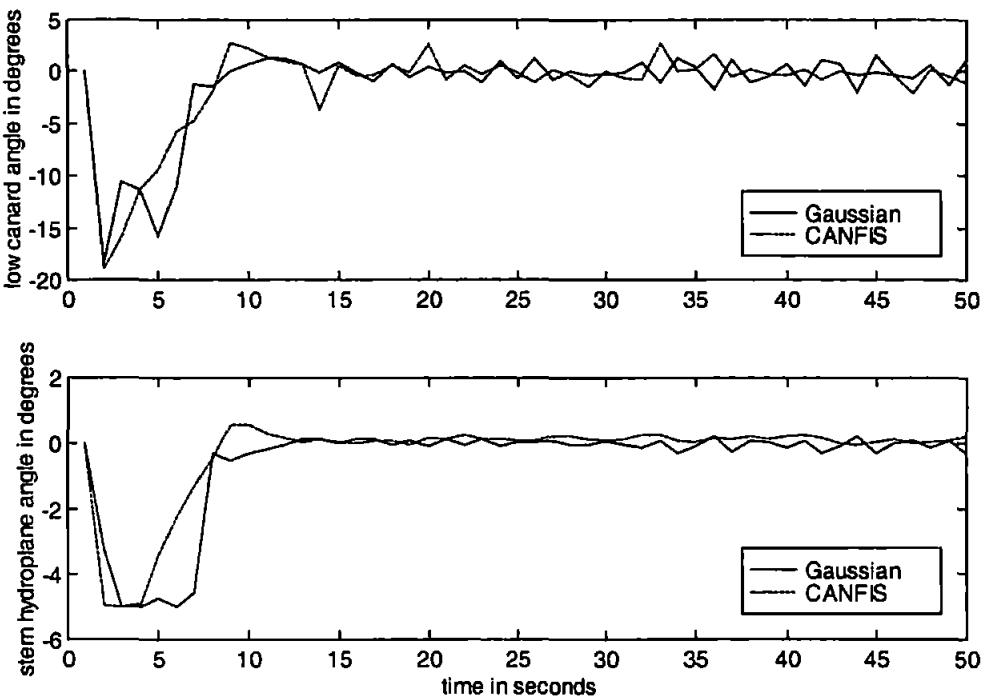


Figure 7.37: Low canard and stern hydroplane responses for the Gaussian and CANFIS autopilots in the presence of a 5% SNR.

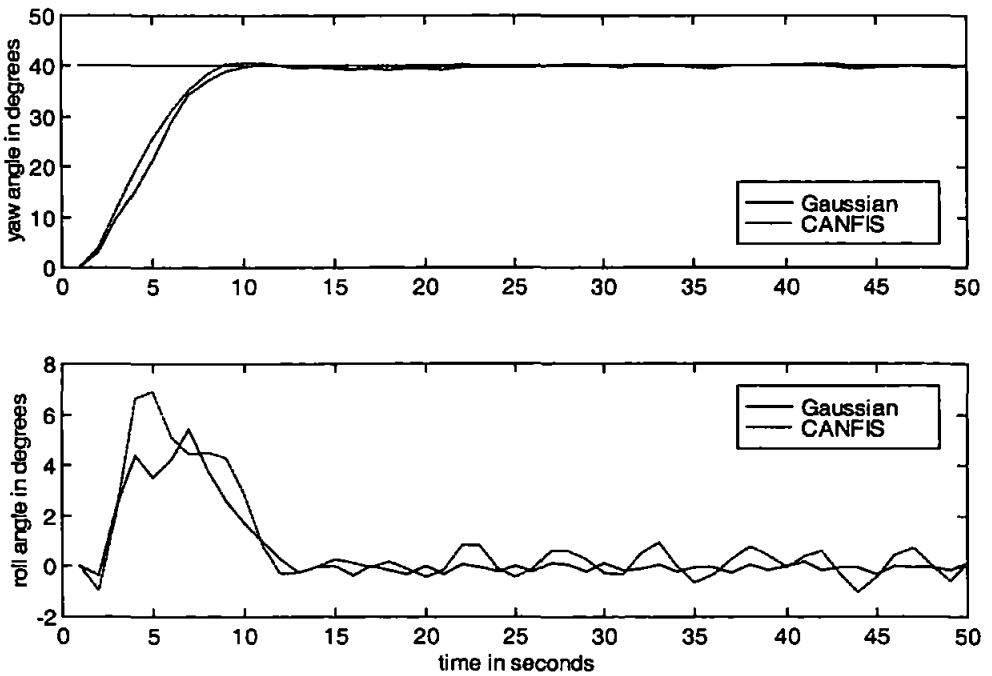


Figure 7.38: Yaw and Roll responses for the Gaussian and CANFIS autopilots in the presence of a 10% SNR.

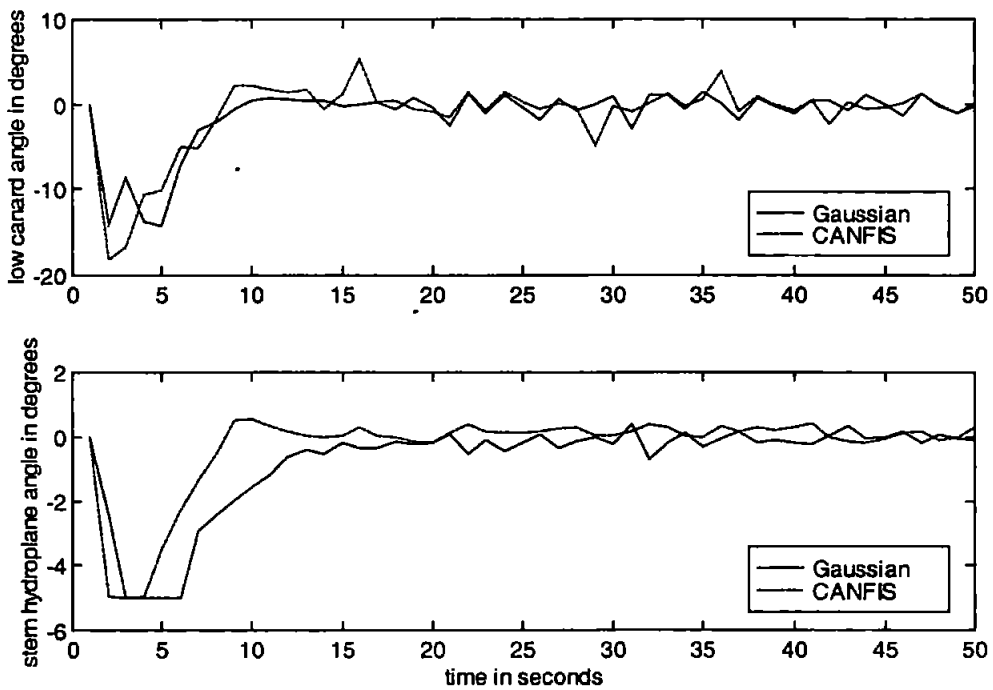


Figure 7.39: Low canard and stern hydroplane responses for the Gaussian and CANFIS autopilots in the presence of a 10% SNR.

Figure 7.34 depicts the course-changing and roll-minimizing responses of the AUV for both the Gaussian and CANFIS multivariable autopilot systems in the presence of a 1% SNR. The corresponding low canard rudder and stern hydroplane responses are reproduced in Figure 7.35. Whilst the course-changing responses are similar for each autopilot, the roll cross-coupling when employing the Gaussian autopilot is reduced. The non-linearity of the Gaussian autopilot consequent functions has been effective in accommodating the cross-coupling between the yaw and roll channels in this instance. The stern hydroplane response of Figure 7.35 highlights the control effort employed by the Gaussian autopilot to achieve this roll reduction.

The responses of each autopilot system for a 5% SNR are detailed in Figures 7.36 and 7.37. Again the Gaussian multivariable autopilot yields a superior roll response to that of the CANFIS autopilot, with a reduced peak roll of approximately 5° as compared to

7° with the CANFIS autopilot. The corresponding control effort is less oscillatory for the Gaussian autopilot at this level of measurement noise, suggesting that the non-linearity of the underlying Gaussian multivariable autopilot structure is effective at reproducing the function relating the input to the output data used to train it.

Introducing a measurement noise level of 10% leads to the AUV responses of Figures 7.38 and 7.39. One would anticipate that the Gaussian autopilot could withstand a higher level of noise than the linear output representation of the CANFIS autopilot if trained effectively. This is indeed the case, the CANFIS autopilot leads to an oscillatory roll response with a smooth stern hydroplane response suggesting that the piecewise linear outputs are not capturing the full detail within the training data. Conversely, the roll response under the Gaussian autopilot is reduced by approximately 2° and is comparatively smooth. Examination of the stern hydroplane response indicates the increased control effort employed by the Gaussian autopilot in maintaining this smooth roll response.

7.7.3 Line of Sight Guidance

Appraisal of the Gaussian multivariable autopilots robustness to sea current disturbances is performed in the manner discussed within Chapters 4, 5 and 6. The results presented (Figures 7.40 to 7.45) are shown in conjunction with the CANFIS multivariable autopilot results of section 5.5.2 for ease of comparison.

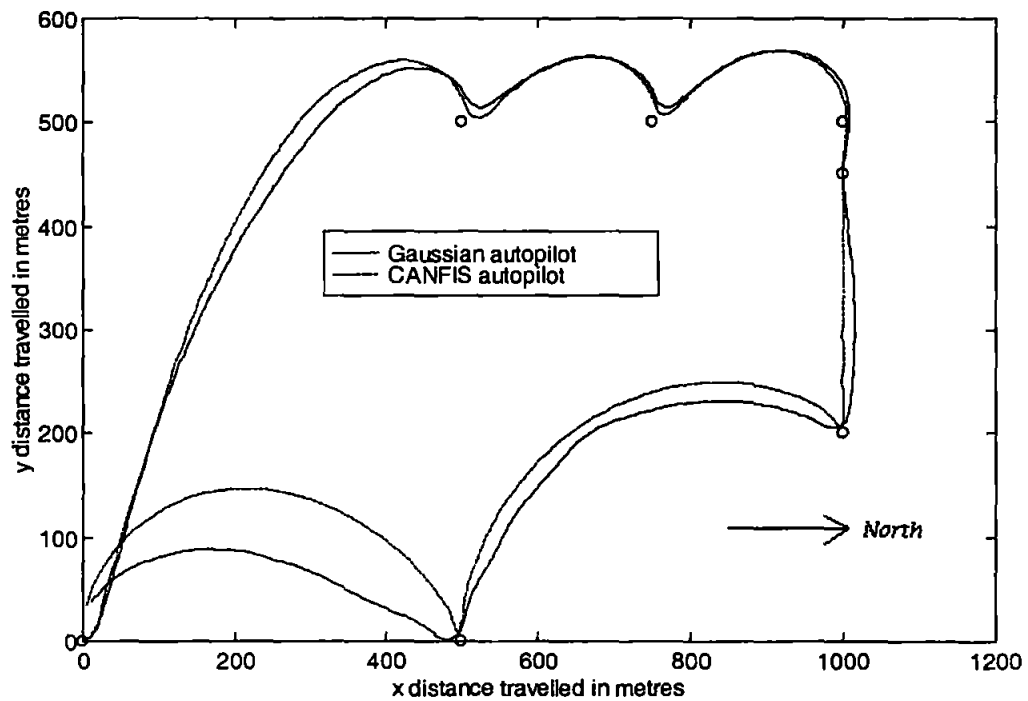


Figure 7.40: Line of sight responses over the verification track in the presence of a current disturbance of 3 ms^{-1} in the Westerly axis.

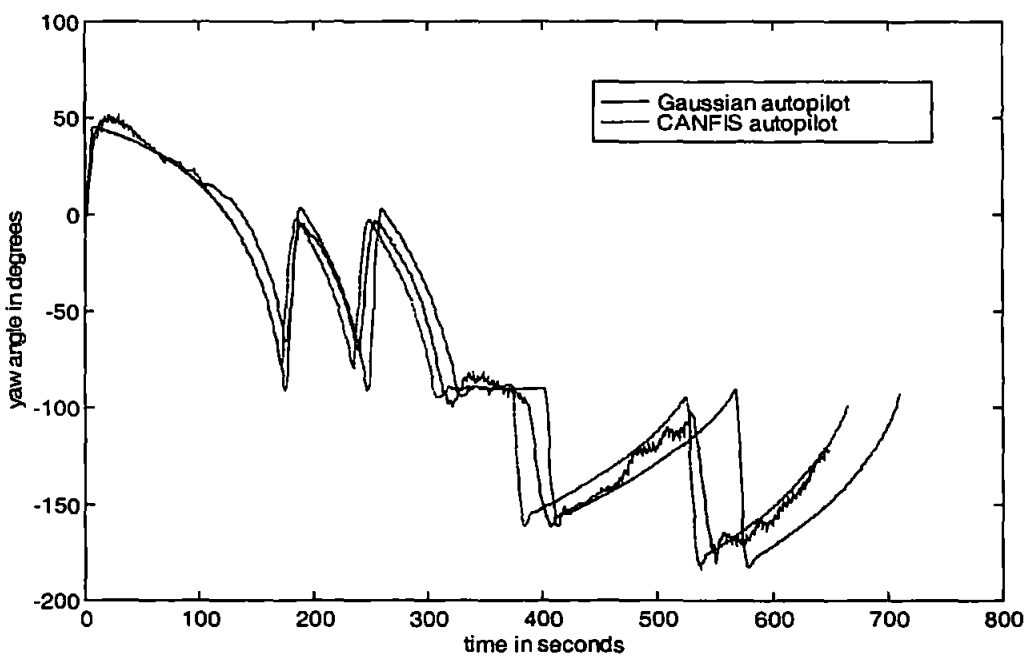


Figure 7.41: Yaw responses over the verification track in the presence of a current disturbance of 3 ms^{-1} in the Westerly axis.

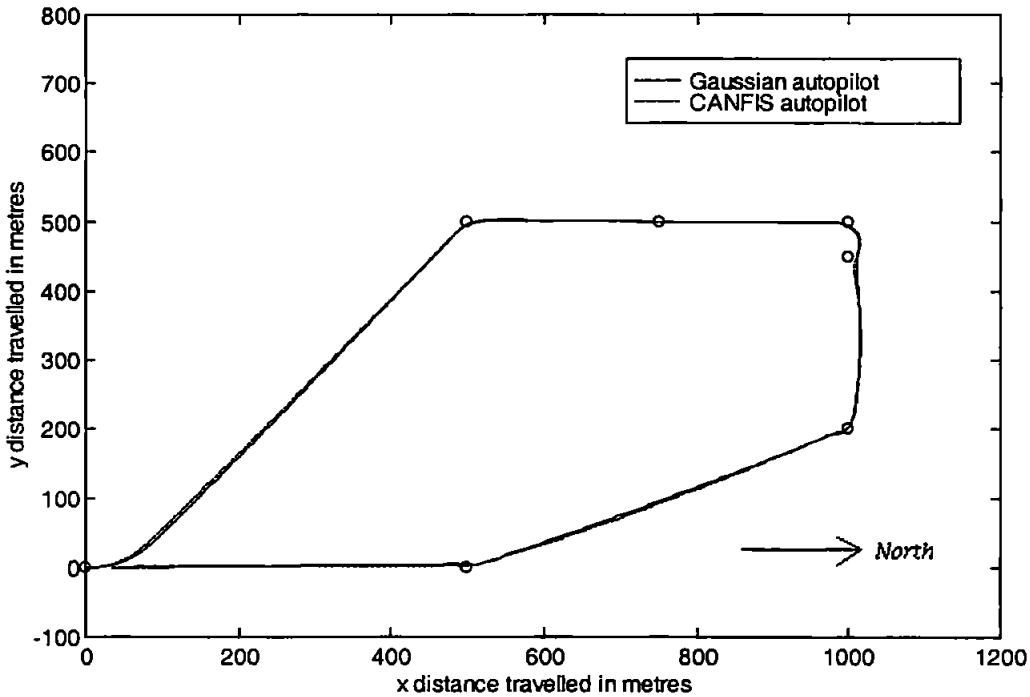


Figure 7.42: Line of sight responses over the verification track in the presence of a current disturbance of 2.5 ms^{-1} in the Northerly axis.

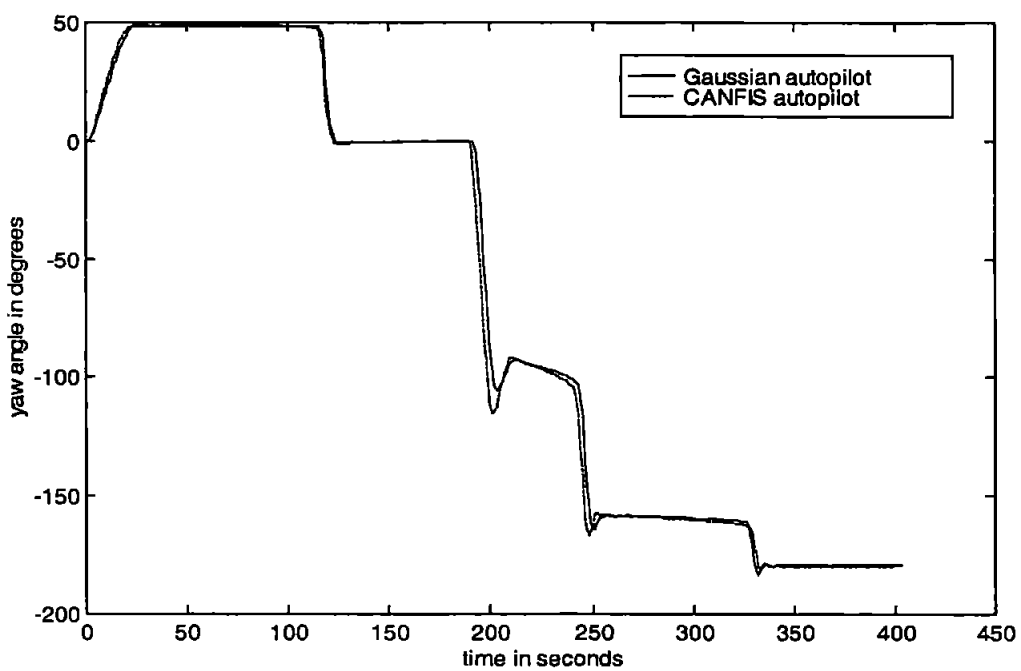


Figure 7.43: Yaw responses over the verification track in the presence of a current disturbance of 2.5 ms^{-1} in the Northerly axis.

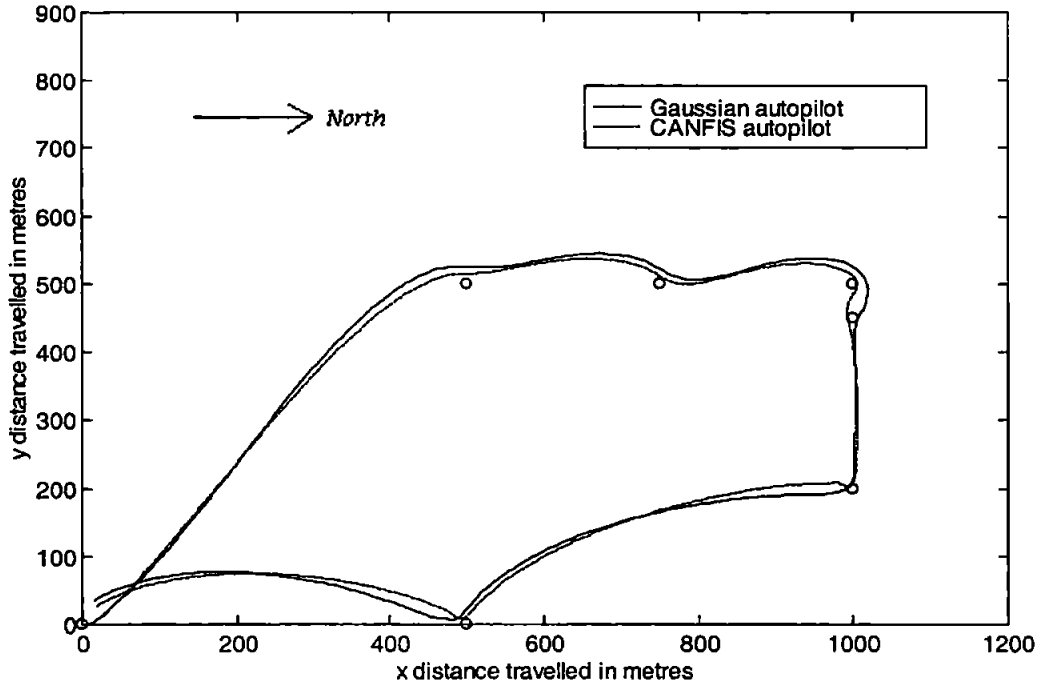


Figure 7.44: Line of sight responses over the verification track in the presence of a current disturbance of 2.83 ms^{-1} in the North Westerly direction.

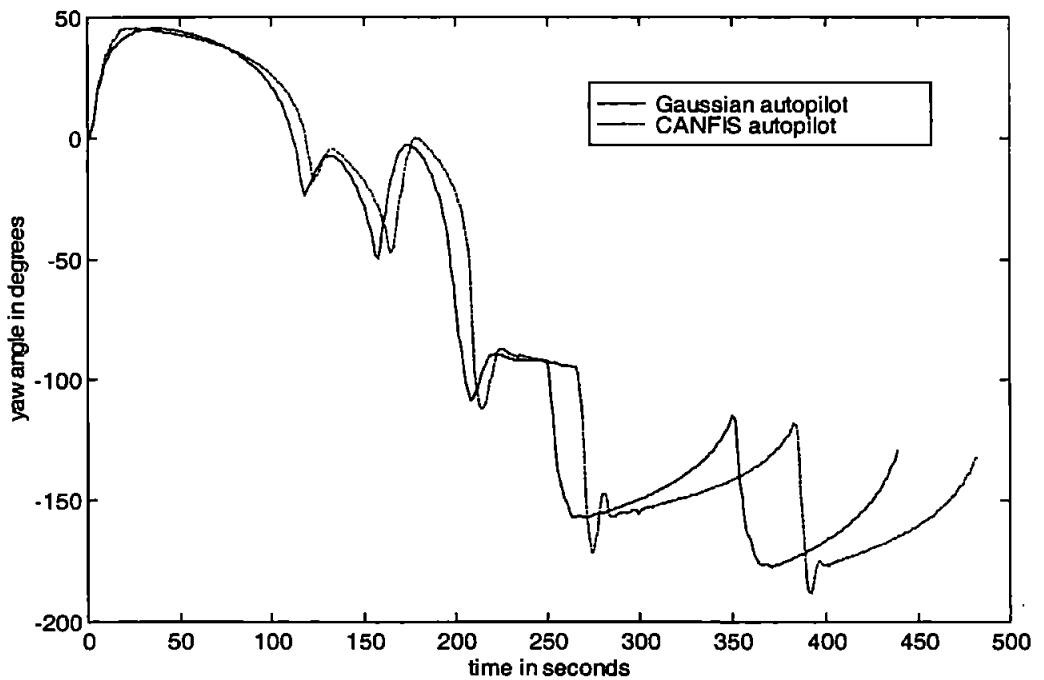


Figure 7.45: Yaw responses over the verification track in the presence of a current disturbance of 2.83 ms^{-1} in the North Westerly direction.

The AUV response over the verification track in the presence of a sea current disturbance of 3 ms^{-1} is depicted in Figure 7.40 for the Gaussian and CANFIS autopilots. The CANFIS autopilot yields larger overshoots for all the way-points; variations in forward speed (surge) caused by the current disturbance illustrate the improved generalization of the Gaussian autopilot compared to the CANFIS autopilot. Indeed, the yaw responses over this verification track (Figure 7.41) indicate the additional activity of the Gaussian autopilot; the linear outputs of the CANFIS autopilot yield a more linear yaw response (with respect to the course demands of the line of sight algorithm) than the non-linear consequent functions of the Gaussian autopilot.

Figures 7.42 and 7.43 illustrate the AUV responses in the presence of a Northerly current of 2.5 ms^{-1}. The CANFIS autopilot can be seen to produce larger overshoots at the way-points, yet there is little to choose between either autopilot in this instance.

Finally, Figures 7.44 and 7.45 represent the AUV responses in the presence of a sea current disturbance of 2.83 ms^{-1} in the North-Westerly direction (2.5 ms^{-1} in both the Northerly and Westerly axes). The yaw response (Figure 7.45) indicates the comparatively superior performance of the Gaussian multivariable autopilot over the final two way-point course changes; smaller overshoots are evident leading to more accurate guidance as displayed in Figure 7.44. When travelling between these way-points the effect of the current disturbance is to reduce the forward speed of the vehicle. Consequently, the Gaussian autopilot provides more robust control of the AUV at lower forward speeds.

7.8 Concluding Remarks

This chapter has demonstrated that a novel type of fuzzy inference, namely Gaussian fuzzy inference, can lead to an improved AUV autopilot design. Particularly, a higher control precision was gained through the use of Gaussian consequent functions, which represent non-linear fuzzy rules. This precision manifested itself in terms of reduced

roll cross-coupling with respect to a 40° course-changing manoeuvre, and improved overall control with respect to forward vehicle speed and hydrodynamic coefficient variations. Additionally, better generalization properties were highlighted, arising due to smoother interpolation between control rules.

The use of composite Gaussian RBF networks can provide a non-linear modelling technique which can be tuned with a linear algorithm such as least-squares. However, the work documented within this chapter highlighted the improvements to be gained by employing a more sophisticated algorithm which made full use of the extra non-linearity introduced within the fuzzy model. Additionally, these local output models retain past knowledge more successfully and thus improve the intelligence gathering process.

References

Bossley, K. M. (1997). Neurofuzzy Modelling Approaches in System Identification. Ph.D Thesis, University of Southampton.

Brown, M. and Harris, C.-J. (1994). Neurofuzzy Adaptive Modelling and Control. Prentice Hall, Hemel Hempstead.

Choi, J.-L. and Hwang, C.-S. (1997). On the Fuzzy-Neural Controller for a Multivariable Nonlinear System. Proceedings of the 2nd Asian Conference, Seoul, South Korea, Vol. III, pp631-634.

Golub, G. H. and Van Loan, C. F. (1989). Matrix Computations. 2nd Edition. Johns-Hopkins University Press, Baltimore, U.S.A.

Hardy, R. L. (1971). Multiquadric Equations of Topography and Other Irregular Surfaces. Journal of Geophysical Research, Vol. 76, No. 8, pp1905-1915.

Heiss, M. and Kampl, S. (1996). Multiplication-Free Radial Basis Function Network. IEEE Transactions on Neural Networks, Vol. 7, No. 6, pp1461-1464.

Horikawa, S.-I., Furuhashi, T. and Uchikawa, Y. (1992). On Fuzzy Modelling using Fuzzy Neural Networks with the Back-Propagation Algorithm. IEEE Transactions on Neural Networks, Vol. 3, No. 5, pp801-806.

Hunt, J. K. and Johansen, T. A. (1997). Design and Analysis of Gain-Scheduled Control using Local Controller Networks. International Journal of Control, Vol. 66, No. 5, pp619-651.

Hu, J.-Q. (1997). Adaptive Fuzzy Predictive Control using a Neuro-Fuzzy Model with Application to Sintering. Ph.D Thesis, University of Sheffield, U.K.

Jang, J.-S. R., Sun, C.-T. and Mizutani, E. (1997). Neuro-Fuzzy and Soft Computing. Prentice-Hall, New Jersey.

Lin, C.-T. and Juang, C.-F. (1997). An Adaptive Fuzzy Filter and its Applications. IEEE Transactions on Systems, Man and Cybernetics – Part B: Cybernetics, Vol. 24, No. 4, pp635-656.

McDowell, D. M., Irwin, G. W., Lightbody, G. and McConnell, G. (1997). Hybrid Neural Adaptive Control for Bank-to-Turn Missiles. IEEE Transactions on Control Systems Technology, Vol. 5, No. 3, pp297-308.

Poggio, T. and Girosi, F. (1989). A Theory of Networks for Approximation and Learning. A. I. Memo No. 1140, Center for Biological Information Processing, Whitaker College, A. I. Laboratory, Massachusetts Institute of Technology.

Poggio, T. and Girosi, F. (1990a). Regularization Algorithms for Learning That Are Equivalent to Multilayer Networks. Science, Vol. 247, pp978-982.

Poggio, T. and Girosi, F. (1990b). Networks for Approximation and Learning. Proceedings of the IEEE, Vol. 78, No. 9, pp1481-1497.

Powell, M. J. D. (1992). The Theory of Radial Basis Function Approximation in 1990. Advances in Numerical Analysis II. W. Light Editor, Oxford University Press, Oxford U.K.

Takagi, T. and Sugeno, M. (1985). Fuzzy Identification of Systems and its Applications to Modelling and Control. IEEE Transactions on Systems, Man and Cybernetics, Vol. 15, pp116-132.

White, D. A. and Sofge, D. A. (1992). Handbook of Intelligent Control. Multiscience Press Inc., Van Nostrand Reinhold, New York, pp52-56.

Chapter 8

Concluding Remarks

8.1 Discussion

The publication of Jang's thesis in 1992 presented a clear method by which a fuzzy inference system (FIS) could be tuned using neural network learning algorithms. This fusion eased the inherent lack of interpretability within typical neural network models, whilst allowing the automatic tuning of poorly designed FISs.

This thesis has exploited Jangs approach to FIS tuning in a novel manner and has lead to the development of multivariable co-active-ANFIS (CANFIS) type structures, as well as new Gaussian inference control algorithms and architectures. Consequently, a wider class of control architectures, and algorithms for tuning FISs is now available.

Specifically, the combination of the CANFIS autopilot of Chapter 5 with a radial basis function network (RBFN) model within the consequent portion of the controller provided more accurate and robust AUV control through more accurate representations

of the underlying training data. This fusion generated the requirement for the development of the extended hybrid learning rule of Chapter 7, which proved effective for tuning the new local controller models of the Gaussian inference system architecture.

Whilst this work has produced numerous technical developments in AUV control system design, there are clear avenues for exploration in the future. Section 8.3 outlines recommendations for follow up research, driven by the conclusions drawn from the thesis.

8.2 Research Objectives

The objectives of the programme of research were outlined in Chapter 1 and are reproduced here for ease of reference:

(a) Critically review the current UUV control literature

(b) Define non-linear models pertaining to the yaw and roll degrees of freedom

(c) Develop traditional single-input single-output (SISO) control algorithms for the yaw and roll degrees of freedom

(d) Investigate various neuro-fuzzy algorithms and structures for yaw and roll control

(e) Produce candidate neuro-fuzzy control algorithms for each degree of freedom

(f) Employ suitable neuro-fuzzy algorithms within a multi-input multi-output (MIMO) configuration to control yaw and roll simultaneously – this configuration should be flexible to allow full six degree of freedom control

(g) Critically assess the performance of the chosen multivariable controller configuration and provide flexible alternatives to the underlying algorithm

Objectives (a) to (e) are discussed within Chapters 2 to 5; specifically, the work within Chapter 4 details the results pertaining to objectives (b) to (e), and Chapter 5 documents work satisfying objective (f) and (g). Whilst (f) required the development of full six degree of freedom control architectures, Chapters 5, and 7 illustrated the technique with respect to two degrees of freedom. However, the technique is applicable to all six degrees of freedom if required.

Satisfying these objectives lead to the following work being included within the thesis:

(h) Examine the suitability of on-line control algorithms with respect to the chosen neuro-fuzzy architecture. Provide a comparative performance assessment between the off-line and on-line control algorithms and recommend a candidate algorithm, suitable for controlling multiple AUV degrees of freedom.

(i) Test the robustness of each candidate control architecture to sea current disturbances and variations within the hydrodynamic coefficients of the AUV.

By incorporating these additional objectives (within Chapter 6 and Chapters 4, 5, 6 and 7 respectively) a more complete description of the factors which influence AUV control architectures was provided. Additionally, this work raised questions on the suitability of various architectures and algorithms; these elements of the work may have escaped further attention had these objectives not been included and thus the work within the thesis has been enhanced beyond the original proposal.

8.3 Recommendations for Future Research

Several different directions for future research have been highlighted through the completion of the work within this thesis. The following points provide a summary of these areas and are not considered to be exhaustive:

- Some form of network reduction and/or pruning when employing the multivariable autopilot to reduce the dimension of the parameter space and thus remove ineffective rules and parameters from the rulebase is envisaged. Various techniques for achieving this aim are discussed within the literature (Bossley (1997), Maeda and De Figueiredo (1997), Mascarilla (1997)).

- The effectiveness of adaptive fuzzy control methods is discussed throughout this thesis. However, the methods considered herein represent parameter adaptive methods, as opposed to approaches that adapt the structure of the controller architecture. Some work is on-going in this area at present [Kuo et *al.* (1994), Lin and Lee (1994), Nie and Linkens (1993)] and to date is proving extremely effective in producing computationally inexpensive fuzzy and neuro-fuzzy architectures. Inherent problems concerning the curse of dimensionality are thus often circumvented. The autopilot structures developed within this thesis are all static in nature and as such cannot create or modify their architectures

to accommodate dynamic variations in the AUV system. Whilst the chosen autopilots are extensively tested, the ability to add and subtract relevant information within such a paradigm could prove very useful if there is a requirement to have controller reconfiguration.

- Testing the significance of rules after tuning off-line to reduce rule-base dimension could also lead to reduced storage requirements within parameter matrices. This suggests the formulation of suitable hypothesis testing techniques, measuring the contribution of each parameter within the holistic controller output. For example, Bossley (1997) examined the design of parsimonious neuro-fuzzy models. This approach considered model transparency and tackled the curse of dimensionality directly. However, the *B*-spline approach adopted therein is not considered the most effective for controller design. Furthering Bossley's efforts, in the context of the work considered within chapter 7 of this thesis, could provide parsimonious fuzzy structures for application to a wider range of control problems.

- The calculation of Gaussian consequent functions within Chapter 7 depends upon a 1 dimensional cross-over point as detailed in Lin and Lu (1995). The use of an α-level set approach to effect cross-over within the calculation of the Gaussian consequent functions could improve the accuracy of the control signal calculation.

- Clearly, as mentioned in many recent fuzzy control articles, more research into the stability of fuzzy controllers is required.

References

Bossley, K. M. (1997). Neurofuzzy Modelling Approaches in System Identification. Ph.D thesis, University of Southampton, U.K.

Jang, J.-S. R. (1992). Neuro-Fuzzy Modelling: Architecture, Analyses and Applications. Ph.D thesis, University of California, Berkeley, U.S.A.

Kuo, R. J., Cohen, P. H. and Kumara, S. R. T. (1994). Neural Network Driven Fuzzy Inference System. Proceedings of the International Conference on Neural Networks, pp1532-1536.

Lin, C.-T. and Lee, C. S. G. (1994). Reinforcement Structure/Parameter Learning for Neural Network-Based Fuzzy Logic Control Systems. IEEE Transactions on Fuzzy Systems, Vol. 2, No. 1, pp46-63.

Lin, C.-T. and Lu, Y.- C. (1995). A Neural Fuzzy System with Linguistic Teaching Signals. IEEE Transactions on Fuzzy Systems, Vol. 3, No. 2, pp169-189.

Maeda, Y. and De Figueiredo, R., J., P. (1997). Learning Rules for Neuro-Controller via Simultaneous Perturbation. . IEEE Transactions on Neural Networks, Vol. 8, No. 5, pp1119-1130.

Mascarilla, L. (1997). Fuzzy Rules Extraction and Redundancy Elimination: An Application to Remote Sensing Image Analysis. International Journal of Intelligent Systems, Vol. 12, pp793-817.

Nie, J. and Linkens, D. A. (1993). Learning Control Using Fuzzified Self-Organizing Radial Basis Function Network. IEEE Transactions on Fuzzy Systems, Vol. 1, No. 4, pp280-287.

Appendix A: Publications

The work within this thesis has contributed to the field via the following publication list:

- Craven, P. J., Sutton, R. and Kwiesielewicz, M. (1997). Non-Linear Dynamic System Identification using a Recurrent Neural Network. Proceedings of the IASTED International Conference on Modelling and Simulation, Pittsburgh, USA, pp15-17.

- Craven, P. J., Kwiesielewicz, M. and Sutton, R. (1997). Dynamic System Control using ANFIS Architecture Based Controller. Proceedings of the 15th Polish Conference on Poly-Optimization and CAD, Mielno, Poland, pp69-75.

- Sutton, R. and Craven, P. J. (1997). A Neuro-Fuzzy Autopilot for Unmanned Underwater Vehicle Control. Proceedings of the 2nd Asian Conference, Seoul, Korea, Vol. 1, pp541-544.

- Craven, P. J., Sutton, R., Tiano, A. and Magenes, G. (1997). A Design Study of a Neuro-Fuzzy Autopilot for an Autonomous Underwater Vehicle. Proceedings of the 4th International Symposium on Methods and Models in Automation and Robotics, Miedrzyzdroje, Poland, Vol. 2, pp491-498. (Invited Paper).

- Kwiesielewicz, M., Craven, P. J. and Sutton, R. (1997). Modelling the Yaw Behaviour of a UUV using a Dynamic Neural Network. Proceedings of the 12th International Conference on Systems Engineering, Coventry, U.K., Vol. 2, pp410-413.

- Craven, P. J., Sutton, R. and Burns, R. S. (1997). Intelligent Course Changing Control of an Autonomous Underwater Vehicle. Proceedings of the 12th International Conference on Systems Engineering, Coventry, Vol. 1, pp159-164.

- Craven, P. J., Sutton, R. and Dai, Y.-M. (1997). A Neurofuzzy Technique Applied to UUV Autopilot Design. Proceedings of the 4th IFAC Conference on Manoeuvring and Control of Marine Craft (MCMC 97), Brijuni, Croatia, pp25-30.

- Tiano, A., Magenes, G., Sutton, R. and Craven, P. J. (1997). Identification Methods Applied to an Unmanned Underwater Vehicle. Proceedings of the 4th IFAC Conference on Manoeuvring and Control of Marine Craft (MCMC 97), Brijuni, Croatia, pp36-41.

- Sutton, R. and Craven, P. J. (1997). Fuzzy Yaw Autopilots for Unmanned Underwater Vehicles Tuned using Artificial Neural Networks. Underwater Technology, Vol. 22, No. 4, pp173-182.

- Craven, P. J., Sutton, R. and Burns, R. S. (1998). Control Strategies for Underwater Vehicles. Journal of Navigation, Vol. 51, No. 1, pp79-105.

- Craven, P. J., Kwiesielewicz, M. and Sutton, R. (1998). Identification of Underwater Vehicle Inverse Model using Recurrent Neural Network and genetic Algorithms. Proceedings of the 16th Polish Conference on Poly-Optimization and CAD, Mielno, Poland, pp75-78.

- Craven, P. J. and Sutton, R. (1998). A Neural Network Based Fuzzy Autopilot Design for an Autonomous Underwater Vehicle. Proceedings of the 3rd IFAC Symposium on Intelligent Autonomous Vehicles, Vol. II, Madrid, Spain, pp417-422.

- Sutton, R., Craven, P. J. and Solomon, C. M. (1998). Intelligent Roll Control of an Autonomous Underwater Vehicle. Proceedings of the Application of Multivariable System Techniques Symposium (AMST 98), Bradford, U.K., pp19-28.

- Craven, P. J., Sutton, R. and Dai, Y.-M. (1998). Multi-Input Single-Output Autopilots for Autonomous Underwater Vehicle Control. Proceedings of the 1998 Summer Computer Simulation Conference, Reno, Nevada, USA, pp87-92.

- Craven, P. J., Sutton, R. and Kwiesielewicz, M. (1998). Neuro-Fuzzy Control of a Non-Linear Multivariable System. Proceedings of the UKACC Control 98 Conference, Swansea, U.K., Vol. 1, pp531-536.

- Sutton, R. and Craven, P. J. (1998). The ANFIS Approach Applied to AUV Autopilot Design. Neural Computing and Applications, Vol. 7, pp131-140.

- Sutton, R. and Craven, P. J. (1998). Control of Unmanned Underwater Vehicles. Proceedings of the Workshop on Offshore Technologies for Aquaculture, Haifa, Israel, pp1-17. (Invited Paper).

These papers are presented overleaf in chronological order for completeness.

The following papers have been accepted for publication:

- Craven P. J., Sutton, R., Burns, R. S. and Dai, Y.-M. Multivariable Intelligent Control Strategies for an Autonomous Underwater Vehicle. International Journal of Systems Science.

- Sutton, R., Burns, R. S., Dai, Y.-M. and Craven P. J. (1998). Intelligent Steering Control of an Autonomous Underwater Vehicle. Journal of Navigation.

NON-LINEAR DYNAMIC SYSTEM IDENTIFICATION USING A RECURRENT NEURAL NETWORK

PAUL J. CRAVEN and ROBERT SUTTON
Institute of Marine Studies
University of Plymouth
Drake Circus, Plymouth PL48AA, Devon
United Kingdom

MIROSLAW KWIESIELEWICZ
Faculty of Electrical Engineering
Technical University of Gdansk
G. Narutowicza 11/12, 80-952 Gdansk
Poland

Key words: Recurrent neural networks, system identification, simulation

ABSTRACT

This paper considers a recurrent dynamic neural network for the identification of a highly non-linear dynamic system. The network structure is described and numerical results are presented for the identification of the yaw dynamics of an underwater vehicle. Finally, the paper concludes with comments concerning further developments of the method presented.

INTRODUCTION

In many practical situations the model of a continuous, non-linear, dynamic system to be controlled is unknown or ill-defined due to its complexity, uncertain character or there is a lack of representative data. In such a situation an identification approach is often adopted to identify the model.

A method for the identification of a highly non-linear, continuous, dynamic system is considered in this paper. It is assumed that an identified system can be described by a set of non-linear state differential equations and a recurrent dynamic neural network [1,2] can be employed as the system model.

On one hand a recurrent dynamic neural network can be viewed as a set of non-linear state equations whilst on the other hand as a set of intercoupled dynamic neurons. Consequently such networks can more accurately represent a non-linear dynamic continuous systems behaviour in comparison with discrete techniques and also a neural oriented method can be applied for adjusting system parameters, e.g. a backpropagation procedure.

DYNAMIC NEURAL NETWORK

The dynamic neural network may be described by the following system of coupled differential equations [1,2,3]

$$T_i \frac{dx_i}{dt} = -x_i + \sigma(s_i) + u_i, \quad i = 1,\mathrm{K}\,,n\,. \tag{1}$$

$$s_i = \sum_{j=1}^{n} w_{ij} x_j, \quad i = 1,\mathrm{K}\,,n\,, \tag{2}$$

where n is the number of dynamic neurons, x_i is the state of the ith dynamic neuron, s_i is the total input to the ith unit, T_i is the time constant of unit i, w_{ij} are weights, u_i are inputs to the system and σ is an arbitrary differentiable function, in this case it is taken to be a sigmoidal function

$$\sigma(\xi) = \left(1 - e^{-\xi}\right)^{-1}.$$

TRAINING ALGORITHM

Without any loss of generality it is assumed that the system to be identified is a SISO one. Consequently the energy function to be minimized is

$$E = \frac{1}{2}\int_{t_0}^{t_1} \left(y(t) - d(t)\right)^2 dt\,, \tag{3}$$

where y is the actual and d is the desired trajectory of the network over the time interval $[t_0, t_1]$, and the network of equations (1)-(2) can be developed based on the approaches of Pineda [3] and Pearlmutter [1,2].

Training of the fixed points of the network constitutes a forward pass using equations (1) and (2), followed by the solving of the differential equation set (backward pass)

$$\frac{dz_i}{dt} = \frac{1}{T_i} z_i - e_i - \sum_{j=1}^{n} w_{ij}\sigma'(s_j) z_j, \quad i = 1,\mathrm{K}\,,n\,, \tag{4}$$

with the boundary condition

$$z_i(t_1) = 0\,.$$

where

$$e_i(t) = \frac{\delta E}{\delta x_i(t)}.$$

and finally adjusting the weights and the time constants according to

$$\frac{\partial E}{\partial w_{ij}} = \frac{1}{T_j}\int_{t_0}^{t_1} x_i \sigma'(s_j) z_j\, dt, \quad i, j = 1,\mathrm{K}\,,n \tag{5}$$

and

$$\frac{\partial E}{\partial T_i} = -\frac{1}{T_j}\int_{t_0}^{t_1} z_i \frac{dx_i}{dt}\, dt, \quad i, j = 1,\mathrm{K}\,,n\,. \tag{6}$$

The variables $z_i(t)$ should be understood as ordered derivatives

$$z_i(t) = \frac{\partial^+ E}{\partial x_i(t)}\,. \tag{7}$$

THE SYSTEM TO BE IDENTIFIED

The system under consideration is an underwater vehicle (UV) which is shown in block diagram form in Fig. 1. To describe the yaw dynamics of a UV use is made of MATLAB/Simulink model supplied by the Defence Research Agency (DRA), Sea Systems Sector, Winfrith. The model having been validated against standard DRA non-linear hydrodynamic code using tank test data and an experimentally derived set of hydrodynamic coefficients from the Institute of Oceanographic Science's AUTOSUB vehicle [4,5].

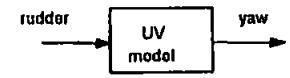


FIG. 1. THE SYSTEM MODEL TO BE IDENTIFIED

ALGORITHM IMPLEMENTATION

The algorithm presented was implemented under the MATLAB environment using the Simulink Toolbox. Training data were collected from the underwater vehicle model.

The training procedure, developed within MATLAB, can be described as follows

1. Set the number of neurons and epochs (Matlab M-file).
2. Initialize weights and time constants randomly (Matlab M-file).
3. Load the desired trajectory from the workspace (Matlab M-file).
4. Perform a forward pass according to the equation sets (1)-(2) and calculate the energy function (3). Save results into the workspace (Simulink block).
5. Perform a backward pass according to the equation set (4) based on data previously stored into the workspace. Save results into the workspace (Simulink block).
6. Calculate the gradient matrix and vector according to the equation sets (5)-(6) basing on data previously stored into the workspace (Matlab M-files and Simulink blocks).
7. Normalize the gradient matrix and vector (Matlab M-file).
8. Adjust weights and time constants (Matlab M-file)
9. If end of the calculations go to 10 else go to 4.
10. Stop.

Some steps of the algorithm are implemented using M-files and some are included as Simulink blocks. Such an approach must be adopted for two reasons. Firstly, it is not possible to implement a matrix differential equation set in the Simulink environment directly. Secondly a forward pass should be integrated forward with some initial conditions whilst the backward pass should be integrated backwards with the boundary conditions as initial conditions.

NUMERICAL RESULTS

In order to obtain the training data, a step change on heading angle was demanded, and the resulting data points were collected from the UV model over a time interval of [0,100] seconds. The training algorithm was then applied to adjust the network weights and time constants. The numerical results for a dynamic neural net (Fig. 2) with 10 neurons are shown in Fig. 3. The final value of the energy function was 0.02.

Scaling factors for the input and output of the dynamic neural net were used.

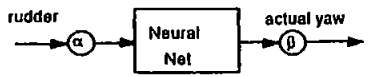


FIG. 2 THE NEURAL NET

During training the algorithm often became trapped in local minima of the error hypersurface. It was noted that the success of the algorithm to converge upon the global minima depended largely on the random initial conditions. The addition of a momentum term did not improve the situation. Consequently it was not possible to obtain an inverse model of the UV.

CONCLUSIONS

It has been shown that it is possible to identify a highly non-linear continuous dynamic system using a dynamic recurrent neural network approach.

However, the numerical training results are not entirely satisfactory. Using the approach presented it is possible to obtain a neural model of the system, however it is very difficult to obtain its inverse model due to wish to thank numerical instabilities during calculations and a large number of local minima on the error hypersurface.

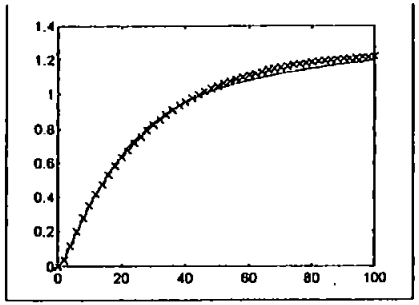


FIG. 3. NUMERICAL RESULTS FOR UV MODEL
- desired trajectory
x actual trajectory

The implementation of the algorithm was limited by certain features of the MATLAB/Simulink software used. In particular it was not possible to perform simulations for matrix state equations in a direct way. For instance, simulation of the equation set (5) can be solved using an M-file with multiple calls to a Simulink block which calculates a weight matrix row or column. To overcome this problem the implementation of all backward pass equation sets in one simulink block is suggested.

As mentioned in the previous section, the solution to the problem mainly depends on the initial conditions. Thus instead of the backward pass a stochastic method or genetic algorithm based method should be applied to adjust the weights and time constants of the network.

Additionally, it should be noted that an inverse model may not exist for the system under consideration.

It is necessary to stress that the algorithm can be implemented using any mathematical library and that the technique proposed is based on a non-linear dynamic network model which can replicate the dynamic behavior of a given dynamic system in comparison with currently applied discrete neural based techniques.

ACKNOWLEDGEMENTS

The authors wish to thank DRA Sea Systems Sector, Winfrith for supplying the underwater vehicle model used in this study.

REFERENCES

[1] B. A. Pearlmutter, "Learning state space trajectories in recurrent neural networks", *Tech. Report.* CMU-CS-88-191, School of Computing Science, Carnegie Mellon University, Pittsburgh, USA, 1988.

[2] B. A. Pearlmutter, "Learning state space trajectories in recurrent neural networks", *Neural Computation* 1, 1989, 263-269.

[3] F. Pineda, "Generalization of backpropagation to recurrent neural networks", *Physical Review Letters* 19 (59) 1987, 2229-2232.

[4] D. Cowling and S.J. Corfield, S, "Control functions for autonomous underwater vehicles on-board command and control systems", IEE *Colloquium on Control and Guidance of Remotely Operated Vehicles*, London, June 1995, 3/1-3/8.

[5] D. Cowling, "Full range autopilot design for an unmanned underwater vehicle", *Proc. 13th IFAC World Congress*, Vol. Q, San Francisco, USA, July 1996, 339-344.

[13] Pokojski J., Wróbel J.: *Some Aspects of Integrated Product and Process Model In Machine Design. Applications of Artificial Intelligence in Structural Engineering.* Proc.of the 3 Workshop of the EG-SEA-AI, Glasgow, UK, August 12-13, 1996, pp.57-60.
[14] Pokojski J., Wróbel J.:*CAD i co dalej*. Przegląd Mechaniczny, 16, 1996, ss.2-3.
[15] Schafer Heiztechnik. *Technische Information – Instalation.* Sanleitung.
[16] Scherer R.: *Integrated product and process model for the design of reinforced concrete structures. Applications of Artificial Intelligence in Structural Engineering.* Proc. of the 1st Workshop of the EG-SEA-AI, Lausanne, Switzerland, March 21-22, 1994, pp 132-145.
[17] Smith I. (Ed.): *Applications of Artificial Intelligence in Structural Engineering.* Proc. of the 1st Workshop of the EG-SEA-AI, Lausanne, Switzerland, March 21-22, 1994.
[18] Tong Ch., Sriram D. (Ed.): *Artificial Intelligence in Engineering Design.* Vol. 1,2,3. Academic Press, 1992.

Streszczenie

W pracy przedstawiono próbę wprowadzenia modułu wspomagającego kreatywność w procesie projektowania. Zostały rozważone dwa podejścia do problemu kreatywności w projektowaniu. Pierwsze oparte jest na zaawansowanych technologiach bazujących na ponownym użyciu wiedzy. Drugie podejście polega na eliminacji rutynowych działań z procesu projektowego. Problem wspomagania kreatywności w projektowaniu został zilustrowany przykładem środowiska wspomagającego projektowanie instalacji centralnego ogrzewania.

TOWARDS CREATIVITY IN DESIGN PROCESS

Summary

The paper shows the idea of application of creativity supporting module in design process. Two approaches to creativity in design process are considered. The first one is based on some sophisticated technologies, the second one on the elimination of routine steps from the design process. As an example the problem of heating system design is mentioned.

Paul J. CRAVEN*
Miroslaw KWIESIELEWICZ**
Robert SUTTON*

DYNAMIC SYSTEM CONTROL USING ANFIS ARCHITECTURE BASED CONTROLLER

1. Introduction

Control system analysis and synthesis are very important problems within the field of control engineering. In particular, the required structure and parameters of a controller should be specified during the design process. During recent years many soft computing techniques, such as fuzzy logic, artificial neural networks and genetic algorithms have been developed and applied to control system analysis and synthesis especially in the area of non-linear systems. Fuzzy logic and artificial neural network based controllers can be used to control non-linear dynamic systems but both methods have some inherent disadvantages. The tuning of a fuzzy logic controller is usually performed heuristically whilst neural network controller behaviour cannot always be forecast. Hence the use of a fuzzy-neural approach for fuzzy controller parameter tuning provides a method by which the linguistic benefits of fuzzy controllers can be fused with the numerical capabilities of neural networks. The Adaptive-Network-Based Fuzzy Inference System (ANFIS) of Jang [4] combines the advantages of both techniques for the neural tuning of fuzzy controllers. Within this paper an ANFIS controller has been implemented for use in the MATLAB environment.

In this paper the parameter tuning process of ANFIS is discussed in application to the control of a highly non-linear dynamic system.

2. Sugeno mode of inference

For simplicity let us recall the first order Sugeno mode of inference [6,7] for a fuzzy controller and assume that the system has two rules (see Fig. 1):

Rule 1: if x is A_1 and y is B_1,
then $f_1 = p_1x + q_1y + r_1$,

Rule 2: if x is A_2 and y is B_2,
then $f_2 = p_2x + q_2y + r_2$,

*University of Plymouth, Institute of Marine Studies, Drake Circus, Plymouth PL4 8AA, Devon,United Kingdom, E-mail:P.J.Craven@Plymouth.ac.uk, R.Sutton-1@Plymouth.ac.uk
**Technical University of Gdańsk, Faculty of Electrical and Control Engineering, G.Narutowicza 11/12, 80-952 Gdańsk, Poland, E-Mail: MKwiat@.ely.pg.gda.pl

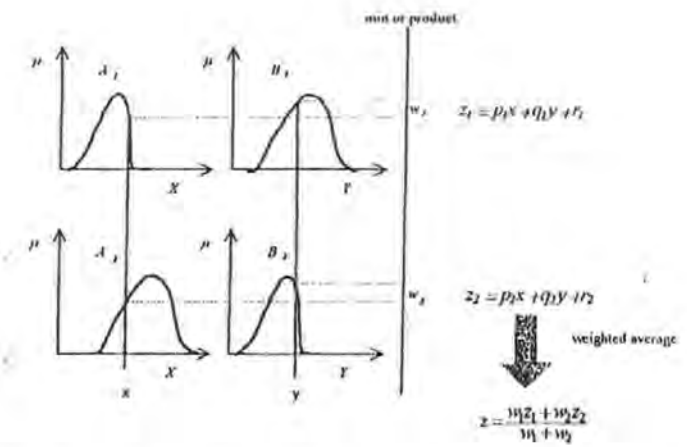


Fig. 1. The two rule Sugeno fuzzy model

3. ANFIS as an adaptive neural net

In the general case ANFIS can be viewed as a adaptive network structure with adaptive and fixed nodes (Fig. 2) [4,5] and, in particular, can be based on a first order Sugeno model (Fig. 3).

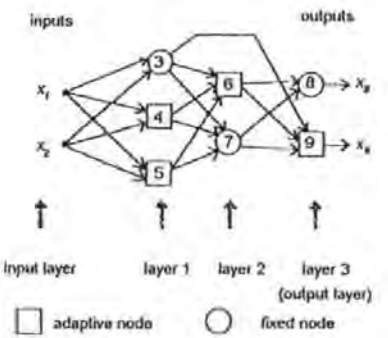


Fig 2. An adaptive network

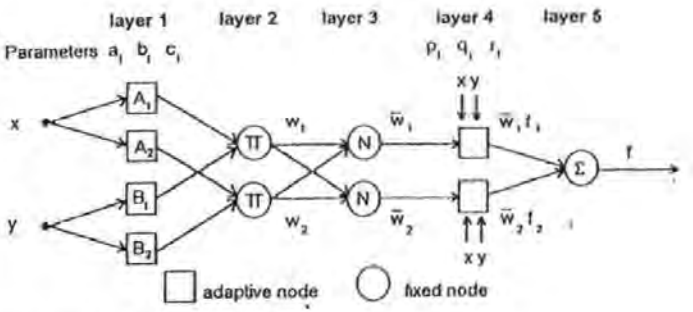


Fig. 3. ANFIS architecture for the Sugeno model

Assuming a fuzzy inference system to have two rules, each individual layer of the ANFIS architecture for the first order Sugeno model can be described as follows:

Layer 1 (adaptive nodes)

$$O_{1,i} = \mu_{A_i}(x), \quad i = 1,2, \text{ or}$$
$$O_{1,i} = \mu_{B_{i-2}}(y), \quad i = 3,4$$

where x or y is the input to the node, $O_{1,i}$ are outputs and A_i (or B_{i-2}) is the fuzzy set associated with this node with a membership function described by:

$$\mu_{A_i}(x) = \frac{1}{1 + \left[\left(\frac{x - c_i}{a_i}\right)^2\right]^{b_i}},$$

where a_i, b_i, c_i are the parameters to be adapted during the training procedure.

Layer 2 (fixed nodes)

$$O_{2,i} = w_i = \mu_{A_i}(x) \times \mu_{B_i}(y), \quad i = 1,2.$$

Layer 3 (fixed nodes)

$$O_{3,i} = \overline{w}_i = \frac{w_i}{w_1 + w_2}, \quad i = 1,2.$$

Layer 4 (adaptive nodes)

$$O_{4,i} = \overline{w}_i f_i = \overline{w}_i \left(p_i x + q_i y + r_i \right),$$

where $\overline{w}_i$ is the output of layer 3 and p_i, q_i, r_i are the parameters to be adjusted.

Layer 5 (single fixed node)

$$O_{5,i} = \sum_i \overline{w}_i f_i = \frac{\sum_i w_i f}{\sum_i w_i}$$

The ANFIS network can be trained using a hybrid procedure [4] which is a combination of the gradient descent and the recursive least squares estimator, [4,5].

4. ANFIS as a controller

The control system structure with ANFIS implemented as a controller is shown on Fig. 4. In order to train the ANFIS controller input/output data pairs are collected over the system trajectory, namely a set of pairs consisting of the error and the control. From a practical point of view there is a problem when attempting to find the desired control action as a function of the system output. In other words it is necessary to specify two sets of input/output data pairs: controller input, system output and controller input, controller output. To overcome this problem several techniques have been applied [4]. In particular the system can be identified using a neural network and the training procedure performed using the control system model with the system replaced by its neural network model. Another method is to obtain an inverse model of the system. In this approach a neural network technique can also be useful. In general the system to be controlled is non-linear and dynamic. In such circumstances the application of a dynamic neural network to the system identification seems quite natural. Alternatively, the ANFIS architecture can be used. In this paper we propose a recurrent dynamic neural network for the system identification problem.

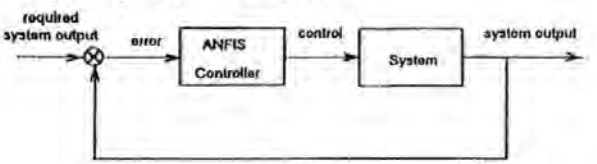


Fig. 4. The control system structure with the ANFIS controller.

5. Recurrent neural network

In order to identify the system to be controlled a dynamic neural network is proposed which can be described by the following system of coupled differential equations [8,9,10]

$$T_i \frac{dx_i}{dt} = -x_i + \sigma(s_i) + u_i, \quad i = 1,\ldots,n, \tag{1}$$

$$s_i = \sum_{j=1}^{n} w_{ij} x_j, \quad i = 1,\ldots,n, \tag{2}$$

where n is the number of dynamic neurons, x_i is the state of the ith dynamic neuron, s_i is the total input to the ith unit, T_i is the time constant of unit i, w_{ij} are weights, u_i are inputs to the system and σ is an arbitrary differentiable function, e.g. a sigmoidal function

$$\sigma(\xi) = \left(1 - e^{-\xi}\right)^{-1} \tag{3}$$

Such a network can easily be implemented in the MATLAB environment. Training of the network (1)-(2) leads to weight and time constant adjustments. There are three main procedures to train this network:

- Forward/backward technique [8]
- Backpropagation through time [11]
- Forward propagation [11]

Because only the forward/backward technique can be implemented directly in the MATLAB environment this one has been chosen for the system identification.

Assuming the energy function

$$E = \frac{1}{2} \int_{t_0}^{t_1} \left(y(t) - d(t) \right)^2 dt, \tag{3}$$

where y is the actual and d is the desired trajectory of the network over the time interval $[t_0, t_1]$, the network of equations (1)-(2) can be developed based on the approaches of Pineda [10] and Pearlmutter [8,9].

Training of the fixed points of the network constitutes a forward pass using equations (1) and (2), followed by the solving of the differential equation set (backward pass)

$$\frac{dz_i}{dt} = \frac{1}{T_i} z_i - e_i - \sum_{j=1}^{n} w_{ij} \sigma'(s_j) z_j, \; z_i(t_1) = 0, \; i = 1,\ldots,n, \tag{4}$$

where

$$e_i(t) = \frac{\delta E}{\delta x_i(t)},$$

and finally adjusting the weights and the time constants according to

$$\frac{\partial E}{\partial w_{ij}} = \frac{1}{T_j}\int_{t_0}^{t_1} e_i \sigma'(s_j) z_j dt, \quad i,j = 1,\ldots,n \tag{5}$$

and

$$\frac{\partial E}{\partial T_i} = -\frac{1}{T_j}\int_{t_0}^{t_1} z_i \frac{dx_i}{dt} dt, \quad i,j = 1,\ldots,n \tag{6}$$

The details of the algorithm implementation within the MATLAB environment can be found in [3].

6. ANFIS controller training

The approach presented has been used to identify the inverse model of an underwater vehicle (UV) [3]. To describe the yaw dynamics of a UV use is made of a MATLAB/Simulink model supplied by the Defence Research Agency (DRA), Sea Systems Sector, Winfrith. [1,2]. The UV model and ANFIS PD controller were implemented as Simulink blocks (see Fig. 5) under the MATLAB environment..

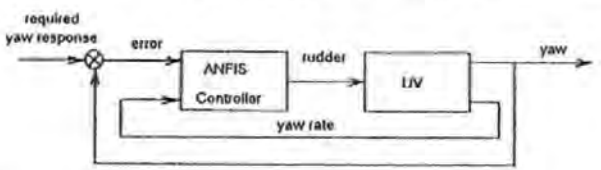


Fig. 5. The UV control with the PD ANFIS controller.

The ANFIS controller had two inputs: yaw error and yaw rate, with 3 generalized bell mf's described on each universe of discourse, and one output with 9 linear mf's functions and hence 9 rules. Training data consisting of a yaw response with a fast rise time were collected. A time interval of [0,20] seconds was selected from this data for the purposes of training the network parameters.

In order to tune the ANFIS controller parameters it was necessary to specify three trajectories over the assumed time interval: yaw error, yaw rate and rudder. Since the yaw signal in this case is a system dynamics response a reference rudder signal should be calculated, i.e. finding the UV inverse model.

The following training procedure was therefore applied

1. System inverse model identification based on the presented recurrent neural net approach.
2. Calculation of the desired control based on the inverse model and desired system output.
3. Training of the ANFIS controller based on the data obtained in the step 2.

A recurrent neural net for the inverse model identification consisted of between 10 and 100 dynamic neurons.

7. Remarks and Comments

Initially it was thought possible to obtain a UV model using the proposed approach (a recurrent neural net with 10 neurons) [3] however it was noted that the success of the algorithm to converge upon the global minima depended largely on the random initial conditions. The addition of a momentum term did not improve this situation. Consequently it was not possible to obtain an inverse model of the UV.

To overcome the aforementioned difficulties the use of a random search technique is proposed in order to develop the dynamic neural network (1)-(2), e.g. genetic algorithms. Recent investigations show that the application of a genetic algorithms to tune the network parameters slightly improves the convergence and gives a possibility of obtaining a inverse UV model. However this procedure is time consuming and in consequence more time is needed for calculations.

As aforementioned in section 4, model identification can be performed by employing the ANFIS architecture, however in the case under consideration it was not possible to obtain either the UV model nor the inverse UV model.

An alternative method for obtaining a UV inverse model may be to calculate desired controller output based on the UV neural model and equation set (4).

References

1. Cowling D., Corfield, S.J,: *Control functions for autonomous underwater vehicles on-board command and control systems*, IEE Colloquium on Control and Guidance of Remotely Operated Vehicles, London, June 1995, 3/1-3/8.
2. Cowling, D.: *Full range autopilot design for an unmanned underwater vehicle*. Proceedings of 13th IFAC World Congress, Vol. Q, San Francisco, USA, July 1996, 339 -344.
3. Craven P.J., Kwiesielewicz M. and Sutton R.: *Non-linear system identification using a recurrent neural network*. Proceedings of 16th IASTED International Conference 'MODELLING, IDENTIFICATION AND CONTROL', February 17 - 19th, Innsbruck, Austria, 1997, pap. No. 258-161.
4. Jang J.-S. R.: ANFIS: *Adaptive-Network-Based Fuzzy Inference System*. IEEE Trans. Systems Man and Cybernetics, Vol. 23., No.3, 1993, 665-685.
5. Jang J.-S. R., Sun C.-T.: *Neuro-Fuzzy Modelling and Control*. Proceedings of the IEEE, Vol. 83, No. 3, March 1995, 341-492.
6. Sugeno M., Kang G.T.,: *Structure identification of fuzzy model*. Fuzzy Sets and Systems, Vol. 28, 1988, 15-33

A NEUROFUZZY AUTOPILOT FOR UNMANNED UNDERWATER VEHICLE CONTROL

Robert Sutton and Paul J Craven
Institute of Marine Studies, University of Plymouth
Drake Circus, Plymouth, PL4 8AA, UK
(E-mail : rsutton@plymouth.ac.uk)

Abstract

This paper describes the application of a neurofuzzy approach in the design of an autopilot for controlling the yaw dynamics of an unmanned underwater vehicle (UUV). The autopilot is designed using an adaptive network-based fuzzy inference system (ANFIS). To describe the yaw dynamic characteristics of a UUV a realistic simulation model is employed. Results are presented which demonstrate the effectiveness of the ANFIS approach. It is concluded that the approach offers a viable alternative method for designing yaw autopilots.

1. Introduction

The dynamic characteristics of unmanned underwater vehicles (UUVs) present a control design problem which linear design methodologies cannot accommodate easily. Fundamentally, UUV dynamics are non-linear in nature and are subject to a variety of disturbances such as varying drag forces, vortex effects and currents. Studies into the development of UUV control strategies have been undertaken using advanced control engineering concepts such as H^{∞}, sliding mode and adaptive theories. These investigations have achieved limited success owing either to the simplification of the problem or the control scheme lacking robustness.

Artificial intelligence approaches are now also being introduced in the design process. Autopilots formulated using fuzzy logic and artificial neural network (ANN) methods have been reported and shown to be endowed with commendable robustness properties. Encouraged by such results, this paper considers the development of a course-keeping autopilot based on the innovative neurofuzzy methodology of Jang [1] known as the adaptive network-based fuzzy inference system (ANFIS).

To describe the yaw dynamics of a UUV use is made of a MATLAB/Simulink model supplied by the Defence Research Agency (DRA), Sea Systems Sector, Winfrith. The model having been validated against standard DRA non-linear hydrodynamic code using tank test data and an experimentally derived set of hydrodynamic coefficients from the Institute of Oceanographic Science's AUTOSUB vehicle.

It should be noted that for this study the upper and lower canards of the UUV were the surfaces used to control its yaw dynamics. Dimensionally, the model represents an underwater vehicle which is 7 m long, 1 m in diameter and has a displacement of 3600 kg.

2. Neurofuzzy Autopilot Design

As mentioned above, the fuzzy controller design used in this study is based on the ANFIS. Functionally, there are almost no constraints on the membership functions of an adaptive network except piecewise differentiability.

Structurally, the only limitation on network configuration is that it should be of feed-forward type. Due to these minimal restrictions, the adaptive network's applications are immediate and immense in various areas.

If it is assumed that the fuzzy inference system under consideration has multiple inputs and one functional output (f) then the fuzzy rule-based algorithm may be represented in the first order Sugeno form as shown below

Rule 1: If x is A_1 and y is B_1 then $f_1 = p_1 x + q_1 y + r_1$
Rule 2: If x is A_2 and y is B_2 then $f_2 = p_2 x + q_2 y + r_2$
: : : :
Rule n: If x is A_n and y is B_n then $f_n = p_n x + q_n y + r_n$

The corresponding ANFIS architecture being shown in Fig 1.

The node functions in the same layer are of the same function family as described by the following:

Layer 1: Every *ith* node in this layer is an adaptive node with a node output defined by:

$$O_{1,i} = \mu_{A_i}(x) \tag{1}$$

where x is the input to the general node and A_i is the fuzzy set associated with this node. In other words, outputs of this layer are the membership values of the premise part. Here the membership functions for A_i can be any appropriate parameterized membership functions. Here A_i is characterized by the generalized bell function:

$$\mu_{A_i}(x) = \frac{1}{1+\left[\left(\frac{x-c_i}{a_i}\right)^2\right]^{b_i}} \tag{2}$$

where $\{a_i, b_i, c_i\}$ is the parameter set. Parameters in this layer are referred to as *premise parameters*.

Layer 2: Every node in this layer is a fixed node labelled Π, which multiplies the incoming signals and outputs the product or T-norm operator result, e.g.

$$O_{2,i} = w_i = \mu_{A_i}(x) \times \mu_{B_i}(y), \quad i = 1, 2 \tag{3}$$

Each node output represents the *firing strength* of a rule. (In fact, any other T- norm operators that perform the fuzzy AND operation can be used as the node function in this layer).

Layer 3 : Every node in this layer is a fixed node labelled N. The *ith* node calculates the ratio of the *ith* rules' firing strength to the sum of all rules' firing strengths:

$$O_{3,i} = \bar{w}_i = \frac{w_i}{w_1 + w_2}, \quad i = 1, 2 \tag{4}$$

For convenience, outputs of this layer are called *normalized firing strengths*.

Layer 4 : Every *ith* node in this layer is an adaptive node with a node function:

$$O_{4,i} = \bar{w}_i f_i = \bar{w}_i (p_i x + q_i y + r_i) \tag{5}$$

where $\bar{w}_i$ is the output of layer 3 and $\{p_i, q_i, r_i\}$ is the parameter set. Parameters in this layer are referred to as *consequent parameters*.

Layer 5 : The single node in this layer is labelled $\sum$, which computes the overall output as the summation of incoming signals:

$$O_{5,1} = \text{overall output} = \sum_i \bar{w}_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \tag{6}$$

Thus an adaptive network that has exactly the same function as a Sugeno fuzzy model may be constructed.

3. The Hybrid Learning Rule

This learning rule was based upon the hybrid learning rule of Jang. The system is simulated using the dynamic model and data is collected across a trajectory. This training data is used to compare the system trajectory with the desired trajectory, and so form the error measure to be used for training of the adaptive network parameters. The error measure chosen was the integral square of heading error over time (ITSE) with a rudder square component added to ensure efficient control effort and is a modified version of that employed by Jang :

$$E = \sum \left[(\psi_d - \psi_a)^2 + \rho(\delta)^2 \right] t \tag{7}$$

The parameters to be altered are the fuzzy parameters of both the premise and consequent layers. The hybrid learning rule employs the backpropagation method to update the fuzzy premise parameters and the recursive least squares method to update the fuzzy consequent parameters.

Writing the premise membership function of equation (2) as:

$$\mu_{ij}(x) = \frac{1}{1+\left[\left(\frac{x-c_{ij}}{a_{ij}}\right)^2\right]^{b_{ij}}} \tag{8}$$

It can be shown that by continuing the backpropagation process through each layer the following learning rules for each individual parameter within layer 1 are determined:

$$\Delta a_{ij} = -\eta \sum_{s=1}^{r} \frac{\partial E_s}{\partial O_{1s}} \cdot \frac{\partial O_{1s}}{\partial O_{1i}} \cdot \left[\frac{2b_{ij} a_{ij}^{(2b_{ij}-1)} \left(x-c_{ij}\right)^{2b_{ij}} a_{ij}^{2b_{ij}} \left(x-c_{ij}\right)^{2b_{ij}}}{\left\{ a_{ij}^{2b_{ij}} \left(x-c_{ij}\right)^{2b_{ij}} + \left(x-c_{ij}\right)^{2b_{ij}} a_{ij}^{2b_{ij}} \right\}^2} \right] \tag{9}$$

$$\Delta b_{ij} = -\eta \sum_{s=1}^{r} \frac{\partial E_s}{\partial O_{1s}} \cdot \frac{\partial O_{1s}}{\partial O_{1i}} \cdot \left[\frac{2a_{ij}^{2b_{ij}} \left(x-c_{ij}\right)^{2b_{ij}} a_{ij}^{2b_{ij}} \left(x-c_{ij}\right)^{2b_{ij}} \ln\left\{ \left| \frac{a_{ij}}{x-c_{ij}} \right| \right\}}{\left\{ a_{ij}^{2b_{ij}} \left(x-c_{ij}\right)^{2b_{ij}} + \left(x-c_{ij}\right)^{2b_{ij}} a_{ij}^{2b_{ij}} \right\}^2} \right] \tag{10}$$

$$\Delta c_{ij} = -\eta \sum_{s=1}^{r} \frac{\partial E_s}{\partial O_{1s}} \cdot \frac{\partial O_{1s}}{\partial O_{1i}} \cdot \left[\frac{-2b_{ij} a_{ij}^{2b_{ij}} \left(c_{ij}-x\right)^{(2b_{ij}-1)} a_{ij}^{2b_{ij}} \left(-\left(x-c_{ij}\right)\right)^{2b_{ij}}}{\left\{ a_{ij}^{2b_{ij}} \left(-\left(x-c_{ij}\right)\right)^{2b_{ij}} + \left(c_{ij}-x\right)^{2b_{ij}} a_{ij}^{2b_{ij}} \right\}^2} \right] \tag{11}$$

4. Results and Discussion

Herein a nine rule Sugeno type fuzzy autopilot has been developed in which the hybrid learning algorithm of Jang was applied to the task of tuning the antecedent and consequent parameters. In order to adapt the fuzzy parameters of the autopilot, the ANFIS autopilot was encoded as an adaptive network architecture. To account for some form of control effort reduction in the resulting fuzzy autopilot, a new cost function (error measure, equation 7) was introduced.

Tuning of the network parameters took place over a series of positive and negative course changes of 40° , at a surge velocity of 7.5 knots. Time intervals of 60 seconds were allowed between consecutive course changing demands to ensure that the UUV transitional and rotational motions had stabilised, and this each course change was applied at similar initial conditions. This method was considered effective and necessary to ensure rule base symmetry.

As mentioned previously, the cost function employed during ANFIS tuning of the fuzzy autopilot incorporated a weighting parameter ρ to allow a for a concession between error and control effort minimization to be achieved. This value was varied during the tuning procedure to obtain varying degrees of compromise. The value used throughout these results gave a control effort weighting of 0.158 within the cost function.

Resulting from this tuning regime the 7.5 knot, the ANFIS autopilot was taken as:

if ψ_e is N and $\dot{\psi}$ is N then $\dot{\delta}$ =

$-1.4619\,\psi_e - 0.8922\,\dot{\psi} + 0.6559$

if ψ_e is N and $\dot{\psi}$ is Z then $\dot{\delta}$ =

$-0.4916\,\psi_e - 0.8833\,\dot{\psi} - 0.0502$

if ψ_e is N and $\dot{\psi}$ is P then $\dot{\delta}$ =

$-0.5074\,\psi_e - 0.8987\,\dot{\psi} - 0.6972$

if ψ_e is Z and $\dot{\psi}$ is N then δ =

$-0.4542\,\psi_e - 0.1090\,\dot{\psi} + 0.7879$

if ψ_e is Z and $\dot{\psi}$ is Z then δ =

$0.0000\,\psi_e + 0.0000\,\dot{\psi} + 0.0000$

if ψ_e is Z and $\dot{\psi}$ is P then δ =

$-0.4542\,\psi_e - 0.1090\,\dot{\psi} - 0.7879$

if ψ_e is P and $\dot{\psi}$ is N then δ =

$-0.5074\,\psi_e - 0.8987\,\dot{\psi} + 0.6972$

if ψ_e is P and $\dot{\psi}$ is Z then δ =

$-0.4916\,\psi_e - 0.8833\,\dot{\psi} + 0.0502$

if ψ_e is P and $\dot{\psi}$ is P then δ =

$-1.4619\,\psi_e - 0.8922\,\dot{\psi} - 0.6559$

The input fuzzy sets for the autopilot can be seen in Fig 2.

In order to test the effectiveness of the design, the ANFIS autopilot was required to react to a series of random course changes as shown in Fig 3. The results show the ability of the autopilot to cope with high heading demands whilst producing reasonable canard responses.

5. Conclusions

The paper demonstrates that a yaw autopilot for a UUV may be designed using the ANFIS approach. In the design of the autopilot a fusion of neural and fuzzy techniques were used. However, the autopilot itself is entirely fuzzy and the network style implementation of the working controller is merely a convenience.

Acknowledgements

The authors wish to thank the Sea Systems Sector, DRA, Winfrith, for supplying the UUV model and their continuing support throughout this study.

Reference

[1] J-S.R. Jang, "ANFIS : adaptive network-based fuzzy inference system", IEEE Trans. Systems, Man and Cybernetics, vol. 23, no. 3, pp. 665-685, 1993.

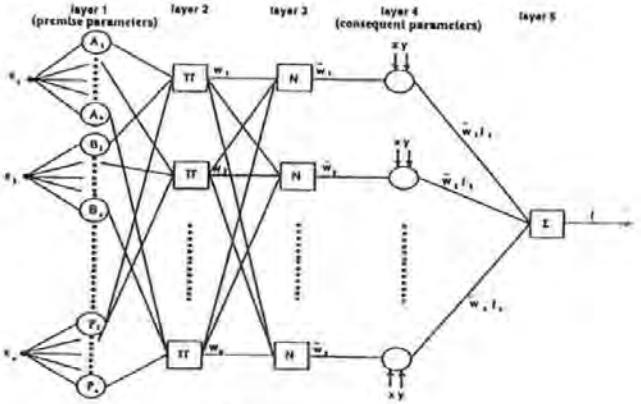


Fig. 1 The ANFIS structure

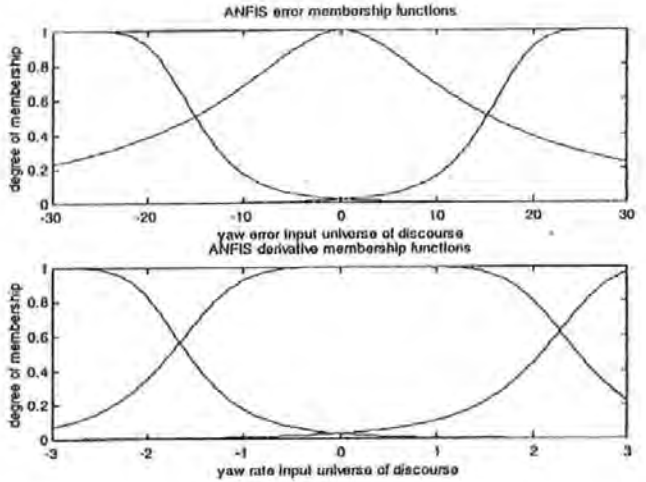


Fig. 2 The input fuzzy sets

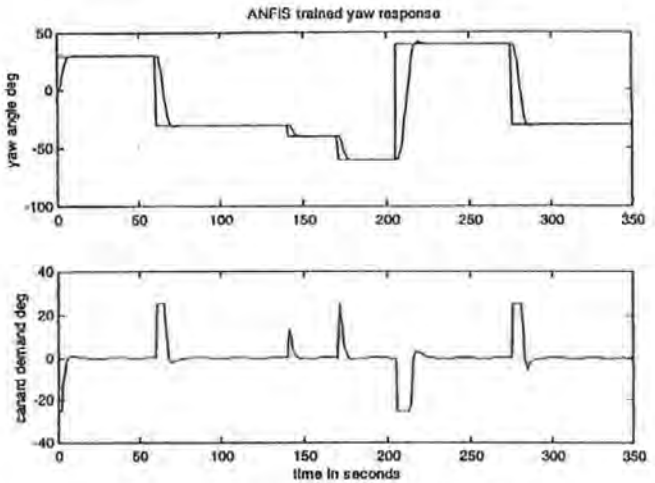


Fig. 3 Yaw and canard responses of the ANFIS trained autopilot

A DESIGN STUDY OF A NEUROFUZZY AUTOPILOT FOR AN AUTONOMOUS UNDERWATER VEHICLE

P J CRAVEN†, R SUTTON†, A TIANO‡, G MAGENES‡

† University of Plymouth, Institute of Marine Studies, Plymouth, UK, pcraven@plymouth.ac.uk, r1sutton@plymouth.ac.uk

‡ Universita di Pavia, Dipartimento di Informatica e Sistemistica, Pavia, Italy, antonio@control1.unipv.it, giovanni@bioing2.unipv.it.

Abstract. This paper describes the application of a neurofuzzy technique in the design of autopilots for controlling the yaw dynamics of an autonomous underwater vehicle (AUV). The autopilots are designed using adaptive network structures which are tuned with simulated annealing and chemotaxis algorithms. To describe the yaw dynamics of an AUV, a realistic simulation model is employed. Results are presented which confirm the approach offers a viable alternative method for designing such autopilots.

Key Words. Neurofuzzy, control, autopilots, autonomous underwater vehicles.

1. INTRODUCTION

Serious consideration is now being given to the design and development of autonomous underwater vehicles (AUVs) in order that they may undertake such tasks as oceanic surveying, pipeline inspection, explosive ordnance disposal and covert surveillance. Such vehicles require to have on board reliable and robust navigation, control and guidance (NCG) systems. An important component in any NCG system is the autopilot subsystem which is responsible for maintaining a vehicle course.

Several advanced control engineering concepts have been employed in the design of underwater vehicle autopilots and have met with varying degrees of success. Artificial intelligence approaches are now also being introduced into the design process. This paper considers the development of course-keeping autopilots based on an adaptive network structure. To optimize the autopilot structure, simulated annealing and chemotaxis algorithms are employed.

2. MODELLING THE AUV DYNAMICS

Figure 1 shows the complete control authority of the AUV model. However, it should be noted that for this study the upper and lower canards are the only surfaces used to control its yaw dynamics. Dimensionally, the model represents an underwater vehicle which is 7 m long, 1 m in diameter and has a displacement of 3600 kg.

The equation of motion describing the dynamic behaviour of the vehicle in the lateral plane is as follows [1] :

$$E\dot{x} = Fx + Gu \qquad (1)$$

where :

$$E = \begin{bmatrix} (m - Y_{\dot{v}}) & -Y_{\dot{r}} & 0 & -(Y_p + mZ_Q) & 0 \\ -N_{\dot{v}} & (I_Z - N_{\dot{r}}) & 0 & -N_p & 0 \\ 0 & 0 & 1 & 0 & 0 \\ -(K_{\dot{v}} + mZ_Q) & -K_{\dot{r}} & 0 & (I_X - K_p) & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$u = \begin{bmatrix} \delta_{UP} \\ \delta_{LO} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$F = \begin{bmatrix} Y_{UV}U & (Y_{UR} - m)U & 0 & Y_{UP}U & 0 \\ N_{UV}U & N_{UR}U & 0 & N_{UP}U & 0 \\ 0 & 1 & 0 & 0 & 0 \\ K_{UV}U & (K_{UR} + mZ_Q)U & 0 & K_{UP}U & -mgBG \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$G = \begin{bmatrix} Y_{UU\delta_W}U^2 & Y_{UU\delta_I}U^2 & 0 & 0 & 1 & 1 \\ N_{UU\delta_W}U^2 & N_{UU\delta_I}U^2 & \ell_\phi & -\ell_\phi & \ell_\psi & -\ell_\psi \\ 0 & 0 & 0 & 0 & 0 & 0 \\ K_{UU\delta_W}U^2 & K_{UU\delta_I}U^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

and the state variables are V, R, ψ, P, ϕ. To implement equation (1) use is made of an AUV MATLAB / Simulink simulation model supplied by the Defence Research Agency (DRA), Sea Systems Sector, Winfrith. The model having been validated against standard DRA non-linear hydrodynamic code using tank test data and an experimentally derived set of hydrodynamic coefficients from the Institute of Oceanographic Science's AUTOSUB vehicle. In addition, the MATLAB / Simulink model structure also takes into account the dynamic behaviour of the canard actuators by describing them as first order lags with appropriate limiters.

As can be seen in equation (1) the roll cross-coupling dynamics are included. However, control of the roll channel is not considered here.

3. ADAPTIVE NETWORK ARCHITECTURE

In order to adapt the parameters of the fuzzy autopilot it is necessary to encode the autopilot in an adaptive network architecture. Functionally, there are almost no constraints on the node functions of an adaptive network except piecewise differentiability. Structurally, the only limitation of network configuration is that it should be of feed-forward type.

Suppose that the rule base contains n fuzzy if-then rules of Takagi and Sugeno form:

Rule1: If x *is* A_1 *and* y *is* B_1 *then* $f_1 = p_1 x + q_1 y + r_1$

Rule2: If x *is* A_2 *and* y *is* B_2 *then* $f_2 = p_2 x + q_2 y + r_2$

Rule i: If x is A_i and y is B_i then $f_i = p_i x + q_i y + r_i$

Rule n: If x is A_n and y is B_n then $f_n = p_n x + q_n y + r_n$

Then the fuzzy adaptive network architecture is illustrated in Fig. 2. The node functions in the same layer are of the same function family as described below.

Layer 1: Every node *i* in this layer is an adaptive node with a node output defined by :

$$Output_i^1 = \mu_{A_i}(x) = \frac{1}{1 + \left[\left(\frac{x - c_i}{a_i}\right)^2\right]^{b_i}} \qquad (2)$$

where A_i is characterised by the generalised bell function. $\{a_i, b_i, c_i\}$ is the parameter set for each membership function which is to be tuned appropriately, x is the input to the node and A_i is the fuzzy set associated with this node. In other words, outputs of this layer are the membership values of the premise part.

Layer 2: Every node in this layer is a fixed node labelled $\prod$, which multiplies the incoming signals and outputs the product or T-norm operator result, e.g.

$$Output_i^2 = w_i = \mu_{A_i}(x) \times \mu_{B_i}(y) \qquad (3)$$

Each node output represents the *firing strength* of a rule.

Layer 3: Every node in this layer is a fixed node labelled *N*. The *ith* node calculates the ratio of the *i*th rule's firing strength to the sum of all rule's firing strengths:

$$Output_i^3 = \bar{w}_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \qquad (4)$$

Layer 4: Every node *i* in this layer is a static node with a node function

$$Output_i^4 = \bar{w}_i f_i = \bar{w}_i (p_i x + q_i y + r_i) \qquad (5)$$

where $\{p_i, q_i, r_i\}$ is the fixed parameter set. Parameters in this layer are referred to as *consequent parameters*.

Layer 5: The single node in this layer is labelled Σ, which computes the overall output as the summation of incoming signals:

$$Output_i^5 = \sum_i \bar{w}_i f_i = \frac{\sum_i w_i f}{\sum_i w_i} \qquad (6)$$

The resulting adaptive network has exactly the same function as a Sugeno fuzzy model.

4. TRAINING ALGORITHMS

Using the network architecture shown in Fig 2, input data are fed forward through the structure in order to generate an error function. The algorithms which are described below were then used to optimize the premises by tuning the parameters (a_i, b_i and c_i) shown in equation (2).

4.1 Simulated Annealing Algorithm

The main problematic aspect of gradient descent based learning algorithms for optimization problems, such as backpropagation, is the tendency for these methods to spend long periods of time in the neighbourhood of poor or sub-optimal local minima on the error hypersurface. A technique which can be employed to overcome this shortcoming is simulated annealing which was first introduced by Kirkpatrick et al [2].

Simulated annealing is a very efficient random search method for global minimization. This method is based on an analogy between the global minimization problem and that of determining the lowest energy state of a physical system.

Kirkpatrick et al [2] adapted an algorithm taken from the statistical mechanics field for converging to one of many possible cooled or low energy states. Energies of this algorithm are described by a Boltzman probability distribution such that the probability of any given energy E is an exponentially decreasing function of E. Thus, if a new matrix of parameters θ, which have been perturbed from an initially assumed solution by a randomly generated amount, lead to an improved performance of the system under consideration, then they are accepted and the process is repeated. However, if this new matrix leads to a worsened performance of the system it may be occasionally accepted with probability $P(\theta)$ such that :

$$P(\theta) = \exp\left[\frac{-E(\theta)}{kT}\right] \qquad (7)$$

where $E(\theta)$ is the energy associated with the state θ, k is the Boltzman's constant and T is a temperature parameter.

For a thermodynamic system, it can be demonstrated both by theoretical arguments and experimental verification that the most effective strategy for obtaining a global minimum energy state requires the temperature to be cooled slowly. Indeed, provided the cooling process is performed sufficiently slow, the system will by-pass locally stable states to reach one which is a global minimum. Thus, in analogous systems, the temperature T is allowed to decay during training according to the following equation :

$$T = \frac{T_o}{1 + an} \qquad (8)$$

where T_o is the initial temperature, a is a constant which governs the decay rate and n is the training epoch.

Hence, simulated annealing may be considered to consist of three distinct phases :

(i) A random search step;
(ii) A minimization stage, and
(iii) A stopping rule.

The random search step is basically the iterative generation of random matrices in a domain $S(\theta_k)$, constituted by neighbouring matrices associated to the current matrix θ_k by :

$$\theta_k = \begin{bmatrix} \theta_k^{11} & \cdots & \theta_k^{1n} \\ \vdots & \ddots & \vdots \\ \theta_k^{m1} & \cdots & \theta_k^{mn} \end{bmatrix} \qquad (9)$$

$\theta_k \in \Re^n$

The minimization stage consists of applying a local minimization routine to some of the sampled matrices. Whilst, the stopping rule terminates the algorithm provided there is sufficient evidence that the global minimum has been detected within the limits of a specified accuracy or some explicit iteration number is reached.

Table 1. Simulated Annealing Algorithm

1. Generate set of initial parameters and simulate system.
2. Make random changes to the parameters and re-simulate the system.
3. If performance improved then retain changes and re-apply.
4. If performance degraded then compute probability of accepting poorer parameters according to equations (7) and (8).
5. Generate random number in the range 0-1 and compare with probability computed at 4. If random number less then accept poorer parameters; otherwise reject.
6. Re-simulate and return to 3 until convergence.

4.2 Chemotaxis Algorithm

The chemotaxis algorithm was inspired by observations of the movement of bacteria in a chemical environment, hence 'chemo' - chemical and 'taxis' - movement [3]. In the presence of an irritant, bacteria would move randomly away in any direction which reduced the irritation, until this direction took them into an area where the irritation began to increase again. A new, random direction would then be chosen and if this again led to less irritation the bacteria would head in the new direction, otherwise another random direction would be tried. In time the bacteria would be located at the global minima, furthest from the source of irritation.

This behaviour may be transformed into a general search algorithm for an optimum set of weights or parameters. The increase/decrease in irritation may be characterized by an increase/decrease in a suitable cost function for the optimization, and by converting this better/worse situation into a reinforcement signal according to:

$r(t) = 1$, better
$r(t) = 0$, worse (10)

the chemotaxis search algorithm may be classed as a reinforcement learning technique. The algorithm is summarized in Table 2.

Table 2 : Chemotaxis Algorithm

1. Simulate the system with an initial set of parameters.
2. Generate some small random changes in the parameters and re-simulate the system.
3. If the system's performance has improved with the new set of parameters, retain the changes and re-apply. This is essentially assuming that the local cost function gradient will continue to be negative in the local area.
4. If the system's performance has worsened, return to step 2.
5. Continue until the system has reached an acceptable level.

Given sufficient training time, the algorithm should converge to a global minimum of the cost function, although given the random nature of the search an extended training period may be necessary.

5. RESULTS AND DISCUSSION

The previous sections have discussed the development of two nine rule Sugeno type fuzzy autopilots. Firstly, the simulated annealing algorithm was applied to the task of tuning the premise parameters of a fuzzy autopilot only, whilst the consequent parameters remained fixed as equally spaced singletons. Secondly, the chemotaxis algorithm was also applied in a similar manner.

Tuning of the fuzzy autopilot premise parameters took place over a series of positive and negative course changes of 40°, at a surge velocity of 7.5 knots. Time intervals of 60 seconds were allowed between consecutive course changing demands to ensure that the UUV translational and rotational motions had stabilised, and thus each course change was applied at similar initial conditions. This method was considered effective and necessary to ensure rule base symmetry.

Again it should be noted that the only parameters for adaptation within the autopilots were the antecedent parameters, the consequent parameters thus being equally spaced upon the output universe of discourse:

if ψ_e is N and $\dot{\psi}$ is N then δ = +25.00
if ψ_e is N and $\dot{\psi}$ is Z then δ = +18.75
if ψ_e is N and $\dot{\psi}$ is P then δ = +12.50
if ψ_e is Z and $\dot{\psi}$ is N then δ = +6.250
if ψ_e is Z and $\dot{\psi}$ is Z then δ = 0.00
if ψ_e is Z and $\dot{\psi}$ is P then δ = -6.250
if ψ_e is P and $\dot{\psi}$ is N then δ = -12.50
if ψ_e is P and $\dot{\psi}$ is Z then δ = -18.75
if ψ_e is P and $\dot{\psi}$ is P then δ = -25.00

By using simplified fuzzy if-then rules of this form the difficulty experienced in assigning appropriate linguistic terms to the nonfuzzy consequents is avoided. Indeed it can be proven that under this form of fuzzy if-then rule the resulting fuzzy inference system has unlimited approximation power to match any non-linear functions arbitrarily well.

Given sufficient training time the resulting input sets for the fuzzy autopilots were as shown in Fig 3.

A qualitative assessment of the autopilot responses was provided by the UUV model's responses to a series of random course changes as shown in Fig 4. Such a track configuration was deemed necessary to assess the generalisation capabilities of the neurally tuned fuzzy autopilots. Indeed, it would seem that the chemotaxis tuned autopilot is more effective at the course changing task than the simulated annealing autopilot with better generalisation over this particular track configuration. The course changing task of 100° illustrates the poor generalisation capability of the simulated annealing tuned fuzzy autopilot, the resulting response being somewhat sluggish.

At 5 knots the effectiveness of the canard control surfaces is significantly reduced due to the diminished hydrodynamic forces acting on them. Intuitively one would expect increased rise times as a consequence of this. At 10 knots the response times of each autopilot were significantly reduced often at the expense of increased overshoots and, in general, more oscillatory behaviour. Both the neurally tuned autopilots faired well with no evidence of steady state errors or unstable behaviour over the whole speed envelope of the UUV thus showing good autopilot robustness to forward speed variation.

On the basis of the qualitative performances the chemotaxis tuned autopilot was deemed the better at the required course changing manoeuvres due to its superior generalisation capability. The response times of the chemotaxis autopilot were faster with less induced overshoot.

Earlier the qualitative performance of each fuzzy autopilot was discussed. This section addresses the performances of each autopilot in a quantitative manner. As a means of measuring the accuracy and rudder activity of a given autopilot, the integral square error (ISE) for the yaw error (ψ_e) and the canard demand (δ_e) are employed:

$$\psi_e = \int_{t_0}^{t_f} (\psi_d - \psi_a)^2 \, dt \tag{11}$$

$$\delta_e = \int_{t_0}^{t_f} (\delta_d - \delta_a)^2 \, dt \tag{12}$$

where ψ_d and δ_d represent desired yaw angle and canard demand, and ψ_a and δ_a represent actual yaw angle and canard demand respectively. To assess the speed of response of the control system, the rise time (T_R) was calculated for each fuzzy autopilot, and the percentage peak overshoot($M_p(t)$) was calculated to assess the oscillatory nature of each response. Here rise time is taken as the time to reach 99 per cent of the desired 40° course change, i.e. 39.6°, and the peak overshoot is measured as a relative percentage of the 40° course change demand.

As the training took place at 7.5 knots, the robustness of each fuzzy autopilot was assessed by testing at speeds of 5, 7.5 and 10 knots. Thus Table 3 contains figures pertaining to these three speeds.

Table 3. Yaw responses over a course changing manoeuvre.

UUV Model	Simulated Annealing autopilot				Chemotaxis autopilot			
	$\psi_e(^{\circ})^2$	$\delta_e(^{\circ})^2$	T_R s	$M_p(t)$%	$\psi_e(^{\circ})^2$	$\delta_e(^{\circ})^2$	T_R s	$M_p(t)$%
5 Knots	117.39	18.71	19.98	0.61	117.74	18.81	16.07	0.02
7.5 Knots	85.46	11.33	15.29	0.01	86.53	10.09	12.78	0.00
10 Knots	68.58	7.79	14.30	0.01	68.32	9.89	11.30	0.00

Testing each autopilot designed at 7.5 knots over the UUV speed envelope provided suitable insight into the robustness of each controller. Table 3 illustrates the yaw response times of each autopilot to a 40° course changing manoeuvre at 5, 7.5 and 10 knots. Additionally figures are supplied for course changing error, canard activity error and peak overshoot.

When operating at 7.5 knots it appears the autopilot designed using the simulated annealing technique was 0.29% more accurate than that of the chemotaxis tuned autopilot. However the minimum rudder demand was exercised by the chemotaxis autopilot, the activity being 10.94% less than the simulated annealing tuned autopilot. This highlights the fact that the simulated annealing algorithm has reduced course changing error at the expense of an increase in canard activity.

Again, at 5 knots the autopilot developed using simulated annealing was 0.30% more accurate than the chemotaxis tuned autopilot at the expense of an increased canard activity.

Finally, the increased effectiveness of the canard control surfaces at the higher operating speed of 10 knots leads to reduced periods of canard saturation over the larger course changing manoeuvres. The simulated annealing autopilot performed well at this

operating speed showing a similar accuracy to that of the chemotaxis autopilot corresponding with a reduction of 21.23% in canard activity.

6. CONCLUSIONS

This paper has discussed the tuning of fuzzy autopilots for yaw control of an UUV. The resulting autopilots remain purely fuzzy as parameter tuning is conducted off-line. From the results presented it may be concluded that the chemotaxis approach provides the better autopilot design solution.

7. ACKNOWLEDGEMENTS

The authors wish to thank the Sea Systems Sector, DRA, Winfrith, for supplying the UUV model and their continuing support throughout this work.

8. REFERENCES

1. Marshfield W. B. : Submarine data set for use in autopilot research, Technical Memorandum, DRA/MAR TM (MTH) 92314, DRA Haslar, April 1992.
2. Kirkpatrick S., Gelatt C., and Vecchi M. : Optimization by simulated annealing, Science, 220, 1983, pp671-680.
3. Koshland D. E. : Bacterial chemotaxis in relation to neurobiology, Annual Review of Neuroscience, Vol 3, 1980, pp43-75.

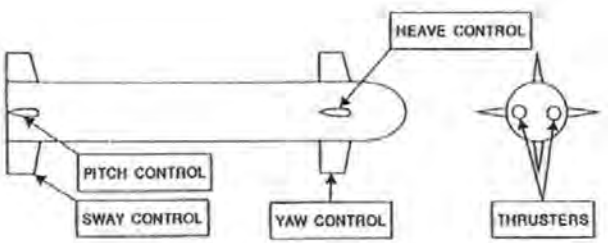


Fig. 1. Control authority of the autonomous underwater vehicle

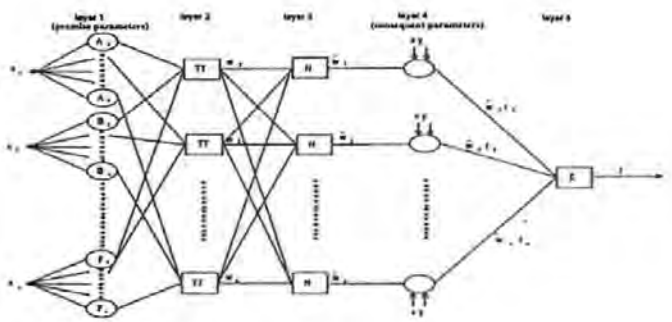


Fig. 2. The adaptive network architecture.

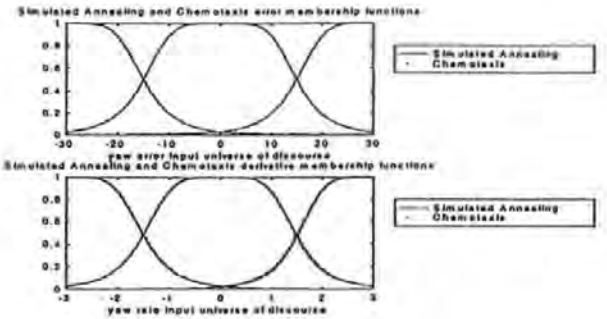


Fig. 3. The final input fuzzy sets.

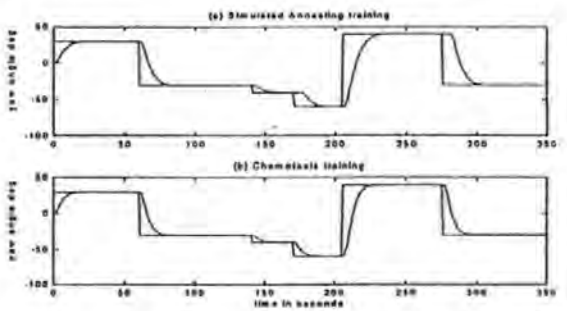


Fig. 4. The autopilot response to a series of random course changes.

Proc. 12th Int. Conf. on Systems Engineering, IMACS SE
Coventry University, UK, 9-11 September 1997

MODELLING THE YAW BEHAVIOUR OF A UUV USING A DYNAMIC NEURAL NETWORK

M Kwiesielewicz

Department of Control Engineering
Technical University of Gdansk, Poland
Tel : Fax : E-mail : mkwies@sparc10.ely.pg.gda.pl

P J Craven and R Sutton

Institute of Marine Studies
University of Plymouth, UK
Tel : 01752 232425 Fax : 01752 232406 E-mail : rlsutton@plymouth.ac.uk
pcraven@plymouth.ac.uk

Keywords : recurrent neural networks, system identification, underwater vehicle.

Abstract

This paper considers a recurrent dynamic neural network for the identification of the non-linear yaw dynamics of an unmanned underwater vehicle. The network structure is described and results are presented to show the validity of the approach.

1. Introduction

The dynamic characteristics of unmanned underwater vehicles (UUVs) present a control system design problem which design methodologies cannot accommodate easily. Fundamentally, UUV dynamics are non-linear in nature and offer a challenging task in the development of suitable algorithms for motion and position control in the six degrees of freedom in which such craft operate. In order to develop a appropriate control strategy, it is necessary to have available a suitable model of the vehicle. Herein a recurrent dynamic neural network is used to identify the non-linear yaw dynamics of a UUV.

2. Dynamics of the UUV

Figure 1 shows the complete control authority of the UUV simulation model. However, it should be noted that for this study the upper and lower canards are the only surfaces used to control its yaw dynamics. Dimensionally, the model represents an underwater vehicle which is 7 m long, 1 m in diameter and has a displacement of 3600 kg.

The equation of motion describing the dynamic behaviour of the vehicle in the lateral plane is as follows [1] :

$$E\dot{x} = Fx + Gu \qquad (1)$$

where :

$$E = \begin{bmatrix} (m - Y_{\dot{v}}) & -Y_{\dot{R}} & 0 & -(Y_P + mZ_G) & 0 \\ -N_{\dot{v}} & (I_Z - N_{\dot{R}}) & 0 & -N_P & 0 \\ 0 & 0 & 1 & 0 & 0 \\ -(K_{\dot{v}} + mZ_G) & -K_{\dot{R}} & 0 & (I_X - K_P) & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$F = \begin{bmatrix} Y_{UV}U & (Y_{UR} - m)U & 0 & Y_{UP}U & 0 \\ N_{UV}U & N_{UR}U & 0 & N_{UP}U & 0 \\ 0 & 1 & 0 & 0 & 0 \\ K_{UV}U & (K_{UR} + mZ_G)U & 0 & K_{UP}U & -mgBG \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$G = \begin{bmatrix} Y_{UU\delta_{up}}U^2 & Y_{UU\delta_{LO}}U^2 & 0 & 0 & 1 & 1 \\ N_{UU\delta_{up}}U^2 & N_{UU\delta_{LO}}U^2 & \ell_{\delta} & -\ell_{\delta} & \ell_{\nu} & -\ell_{\nu} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ K_{UU\delta_{up}}U^2 & K_{UU\delta_{LO}}U^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$u^T = [\delta_{UP} \quad \delta_{LO} \quad 0 \quad 0 \quad 0 \quad 0]^T$$

and the state variables are V, R, Ψ, P, ϕ. To implement equation (1) use is made of the UUV MATLAB/Simulink simulation model supplied by the Defence Evaluation and Research Agency (DERA), Sea Systems Sector, Winfrith. The model having been validated against standard DERA non-linear hydrodynamic code using tank test data and an experimentally derived set of hydrodynamic coefficients from the Institute of Oceanographic Science's AUTOSUB vehicle. In addition, the MATLAB/Simulink model structure also takes into account the dynamic behaviour of the canard actuators by describing them as first order lags with appropriate limiters.

As can be seen in equation (1) the roll cross-coupling dynamics are included. However, control of the roll channel is not considered here.

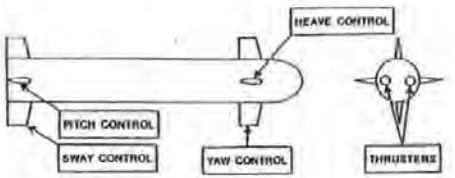


Figure 1. Control authority of the UUV

3. Dynamic Neural Network

Dynamic or recurrent networks differ from static ones in that their structure incorporate feedback. The method adopted to develop the dynamic neural network is based on the approaches of Pineda [2] and Pearlmutter [3], [4] to train the fixed points of a recurrent temporally continuous backpropagation network.

It is assumed the network model is of the form :

$$T_i \frac{dx_i}{dt} = -x_i + \sigma(s_i) + u_i, i = 1, \ldots, n \qquad (2)$$

$$s_i = \sum_{j=1}^{n} w_{ij} x_j, \quad i = 1, \ldots, n \qquad (3)$$

where n is the number of neurons, x_i is the state of the ith dynamic neuron, s_i is the total input to the ith unit, T_i is the time constant of unit i, w_{ij} are weights, σ is an arbitrary differentiable function and u_i are inputs to the system. For this study, the sigmoidal function is employed as follows:

$$\sigma(x) = \frac{1}{(1 - e^{-x})} \qquad (4)$$

3.1 The training algorithm and its implementation

Consideration is given to the minimization of the function:

$$E = \frac{1}{2}\int_{t_0}^{t_1} (y(t) - d(t))^2 dt \qquad (5)$$

where y is the actual trajectory of the network over the time interval $[t_o, t_1]$ and d is the desired trajectory over the same time interval.

Defining:

$$e_i(t) = \frac{\delta E}{\delta y_i(t)}, \quad i = 1, \ldots, m \leq n \qquad (6)$$

where y_i is the system output (a chosen number of system state variables) and:

$$z_i(t) = \frac{\partial^* E}{\partial x_i(t)}, \quad i = 1, \ldots, n \qquad (7)$$

It should be noted that ∂^* is the ordered derivative with variables ordered backpropagating over the time. Thus the differential equations used for backpropagation may be described by:

$$\frac{dz_i}{dt} = \frac{1}{T_i} - e_i - \sum_{j=1}^{n} w_{ij} \sigma'(s_j) z_j \qquad (8)$$

with boundary conditions:

$$z_i(t_1) = 0 \qquad (9)$$

Hence the calculations can be made:

$$\frac{\partial E}{\partial w_{ij}} = \frac{1}{T_i} \int_{t_o}^{t_1} x_i \, \sigma'(s_j) z_j \, dt, \quad i, j = 1, \ldots, n \qquad (10)$$

and

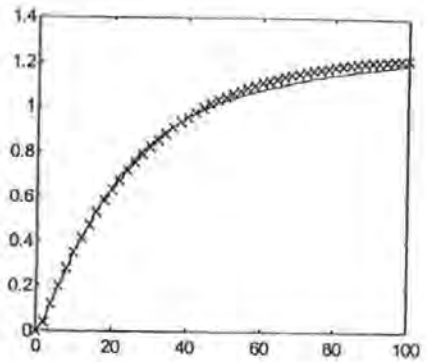
$$\frac{\partial E}{\partial T_i} = \frac{-1}{T_i} \int_{t_o}^{t_1} z_i \frac{dx_i}{dt} \, dt, \quad i = 1, \ldots, n \qquad (11)$$

Thus, the algorithm is implemented initially by the forward calculations of equations (2) and (3). This is followed by the backward calculation of equation (8) and the boundary condition of equation (9). Finally, the partial derivatives shown as equations (10) and (11) are calculated forward to adjust the weights and time constants of the dynamic time continuous network.

4. Results

In order to obtain the necessary training data, a step change in the heading angle of the vehicle was demanded. The resulting data points were collected over a time interval of [0,100] seconds. The training algorithm was then applied to adjust the weights of the network and its time constants. Using a dynamic neural network which contained 10 neurons produced the results shown in Figure 2. The final value of the energy function being 0.02.

During the training period the algorithm became prone to being trapped in local minima of the error hypersurface. It was found that in order for the algorithm to converge to a global minima depended to a large extent on the random initial conditions. Also, the addition of a momentum term did not improve the situation.



desired trajectory

x x x dynamic neural network response

Figure 2. Typical yaw responses obtained using the dynamic neural network

5. Conclusions

It has been shown that it is possible to identify a highly non-linear continuous dynamic system using a dynamic recurrent neural network approach. Comparisons were made between the responses obtained from the DERA UUV model with those from the neural network. The results presented illustrate the viability of the technique to model such non-linear vehicular dynamics and gives encouragement and confidence for its application in actual hardware systems.

Acknowledgement

The authors wish to thank DERA, Sea Systems Sector, Winfrith, for supplying the UUV model used in this study.

References

[1] Marshfield, W.B., Submarine data set for use in autopilot research, Technical Memorandum, DRA/MAR TM(MTH) 92314, DRA Haslar, April, (1992).

[2] Pineda, F., Generalization of backpropagation to recurrent neural networks, Physical Review Letters, 19(59), 2229-2232, (1987).

[3] Pearlmutter, B.A., Learning state space trajectories in recurrent neural networks, Technical Report CMU-CS-88-1991, School of Computing Science, Carnegie Mellon University, Pittsburgh, USA, (1988).

[4] Pearlmutter, B.A., Learning state space trajectories in recurrent neural networks, Neural Computation, 1,263-269, (1989).

Proc. 12th Int. Conf. on Systems Engineering, ICSE'97
Coventry University, UK, 9-11 September 1997

INTELLIGENT COURSE-CHANGING CONTROL OF AN AUTONOMOUS UNDERWATER VEHICLE

P J Craven, R Sutton and RS Burns

University of Plymouth, Drake Circus, Plymouth, PL4 8AA
Tel: +44 (0)1752 232407 Fax: +44 (0)1752 232406 E-mail: pcraven@plymouth.ac.uk

KEY WORDS: Neurofuzzy Control; Autopilots; Autonomous Underwater Vehicles.

ABSTRACT

This paper describes the application of neurofuzzy techniques in the design of autopilots for controlling the yaw dynamics of an autonomous underwater vehicle. Autopilots are designed using an adaptive-network-based fuzzy inference system (ANFIS) and a chemotaxis tuning methodology. To describe the yaw dynamic characteristics of an autonomous underwater vehicle a realistic simulation model is employed. Results are presented which demonstrate the superiority of the ANFIS approach. It is concluded that the approach offers a viable alternative method for designing such autopilots.

1 INTRODUCTION

Encouraged by the prospect of autonomous underwater vehicle (AUV) mission scenarios considerable interest is now being shown in the development and construction of such craft to undertake tasks such as ocean surveying, pipeline inspection, explosive ordnance disposal and covert surveillance. In order that such a craft can possess a suitable level of autonomy it is necessary for it to possess a reliable and robust onboard navigation, guidance and control (NGC) system. A key element of the NGC system is the control subsystem which is responsible for maintaining the vehicle trajectory. Several advanced control engineering concepts including H_∞ [1] and adaptive theories [2] are being employed in the design of autopilots and have met with varying degrees of success.

More recent control system designs have usually incorporated some form of artificial intelligence. Autopilots formulated using fuzzy logic [3] and artificial neural network (ANN) methods [4] have been reported and shown to be endowed with commendable robustness properties. Encouraged by such results, this paper considers the development of a course-keeping autopilot based on the innovative neurofuzzy methodology of Jang [5] known as the adaptive-network-based fuzzy inference system (ANFIS) which was successfully employed to produce a control strategy for the classical inverted pendulum problem.

With the ANFIS approach implementation of the controller design differs in form from the more traditional ANN in that it is not fully connected, and not all the weights or nodal parameters are modifiable. Essentially, the fuzzy rule base is encoded in a parallel fashion so that all the rules are activated simultaneously so as to allow network training algorithms to be applied. As in Jang's original work, here a back propagation algorithm is used to optimize the fuzzy sets of the premises in the ANFIS architecture and a least squares procedure is applied to the linear coefficients in the consequent terms. However, for this study a new cost function is introduced. For performance assessment purposes, comparisons are made with a fuzzy controller whose premises are tuned using a chemotaxis algorithm [6].

2 MODELLING THE DYNAMICS

Figure 1 shows the complete control authority of the AUV model. However, it should be noted that for this study the upper and lower canards are the only surfaces used to control its yaw dynamics. Dimensionally, the model represents an underwater vehicle which is 7 m long, 1 m in diameter and has a displacement of 3600 kg.

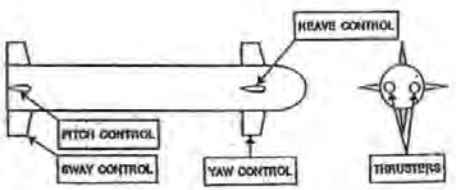


Figure 1. Control Authority of the AUV

The equation of motion describing the dynamic behaviour of the vehicle in the lateral plane is as follows:

$$E\dot{x} = Fx + Gu \qquad (1)$$

where :

$E=$

$$\begin{bmatrix} (m - Y_{\dot{v}}) & -Y_{\dot{r}} & 0 & -(Y_r + mZ_0) & 0 \\ -N_{\dot{v}} & (I_z - N_{\dot{r}}) & 0 & -N_r & 0 \\ 0 & 0 & 1 & 0 & 0 \\ -(K_{\dot{v}} + mZ_0) & -K_{\dot{r}} & 0 & (I_x - K_r) & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$F=$

$$\begin{bmatrix} Y_v U & (Y_{\dot{x}} - m) U & 0 & Y_p U & 0 \\ N_v U & N_{\dot{r}} U & 0 & N_p U & 0 \\ 0 & 1 & 0 & 0 & 0 \\ K_v U & (K_{\dot{x}} + mZ_0) U & 0 & K_p U & -mgG \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$G=$

$$\begin{bmatrix} Y_{\delta_{UP}} U^2 & Y_{\delta_{LO}} U^2 & 0 & 0 & 1 & 1 \\ N_{\delta_{UP}} U^2 & N_{\delta_{LO}} U^2 & \lambda_y & -\lambda_y & \lambda_y & -\lambda_y \\ 0 & 0 & 0 & 0 & 0 & 0 \\ K_{\delta_{UP}} U^2 & K_{\delta_{LO}} U^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$u = \left[\delta_{UP} \delta_{LO} 0000\right]^T$$

and the state variables are V, R, ψ, P, ϕ. To implement equation (1) use is made of an AUV MATLAB / Simulink simulation model supplied by the Defence Evaluation and Research Agency (DERA), Sea Systems Sector, Winfrith, the model having been validated against DERA non-linear hydrodynamic code using tank test data and experimentally derived hydrodynamic coefficients from the Institute of Oceanographic Science's AUTOSUB vehicle. As can be seen in equation (1) the roll cross-coupling dynamics are included. However, control of the roll channel is not considered here.

3 NEUROFUZZY AUTOPILOT DESIGN

If it is assumed that the fuzzy inference system under consideration has multiple inputs and one functional output (f) then the fuzzy rule-based algorithm may be represented in the first order Sugeno form as shown below:

Rule 1 : If x is A_1 and y is B_1 then $f_1 = p_1 x + q_1 y + r_1$
Rule 2 : If x is A_2 and y is B_2 then $f_2 = p_2 x + q_2 y + r_2$
. . .
Rule n : If x is A_n and y is B_n then $f_n = p_n x + q_n y + r_n$

The corresponding ANFIS architecture is shown in Figure 2.

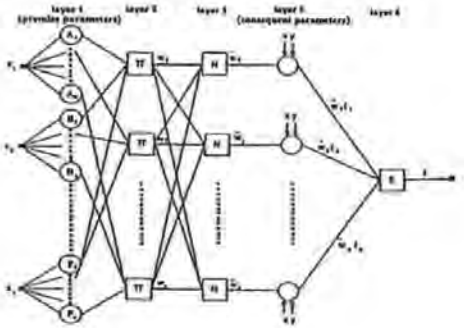


Figure 2. The adaptive network architecture

The node functions in the same layer are of the same function family as described by the following:

Layer 1: Every *ith* node in this layer is an adaptive node with a node output defined by:

$$O_{1,i} = \mu_{A_i}(x) \qquad (2)$$

where x is the input to the general node and A_i is the fuzzy set associated with this node. In other words, outputs of this layer are the membership values of the premise part. Here the membership functions for A_i can be any appropriate parameterized membership functions. Here A_i is characterized by the generalized bell function:

$$\mu_{A_i}(x) = \frac{1}{1 + \left[\left(\frac{x - c_i}{a_i}\right)^2\right]^{b_i}} \quad (3)$$

where (a_i, b_i, c_i) is the parameter set. Parameters in this layer are referred to as *premise parameters*.

Layer 2: Every node in this layer is a fixed node labelled Π, which multiplies the incoming signals and outputs the product or T-norm operator result, e.g.

$$O_{2,i} = w_i = \mu_{A_i}(x) \times \mu_{B_i}(y), \quad i = 1, 2 \quad (4)$$

Each node output represents the *firing strength* of a rule.

Layer 3: Every node in this layer is a fixed node labelled N. The *ith* node calculates the ratio of the *ith* rules' firing strength to the sum of all rules' firing strengths:

$$O_{3,i} = \bar{w}_i = \frac{w_i}{w_1 + w_2}, \quad i = 1, 2 \quad (5)$$

For convenience, outputs of this layer are called *normalized firing strengths*.

Layer 4: Every *ith* node in this layer is an adaptive node with a node function:

$$O_{4,i} = \bar{w}_i f_i = \bar{w}_i (p_i x + q_i y + r_i) \quad (6)$$

where $\bar{w}_i$ is the output of layer 3 and (p_i, q_i, r_i) is the parameter set. Parameters in this layer are referred to as *consequent parameters*.

Layer 5: The single node in this layer is labelled $\sum$, which computes the overall output as the summation of incoming signals:

$$O_{5,i} = \text{overall output} = \sum_i \bar{w}_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \quad (7)$$

Thus an adaptive network that has exactly the same function as a Sugeno fuzzy model may be constructed.

4 TRAINING ALGORITHMS

4.1 The Hybrid Learning Rule

This learning rule the hybrid learning rule of Jang. The system is simulated using the dynamic model and data is collected across a trajectory. This training data is used to compare the system trajectory with the desired trajectory, and so form the error measure to be used for training of the adaptive network parameters. The error measure chosen was the integral square of heading error over time (ITSE):

$$E = \int_0^t (\psi_d - \psi_a)^2 \, dt \quad (8)$$

The parameters to be altered are the fuzzy parameters of both the premise and consequent layers. The hybrid learning rule employs the backpropagation method to update the fuzzy premise parameters and the recursive least squares method to update the fuzzy consequent parameters. Consequently the gradient descent method and the least-squares method have been combined to update the parameters in an adaptive network. Each epoch consists of a forward pass in which inputs are presented and the matrices A and B are calculated and the consequent parameters are updated via the recursive least-squares method. Additionally each epoch consists of a backward pass in which the derivative of the error measure with respect to each nodes output is propagated from the output to the input of the network architecture. At the end of the backward pass the parameters of the premise layer are updated by the gradient descent method.

4.2 Chemotaxis Algorithms

The main disadvantage of backpropagation is the tendency for the search to become trapped in a local minimum on the error surface. The more complex the network the more likely this is to happen as the error surface is increasingly multi-dimensional and therefore irregular, with more local minima into which the partially trained network can fall. An alternative is to use less guided methods to search the parameter space. Such random methods are virtually guaranteed to find a global solution but training times may be somewhat extended as there is little direction in the search.

The chemotaxis algorithm was inspired by observations of the movement of bacteria in a chemical environment, hence 'chemo' - chemical and 'taxis' - movement [6]. In the presence of an irritant, bacteria would move randomly away in any direction which reduced the irritation, until this direction took them into an area where the irritation began to increase again. A new, random direction would then be chosen and if this again led to less irritation the bacteria would head in the new direction, otherwise another random direction would be tried. In time the bacteria would be located at the global minima, furthest from the source of irritation. This behaviour may be transformed into a general search algorithm for an optimum sets of weights or parameters. The increase/decrease in irritation may be characterized by an increase/decrease in a suitable cost function for the optimization, and by converting this better/worse situation into a reinforcement signal according to:

$r(t) = 1$, better
$r(t) = 0$, worse (9)

the chemotaxis search algorithm may be classed as a reinforcement learning technique. The algorithm is summarized in Table I.

1. Simulate the system with an initial set of parameters.
2. Generate some small random changes in the parameters, and re-simulate the system.
3. If the system's performance has improved with the new set of parameters, retain the changes and re-apply. This is essentially assuming that the local cost function gradient will continue to be negative in the local area.
4. If the system performance has worsened, return to step 2.
5. Continue until the system has reached an acceptable level.

Table I. The Chemotaxis Algorithm

Given sufficient training time, the algorithm should converge to a global minimum of the cost function, although given the random nature of the search an extended training period may be necessary.

The structure of the chemotaxis tuned autopilot is similar to that described in section 3 and depicted in Figure 2 for the ANFIS architecture. However, there are some minor dissimilarities. In this case, the nodes in Layer 4 are static and therefore are not modifiable. Also during the tuning process, input data are used to generate an error function to generate which is backpropagated through the adaptive network architecture. The chemotaxis algorithm is then applied to optimize the premise parameters.

5 RESULTS AND DISCUSSION

In the previous sections the development of two nine rule Sugeno type fuzzy autopilots has been discussed. Firstly, the hybrid learning algorithm of Jang was applied to the task of tuning the premise and consequent parameters of a fuzzy autopilot. Secondly, the chemotaxis algorithm was applied to the task of tuning the premise parameters of a fuzzy autopilot only, whilst the consequent parameters remained fixed as equally spaced singletons.

In order to adapt the fuzzy parameters of the autopilots, the ANFIS and chemotaxis autopilots were encoded as adaptive network architectures. Tuning of the network parameters then took place over a series of positive and negative course changes of 40°, at a surge velocity of 7.5 knots. This method was considered effective and necessary to ensure rule base symmetry.

Resulting from this tuning regime the 7.5 knot, the ANFIS autopilot was taken as:

if ψ_e is N and $\dot{\psi}$ is N then $\delta = -1.46\psi_e - 0.89\dot{\psi} + 0.66$

if ψ_e is N and $\dot{\psi}$ is Z then $\delta = -0.49\psi_e - 0.88\dot{\psi} - 0.05$

if ψ_e is N and $\dot{\psi}$ is P then $\delta = -0.51\psi_e - 0.89\dot{\psi} - 0.69$

if ψ_e is Z and $\dot{\psi}$ is N then $\delta = -0.45\psi_e - 0.11\dot{\psi} + 0.79$

if ψ_e is Z and $\dot{\psi}$ is Z then $\delta = 0.00\psi_e + 0.00\dot{\psi} + 0.00$

if ψ_e is Z and $\dot{\psi}$ is P then $\delta = -0.45\psi_e - 0.11\dot{\psi} - 0.79$

if ψ_e is P and $\dot{\psi}$ is N then $\delta = -0.51\psi_e - 0.89\dot{\psi} + 0.70$

if ψ_e is P and $\dot{\psi}$ is Z then $\delta = -0.49\psi_e - 0.88\dot{\psi} + 0.05$

if ψ_e is P and $\dot{\psi}$ is P then $\delta = -1.46\psi_e - 0.89\dot{\psi} - 0.66$

Again, it should be noted that the only parameters for adaptation within the chemotaxis tuned autopilot were the premise parameters and thus the consequent parameters of the chemotaxis autopilot are fixed fuzzy singletons, equally spaced over the output universe of discourse.

On completion of 300 training epochs the resulting input fuzzy sets for the autopilots were as shown in Figure 3. Note the non-symmetrical nature of the tuned input fuzzy sets over the fixed fuzzy sets. This was due to computer truncation errors arising during the training process, and the approximate nature of the initial conditions required to bootstrap the calculation of the sequential least squares estimate for the ANFIS tuned input membership functions.

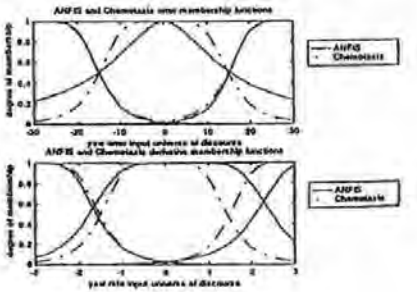


Figure 3. The trained input fuzzy sets

A qualitative assessment of the autopilot responses was provided by the AUV models responses to a series of random course changes as shown in Figure 4.

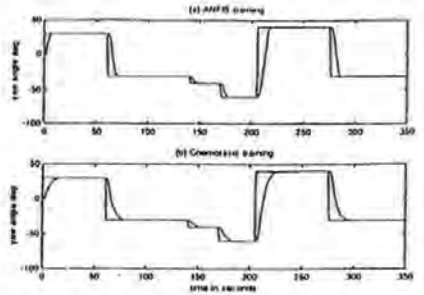


Figure 4. Autopilot responses to random track configuration

It can be seen from these results that each course changing manoeuvre corresponds to a reduction in AUV surge velocity. Although Figure 4 does not provide conclusive evidence of the ANFIS autopilot's superior performance over the chemotaxis autopilot, it is apparent that the hybrid learning algorithm of the ANFIS technique leads to faster, more accurate responses. Indeed it would seem that the ANFIS tuned autopilot is somewhat more effective at the course changing task than the chemotaxis autopilot with improved response times and only minor increases in overshoot as a consequence.

Testing each autopilot designed at 7.5 knots over the AUV speed envelope provided suitable insight into the robustness of each controller. Figures pertaining to the three speeds are displayed in Table 2:

AUV model	ANFIS autopilot				Chemotaxis autopilot			
	ψ_e deg	δ_c deg	T_R secs	$M_p(t)$ %	ψ_e deg	δ_c deg	T_R secs	$M_p(t)$ %
5 Knots	83.18	33.86	9.76	2.86	117.74	18.81	16.07	0.02
7.5 Knots	39.29	20.98	7.79	1.90	86.53	10.09	12.78	0.00
10 Knots	46.02	13.90	7.51	1.22	68.32	9.89	11.10	0.00

Table 2. Yaw responses over a course changing manoeuvre of forty degrees

At 5 knots the effectiveness of the canard control surfaces is significantly reduced due to the diminished hydrodynamic forces acting on them. Intuitively one would expect increased rise times as a consequence of this. During the simulations the chemotaxis tuned autopilot produced some oscillatory responses about the set points of the validation track at the lower operating speed. At 10 knots the response times of each autopilot were significantly reduced often at the expense of increased overshoots and in general more oscillatory behaviour. The ANFIS autopilot faired exceptionally well with no evidence of steady state errors or unstable behaviour over the whole speed envelope of the AUV showing good autopilot robustness to speed parameter variation.

On the basis of the qualitative performances the ANFIS tuned autopilot was deemed the best at the required course changing manoeuvres than the chemotaxis tuned autopilot.

In the previous section the qualitative performance of each fuzzy autopilot was discussed. This section addresses the performances of each autopilot in a quantitative manner. As a means of measuring the accuracy and rudder activity of a given autopilot, the integral square error (ITSE) for the yaw error (ψ_ϵ) and the canard demand (δ_ϵ) are employed:

$$\psi_\epsilon = \int_{t_0}^{t} (\psi_d - \psi_a)^2 \, dt \tag{10}$$

$$\delta_\epsilon = \int_{t_0}^{t} (\delta_d - \delta_a)^2 \, dt \tag{11}$$

where ψ_d and δ_d represent desired yaw angle and canard demand, and ψ_a and δ_a represent actual yaw angle and canard demand respectively. To assess the speed of response of the control system the rise time (T_R) was calculated for each fuzzy autopilot, and the peak overshoot($M_p(t)$) was calculated to assess the oscillatory nature of each response. Here rise time is taken as the time to reach 99 per cent of the desired 40° course change, i.e. 39.6°, and the peak overshoot is measured as a relative percentage of the 40° course change demand.

As the training took place at 7.5 knots, the robustness of each fuzzy autopilot was assessed by testing at training speed +/- 50%, i.e. 5, 7.5 and 10 knots. Summarized in Table 2 are the results for the two fuzzy autopilots at these three speeds.

When operating at 7.5 knots it appears the autopilot designed using the ANFIS technique was considerably more accurate than that of the chemotaxis tuned autopilot. However the minimum rudder demand was exercised by the chemotaxis autopilot The canard demands of the ANFIS tuned autopilot increased over the training period but remained well within the limits of acceptability. Indeed the canard demands produced by ANFIS training were much sharper than those of chemotaxis tuning. As expected, the yaw responses of the autopilots were noticeably more sluggish at the 5 knot training speed. this resulted in increased periods of canard effort over the validation track configuration, but both autopilots coped adequately with the course-changing task. Finally, the increased effectiveness of the canard control surfaces at the higher operating speed of 10 knots lead to reduced periods of canard saturation over the larger course changing manoeuvres. Again the ANFIS autopilot was deemed the best with the most impressive rise times and quite acceptable overshoots.

6 CONCLUSIONS

The work described in this paper demonstrates that course-changing autopilots for AUVs may be designed using Jang's ANFIS approach. It is important to note that in this study, the design of the autopilot is the result of a fusion of neural and fuzzy techniques. However, a distinction exists in that the autopilot itself is entirely fuzzy and the network style implementation of the working controller is merely a convenience.

ACKNOWLEDGEMENTS

The authors wish to thank the Sea Systems Sector, DERA, Winfrith, for supplying the AUV model and for many helpful discussions concerning its implementation.

REFERENCES

[1] D. Cowling, Full range autopilot design for an unmanned underwater vehicle, Proceedings of 1996 IFAC 13th Triennial World Congress, San Francisco, USA, Vol. Q, pp139-144, 30 June-5 July 1996.

[2] T I Fossen and O-E Fjellstad, Robust adaptive control of underwater vehicles, Proceedings of 3rd IFAC Workshop on Control Applications in Marine Systems, Trondheim, Norway, pp.66-74, 10-12 May, 1995.

[3] S M Smith, G J S Rae, D T Anderson and A M Shien, Fuzzy logic control of an autonomous underwater vehicle. Proceedings of 1st IFAC International Workshop on Intelligent Autonomous Vehicles, Southampton, UK, pp.318-23, 18-21 April 1993.

[4] J Yuh, A neural network controller for underwater robotic vehicles, IEEE Journal of Oceanic Engineering, Vol. 15, No.3, pp.161-166, 1990.

[5] J S R Jang, ANFIS : adaptive network-based fuzzy inference system, IEEE Transactions on Systems, Man and Cybernetics, Vol. 23, No 3, pp.665-685, 1993.

[6] D E Koshland, Bacterial chemotaxis in relation to neurobiology, Annual Review of Neuroscience, Vol. 3, pp.45-75, 1980.

A NEUROFUZZY TECHNIQUE APPLIED TO UUV AUTOPILOT DESIGN

P J CRAVEN, R SUTTON, Y-M DAI

University of Plymouth, Institute of Marine Studies, Plymouth, UK
pcraven@plymouth.ac.uk, r1sutton@plymouth.ac.uk, ydai@plymouth.ac.uk

Abstract: This paper is concerned with the application of a neurofuzzy technique in the design of an autopilot for controlling the yaw dynamics of an unmanned underwater vehicle. A fuzzy controller is developed that has been optimized using a simulated annealing algorithm. The algorithm is employed to tune the fuzzy sets in the premise portions of the fuzzy conditional statements which collectively together constitute the fuzzy controller. Results are presented that demonstrate the effectiveness of the tuning algorithm in the optimization of a fuzzy autopilot and comparisons are made with a PD autopilot. It is concluded that the approach adopted offers a viable alternative method for autopilot design.

Key Words: Autopilot design, unmanned underwater vehicle, fuzzy controller, neurofuzzy tuning, yaw control.

1. INTRODUCTION

The dynamic characteristics of unmanned underwater vehicles (UUVs) present a control system design problem which classical linear design methodologies cannot accommodate easily. Fundamentally, UUV dynamics are non-linear in nature and are subject to a variety of disturbances such as varying drag forces and currents. Therefore they offer a challenging task in the development of suitable algorithms for motion and position control in the six degrees of freedom in which they operate, and are required to be robust in terms of disturbance rejection and varying vehicle speeds and dynamics. The term "unmanned underwater vehicle" as used here is a generic expression to describe both an autonomous underwater vehicle (AUV) and a remotely operated vehicle (ROV). An AUV is a marine craft which fulfils a mission or task without being constantly monitored and supervised by a human operator, whilst an ROV is a marine vessel that requires instructions from an operator via a tethered cable or an acoustic link.

In this paper, an effective method for tuning fuzzy membership function parameters is adopted to develop an autopilot for the yaw control of a UUV using a simulated annealing algorithm.

2. MODELLING THE AUV DYNAMICS

Dimensionally, the model represents an underwater vehicle which is 7 m long, 1 m in diameter and has a displacement of 3600 kg. It is assumed that the control authority for the yaw the dynamics is derived via the upper and lower canard surfaces of the vehicle model.

The equation of motion describing the dynamic behaviour of the vehicle in the lateral plane is as follows:

$$E\dot{x} = Fx + Gu \tag{1}$$

where:

$$E = \begin{bmatrix} (m - Y_{\dot{v}}) & -Y_{\dot{r}} & 0 & -(Y_r + mZ_0) & 0 \\ -N_{\dot{v}} & (I_z - N_{\dot{r}}) & 0 & -N_r & 0 \\ 0 & 0 & 1 & 0 & 0 \\ -(K_{\dot{v}} + mZ_0) & -K_{\dot{r}} & 0 & (I_x - K_r) & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$u = [\,\delta_{UP}\ \delta_{LO}\ 0\ 0\ 0\,]^T$$

$$F = \begin{bmatrix} Y_{uv}U & (Y_{ur} - m)U & 0 & Y_{up}U & 0 \\ N_{uv}U & N_{ur}U & 0 & N_{up}U & 0 \\ 0 & 1 & 0 & 0 & 0 \\ K_{uv}U & (K_{ur} + mZ_0)U & 0 & K_{up}U & -mgBG \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$G = \begin{bmatrix} Y_{uu\delta_{UP}}U^2 & Y_{uu\delta_{LO}}U^2 & 0 & 0 & 1 & 1 \\ N_{uu\delta_{UP}}U^2 & N_{uu\delta_{LO}}U^2 & \lambda_y & -\lambda_y & \lambda_y & -\lambda_y \\ 0 & 0 & 0 & 0 & 0 & 0 \\ K_{uu\delta_{UP}}U^2 & K_{uu\delta_{LO}}U^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

and the state variables are V, R, ψ, P, ϕ. To implement equation (1) use is made of an UUV MATLAB / Simulink simulation model supplied by the Defence Evaluation and Research Agency (DERA), Sea Systems Sector, Winfrith. The model having been validated against standard DERA non-linear hydrodynamic code using tank test data and an experimentally derived set of hydrodynamic coefficients from the Institute of Oceanographic Science's AUTOSUB vehicle. In addition, the MATLAB / Simulink model structure also takes into account the dynamic behaviour of the canard actuators by describing them as first order lags with appropriate limiters.

As can be seen in equation (1) the roll cross-coupling dynamics are included. However, control of the roll channel dynamics is not considered here.

3. ADAPTIVE NETWORK ARCHITECTURE

In order to adapt the parameters of the fuzzy autopilot, the autopilot can be encoded as an adaptive neural network architecture. Suppose that the rule base contains *n* fuzzy if-then rules of Takagi and Sugeno form:

*Rule*1: *If* x *is* A_1 *and* y *is* B_1 *then*
$f_1 = p_1 x + q_1 y + r_1$

Rule$i$: *If* x *is* A_i *and* y *is* B_i *then*
$f_i = p_i x + q_i y + r_i$

Rule$n$: *If* x *is* A_n *and* y *is* B_n *then*
$f_n = p_n x + q_n y + r_n$

Then the fuzzy adaptive network architecture is illustrated in Fig. 1. The node functions in the same layer are of the same function family as described below.

Layer 1: Every node i in this layer is an adaptive node with a node output defined by :

$$Output_i^1 = \mu_{A_i}(x) = \frac{1}{1 + \left[\left(\frac{x - c_i}{a_i}\right)^2\right]^{b_i}} \tag{2}$$

where A_i is characterized by the generalised bell function. $\{a_i, b_i, c_i\}$ is the parameter set for each membership function which is to be tuned appropriately, x is the input to the node and A_i is the fuzzy set associated with this node. In other words, outputs of this layer are the membership values of the premise part.

Layer 2: Every node in this layer is a fixed node labelled Π , which multiplies the incoming signals and outputs the product or T-norm operator result, e.g.

$$Output_i^2 = w_i = \mu_{A_i}(x) \times \mu_{B_i}(y) \tag{3}$$

Each node output represents the *firing strength* of a rule.

Layer 3: Every node in this layer is a fixed node labelled N. The i*th* node calculates the ratio of the ith rule's firing strength to the sum of all rule's firing strengths:

$$Output_i^3 = \bar{w}_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \tag{4}$$

Layer 4: Every node i in this layer is a static node with a node function

$$Output_i^4 = \bar{w}_i f_i = \bar{w}_i (p_i x + q_i y + r_i) \tag{5}$$

where $\{a_i, b_i, c_i\}$ is the fixed parameter set. Parameters in this layer are referred to as *consequent parameters* and remain static during the simulated annealing tuning process.

Layer 5: The single node in this layer is labelled Σ, which computes the overall output as the summation of incoming signals:

$$Output_i^5 = \sum_i \bar{w}_i f_i = \frac{\sum_i w_i f}{\sum_i w_i} \tag{6}$$

The resulting adaptive network has exactly the same function as a Sugeno fuzzy model.

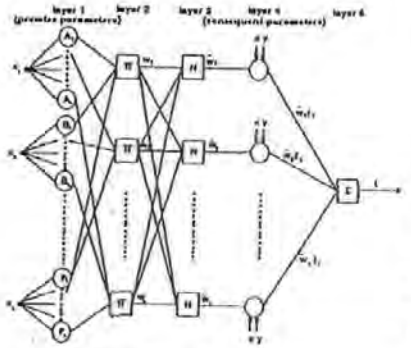


Fig. 1. The adaptive network architecture.

4. TRAINING ALGORITHMS

Using the network architecture shown in Fig. 1, input data are fed forward through the structure in order to generate an error function. The algorithm described below was then used to optimize the premises by tuning the parameters (a_i, b_i, c_i) shown in equation (2).

4.1 Simulated Annealing Algorithm

The most problematic aspect of gradient descent based learning algorithms such as the backpropagation algorithm, is the tendency for these methods to become trapped in the neighbourhood of poor or sub-optimal local minima on the error hypersurface. A technique which can be employed to overcome this shortcoming is simulated annealing.

Simulated annealing (Kirkpatrick, *et al.*, 1983) is a very efficient random search method for global minimization. This method is based on an analogy between the global minimization problem and that of determining the lowest energy state of a physical system.

The simulated annealing algorithm is taken from the statistical mechanics field for converging to one of many possible cooled or low energy states. Energies of this algorithm are described by a Boltzman probability distribution such that the probability of any given energy E is an exponentially decreasing function of E. Thus, if a new matrix of parameters θ, which have been perturbed from an initially assumed solution by a randomly generated amount, lead to an improved performance of the system under consideration, then they are accepted and the process is repeated. However, if this new matrix leads to a worsened performance of the system it may be occasionally accepted with probability P(0) such that

$$P(\theta) = \exp\left[\frac{-E(\theta)}{kT}\right] \qquad (7)$$

where E(0) is the energy associated with the state 0, k is the Boltzman's constant and T is a temperature parameter.

For a thermodynamic system, it can be demonstrated both by theoretical arguments and experimental verification that the most effective strategy for obtaining a global minimum energy state requires the temperature to be cooled infinitely slowly. Indeed, provided the cooling process is performed sufficiently slowly, then the system will by-pass locally stable states to reach one which is a global minimum. Thus, in analogous systems, the temperature T is allowed to decay during training according to the following equation :

$$T = \frac{T_o}{1 + an} \qquad (8)$$

where T_o is the initial temperature, a is a constant which governs the decay rate and n is the training epoch.

Hence, simulated annealing may be considered to consist of three distinct phases :

(i) A random search step;
(ii) A minimization stage, and
(iii) A stopping rule.

The random search step is basically the iterative generation of random matrices in a domain $S(\theta_k)$, constituted by neighbouring matrices associated to the current matrix θ_k by :

$$\theta_k = \begin{bmatrix} \theta_k^{11} & \cdots & \theta_k^{1n} \\ \vdots & \ddots & \vdots \\ \theta_k^{m1} & \cdots & \theta_k^{mn} \end{bmatrix} \qquad (9)$$

$$\theta_k \in \Re^n$$

The minimization stage consists of applying a local minimization routine to some of the sampled matrices. Finally, the stopping rule terminates the algorithm provided there is sufficient evidence that the global minimum has been detected within the limits of a specified accuracy or some explicit iteration number is reached.

Table 1. Simulated Annealing Algorithm

1. Generate set of initial parameters and simulate system.
2. Make random changes to the parameters and re-simulate the system.
3. If performance improved then retain changes and re-apply.
4. If performance degraded then compute probability of accepting poorer parameters according to equations (7) and (8).
5. Generate random number in the range 0-1 and compare with probability computed at 4. If random number less then accept poorer parameters; otherwise reject.
6. Re-simulate and return to 3 until convergence.

Given sufficient training time, the algorithm should converge to the global minimum of the cost function, although the random nature of the search may incur an extended training period. The simulated annealing algorithm is summarized in Table 1.

5. RESULTS AND DISCUSSION

The previous sections have discussed the development of a neurofuzzy autopilot for an UUV. The simulated annealing algorithm was applied to the task of tuning only the premise parameters of a fuzzy autopilot, whilst the consequent parameters remained fixed as equally spaced singletons:

if ψ_e is N and $\dot{\psi}$ is N then δ = + 25.00
if ψ_e is N and $\dot{\psi}$ is Z then δ = + 18.75
if ψ_e is N and $\dot{\psi}$ is P then δ = + 12.50
if ψ_e is Z and $\dot{\psi}$ is N then δ = + 6.25
if ψ_e is Z and $\dot{\psi}$ is Z then δ = 0.00
if ψ_e is Z and $\dot{\psi}$ is P then δ = - 6.25
if ψ_e is P and $\dot{\psi}$ is N then δ = -12.50
if ψ_e is P and $\dot{\psi}$ is Z then δ = -18.75
if ψ_e is P and $\dot{\psi}$ is P then δ = - 25.00

By using simplified fuzzy if-then rules of this form the difficulty experienced in assigning appropriate linguistic terms to the non-fuzzy consequents is avoided.

Tuning of the fuzzy premise parameters took place over a series of positive and negative course changes of 40°, at a surge velocity of 7.5 knots. Time intervals of 60 seconds were allowed between consecutive course changing demands to ensure that the UUV translational and rotational motions had stabilised, and thus each course change was applied at similar initial conditions. This method was considered effective and necessary to ensure rule base symmetry.

A qualitative assessment of the autopilot responses was provided by the UUV model's responses to a series of random course changes as shown in Fig 2. Such a track configuration was deemed necessary to assess the generalization capabilities of the neurally tuned fuzzy autopilot. Indeed, it is shown that the developed neurally tuned autopilot is more effective at the course changing task than a classical PD autopilot with a transfer function:

$$\frac{0.007(1+0.643s)}{1+0.111s} \qquad (11)$$

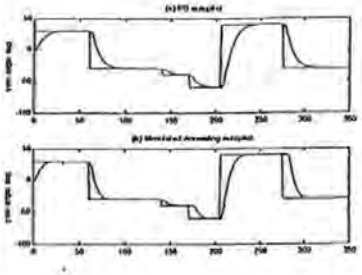


Fig. 2. Autopilot responses to a random track configuration.

The simulated annealing autopilot displays a more accurate response over this particular track configuration, the course changing task of 100° illustrating the sluggish nature of the traditional PD autopilot.

At 5 knots the effectiveness of the canard control surfaces is significantly reduced due to the diminished hydrodynamic forces acting on them. Intuitively one would expect increased rise times as a consequence of this. At 10 knots the response times of each autopilot were significantly reduced often at the expense of increased overshoots and, in general, more oscillatory behaviour. The neurally tuned autopilot faired well with no evidence of steady state errors or unstable behaviour over the whole speed envelope of the UUV thus showing good autopilot robustness to forward speed variation, however the PD autopilot was seen to be somewhat oscillatory at 5 knots showing a lack of autopilot robustness.

On the basis of the qualitative performances the simulated annealing tuned autopilot was deemed the better at the required course changing manoeuvres due to its robust performance at varying forward speeds.

This section addresses the performances of each autopilot in a quantitative manner. As a means of measuring the accuracy and rudder activity of a given autopilot, the integral square error (ISE) for the yaw error (ψ_e) and the canard demand (δ_c) are employed:

$$\psi_e = \int_{t_0}^{t_f} (\psi_d - \psi_a)^2 \, dt \qquad (12)$$

$$\delta_e = \int_{t_0}^{t_f} (\delta_d - \delta_a)^2 \, dt \qquad (13)$$

where ψ_d and δ_d represent desired yaw angle and canard demand, and ψ_a and δ_a represent actual yaw angle and canard demand respectively. To assess the speed of response of the control system, the rise time (T_R) was calculated for each autopilot, and the percentage peak overshoot($M_p(t)$) was calculated to assess the oscillatory nature of each response. Here rise time is taken as the time to reach 99 per cent of the desired 40° course change, i.e.39.6°, and the peak overshoot is measured as a relative percentage of the 40° course change demand.

As the training took place at 7.5 knots, the robustness of each autopilot was assessed by testing at speeds of 5, 7.5 and 10 knots. Thus Table 2 contains figures pertaining to these three speeds.

Table 2. Yaw responses over a course changing manoeuvre of forty degrees

UUV Model	Proportional + Derivative autopilot				Simulated Annealing autopilot			
	$\psi_e(^\circ)^2$	$\delta_e(^\circ)^2$	$T_R s$	$M_p(t)\%$	$\psi_e(^\circ)^2$	$\delta_e(^\circ)^2$	$T_R s$	$M_p(t)\%$
5 Knots	124.75	17.28	20.77	1.14	117.39	18.71	19.98	0.61
7.5 Knots	87.64	11.38	15.16	0.73	85.46	11.33	15.29	0.01
10 Knots	70.51	8.66	14.90	0.42	68.58	7.79	14.30	0.01

Testing each autopilot designed at 7.5 knots over the UUV speed envelope provided insight into the robustness of each controller. Table 2 illustrates the yaw response times of each autopilot to a 40° course changing manoeuvre at 5, 7.5 and 10 knots. Additionally figures are supplied for course changing error, canard activity error and peak overshoot.

When operating at 7.5 knots it appears the autopilot designed using the simulated annealing technique was 2.49% more accurate than the traditional PD autopilot. This highlights the fact that the simulated annealing autopilot has a reduced course changing error over the given course changing configuration.

Again, at 5 knots the autopilot developed using simulated annealing was 5.89% more accurate than the PD autopilot.

Finally, the increased effectiveness of the canard control surfaces at the higher operating speed of 10 knots leads to reduced periods of canard saturation over the larger course changing manoeuvres. The simulated annealing autopilot also performed well at this operating speed showing a superior accuracy of 2.74% and better robustness than that of the PD autopilot.

6. CONCLUSIONS

This paper has discussed the tuning of a fuzzy autopilot for yaw control of an UUV. The resulting autopilot remains purely fuzzy as parameter tuning is conducted off-line. From the results presented it may be concluded that the simulated annealing approach provides a viable autopilot design solution.

7. ACKNOWLEDGEMENTS

The authors wish to thank the Sea Systems Sector, DERA, Winfrith, for supplying the UUV model and their continuing support throughout this work.

8. REFERENCES

Kirkpatrick S., Gelatt C. and Vecchi M (1983). Optimization by simulated annealing. In: *Science*, pp671-680.

IDENTIFICATION METHODS APPLIED TO AN UNMANNED UNDERWATER VEHICLE

A. Tiano [1,2], G. Magenes [1], R. Sutton [2], P.Craven [3]

(1) DIS, University of Pavia, Via Ferrata 1, 27100 Pavia, Italy
Tel:+39 382 505361 - Fax: +39 382 505 373 - email: antonio@control1.unipv.it
(2) Marine Technology Division, Institute of Marine Studies, University of Plymouth, U.K.
Drake Circus, Plymouth, Devon PL4 8AA, U.K.
(3) IAN, National Research Council, Via De Marini 6, Genova, Italy

Abstract: This paper deals with the application of identification methods to the determination of the dynamical behaviour of an UUV (Unmanned Underwater Vehicle). After a concise introduction to the longitudinal equations of the motion, which describe the heave and pitch responses to the action of the control surface deflections and of the thrusters, identification is formulated for a linearized UUV model. The related minimization problem is approached and solved by means of two different random-search methods, respectively based on simulated annealing and on genetic algorithms. The numerical features of such identification methods are discussed and some preliminary promising results are presented, which are obtained by simulation experiments.

Keywords: Identification algorithms, global optimization, genetic algorithms.

1. Introduction

An increasing interest has been devoted in the recent years to the experimental determination of the dynamical behaviour the of naval vehicles by means of system identification methods. Such methods, unlike traditional naval architecture methods, are potentially capable to drastically reduce experiment time and expenses of both towing tank and at sea trials, because a multitude of parameters can be determined from a few dedicated tests.

The mathematical models obtained by means of system identification methods can be, furthermore, directly used for control system design purposes.
The application of system identification techniques to naval vehicles is generally concerned with the estimation of a high number of parameters or hydrodynamic derivatives which are to be estimated (Abkowitz, 1980). In the presence of high dimensional identification problems, however, commonly used identification methods (Ljung, 1987), suffer from a number of serious numerical drawbacks, which may prevent from obtaining an optimal solution to the identification problem. Most of such inconveniences are related to the increasing impossibility of Gauss-Newton iterative minimization algorithms to converge to the global minimum as the unknown parameter vector dimension increases.
It seems, therefore, that minimization methods which do not require derivatives should generally be preferable for multivariable identification problems which are characterized by a high number of parameters to be estimated. Global minimization methods, which are based on a direct cost function evaluation seem to be promising enough, as applied to the identification of the coupled surge-sway-yaw roll dynamics of a surface ship (Tiano and Blanke, 1997) or to the yaw dynamics of an open frame ROV (Caccia et Al., 1997).
This paper is concerned with the application of two global minimization based identification methods to the determination of an UUV (Unmanned Underwater Vehicle) longitudinal dynamics. Since it is foreseen to use the identified model for control system design, a linearized model is herewith identified, according to which it is expected that a diving autopilot should be at a subsequent stage designed..
After presenting in section 2 a concise description of the mathematical model of the vehicle, the identification problem is formulated in section 3. Two parameter estimation methods, based on global optimization approach are discussed: one implements a Simulated Annealing (SA) algorithm, (Hajek, 1985), (Collins et al., 1988), the other one a Genetic Algorithm (GA), (Goldberg, 1989). The basic numerical features of such algorithms are outlined and some preliminary identification results, which have been obtained from simulated data are presented in section 4.

2. UUV Mathematical Model

The considered UUV (Unmanned Underwater Vehicle), which is more extensively described elsewhere (White et Al., 1994), is torpedo shaped and is connected to the surface vessel via an umbilical cable, through which power is supplied, commands issued and data from on board sensors are fed back.
The vehicle, a schematic drawing of which is shown in Fig.1, is fitted with six thrusters and four rear control surfaces, the relative locations of which are indicated in the same figure. More specifically, control in the lateral plane can be achieved by independent use of two stern thrusters located on either side of the vehicle longitudinal centre line, two side thrusters, one at the bow and one at the stern, and an upper and lower rudder mounted aft. For control in the longitudinal plane, two vertical thrusters are provided : one at the bow and one at the stern, and two stern hydroplanes, one to port and the other to starboard. Vertical thrusters can be used for depth positioning and pitch control. All of these devices are capable of bi-directional and independent movements.
The UUV motion in 6 degrees of freedom can be conveniently expressed with respect to a reference system fixed to the vehicle centre of gravity.

Fig. 1 : UUV reference system

The corresponding mathematical model can then be expressed, (Fossen,1994), (White et Al., 1994), (Yoerger et Al., 1987), by a set of non-linear coupled Newtonian equations of the form :

$$M\dot{x} = F(x) + H(x) + G + T + D + K \quad (1)$$

where $x = [u\ v\ w\ p\ q\ r]^T$ is the vector of UUV linear and angular velocities, M is the body inertial matrix, including hydrodynamic added mass, $F(x)$ is the vector of kinematic forces and moments, $H(x)$ is the vector of hydrodynamic forces and moments, while G, T, D and K are vectors denoting forces and moments, respectively due to hydrostatics, thrusters, hydroplanes and umbilical cable.
Identification of the complete set of coefficients and hydrodynamic derivatives which appear in the above equation is a very complex task, owing to non-linearities and to the extremely high number of parameters. The identification problem can be more easily approached if it is assumed that :

- longitudinal motions are decoupled from the lateral ones;
- the cable has a negligible effect on UUV dynamics;
- surge speed is constant and linearization can be carried out around a uniform motion.

Under such assumptions, the following linear model can be deduced for the pitch-heave response to command inputs in the longitudinal plane :

$$M\dot{x} = Fx + G\delta \quad (2)$$

where the state vector $x = [w \ q \ z \ \theta]^T$ is constituted by heave rate, pitch rate, depth and pitch angle and the control vector $\delta = [\delta_s \ \delta_r \ T_r \ T_b]^T$ is given by the port and starboard hydroplane angles and by the stern and bow propeller vertical thrusters. Matrices M, F and G are given by :

$$M = \begin{pmatrix} m - \frac{1}{2}\rho l^3 Z_{\dot{w}} & -\frac{1}{2}\rho l^4 Z_{\dot{q}} & 0 & 0 \\ -\frac{1}{2}\rho l^4 M_{\dot{w}} & I_y - \frac{1}{2}\rho l^5 M_{\dot{q}} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$F = \begin{pmatrix} \frac{1}{2}\rho l^2 Z_w U & \left(\frac{1}{2}\rho l^3 Z_q + m\right)U & mw & 0 \\ \frac{1}{2}\rho l^3 M_w U & \frac{1}{2}\rho l^4 M_q & 0 & mgBG \\ 1 & 0 & 0 & -U \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

$$G = \begin{pmatrix} \frac{1}{2}\rho l^2 Z_{uu_r} U^2 & \frac{1}{2}\rho l^2 Z_{uu_b} U^2 & g_{13} & g_{14} \\ \frac{1}{2}\rho l^3 M_{uu_r} U^2 & \frac{1}{2}\rho l^3 M_{uu_b} U^2 & g_{23} & g_{24} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

where is ρ the water density, BG is the distance between the centre of gravity and the centre of buoyancy, l , m U and u are the UUV length, mass, total speed and surge speed, coefficients g_{ij} are conversion factors related to the hydroplanes and vertical thrusters, which are generally a-priori known. All the other coefficients are hydrodynamic derivatives associated to heave and pitch motions.

If an approximately uniform motion at a constant speed U is assumed, a linear time invariant state space model in the standard form can be derived, by multiplying in equation (2) for the inverse of inertia matrix M:

$$\dot{x} = Ax + B\delta \qquad (3)$$

Matrices A and B can be expressed in terms of the unknown parameter vector $\theta \in \mathbb{R}^{14}$

$$A = \begin{pmatrix} \theta_1 & \theta_2 & \theta_3 & \theta_4 \\ \theta_5 & \theta_6 & \theta_7 & \theta_8 \\ 1 & 0 & 0 & -U \\ 0 & 1 & 0 & 0 \end{pmatrix} \qquad B = \begin{pmatrix} \theta_9 & -\theta_9 & \theta_{11} & \theta_{11} \\ \theta_{10} & -\theta_{10} & \theta_{13} & \theta_{14} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

where it has been assumed that the effects of hydroplanes and of thrusters are perfectly symmetrical.

3. IDENTIFICATION METHODS

The identification problem consists of estimating the unknown parameters which characterise the vehicle dynamics on the basis of a finite number of discrete time measurements, of input vector $\{\delta(t_k)\}$ and state vector $\{x(t_k)\}$. According to the commonly used Prediction Error method (Ljung,1987), identification of the unknown parameter vector θ is equivalent to minimizing a cost function of the form :

$$J(\theta) = \frac{1}{N}\sum_{k=1}^{N} \varepsilon^T(t_k) W^{-1}(t_k) \varepsilon(t_k) \qquad (4)$$

which is constituted by a sum of squares of prediction errors, weighted through the positive definite matrix $W(t_k)$. Such minimization is commonly carried out via Gauss-Newton iterative method, which is based on an estimate of the Hessian matrix associated to prediction error cost function. As it has already been noticed in the introduction, in the presence of an unknown vector θ with a relatively high dimension, it may be difficult, owing to numerical reasons related to Hessian matrix computation, to achieve the global minimum of the assumed cost function.

Minimization methods which do not require derivatives, but are based on direct cost function evaluation seem to be preferable in such situations. The determination of the solution of a global minimization problem without the use of derivatives is generally carried out by a direct search procedure, which attempts to reduce the value of the cost function by means of proper tests near a preset point assumed as initial solution. Such tests, which are based on function evaluations in neighbouring points, determine a direction of search in which the minimum is expected to be located. The procedure is iterated until convergence has been achieved.

Two random search methods that are potentially capable to solve global minimization problems in the case of high dimensional parameter vector are Simulated Annealing (SA) and Genetic Algorithm (GA). Both methods are based on paradigms derived from natural sciences, respectively from physics and biology, and attempt to emulate the way in which the corresponding natural processes carry out optimization procedures. SA exploits an analogy with the way in which a metal cools and freezes into a minimum energy ordered crystalline structure, while the metaphor underlying GA is that of natural evolution of biological species.

Simulated Annealing

One of the most efficient random search methods for global minimization is Simulated Annealing, (Collins et al.,1988), (Aarts and Kurst,1989). This method is based on an analogy between the global minimization problem and the problem of determining the lowest energy state of a physical system. Having a large number of interacting atoms in thermal equilibrium at a specified temperature, if the system states are characterized by a parameter vector θ and $E(\theta)$ is the energy associated with this state, τ is the temperature and k_b is the Boltzmann's constant, then according to statistical mechanics, the probability $P(\theta)$ that the system is in the state θ is given by

$$P(\theta) = \exp\left(-\frac{E(\theta)}{k_b \tau}\right).$$

Under equilibrium, the most probable states at any given temperature are those associated with the lowest energy. As it can be demonstrated by theoretical arguments as well as by experimental evidence, the most effective strategy for obtaining the state with globally minimum energy consists of slowly cooling a thermodynamic system. This enables it to achieve equilibrium during the transition from a given initial state to the lowest energy state. In fact, if the cooling process is carried out sufficiently slowly, the system is allowed to skip over locally stable states and reach the global minimum energy one. Simulated Annealing consists of three distinct steps:a random search step, a minimization step and a stopping rule, as shown in Fig.2.

After the assignment of an initial value θ_0 to the unknown parameter vector, an initial temperature τ_0 is chosen high enough to ensure that virtually all transitions in the parameter space may be possible. The random search is then carried out iteratively. A new vector ξ is randomly chosen belonging to the neighbouring set $S(\theta)$ associated to the current parameter vector. Before taking a new vector ξ into account, however, a statistical test is performed to decide if equilibrium has been reached. Such test essentially consists in the verification that a given finite sequence of vectors generated by the algorithm inside the inner loop can be regarded as a realization of a time-homogeneous Markov chain.

Following a previous application of SA to the identification of a containership (Tiano and Blanke,1997), the neighbouring set $S(\theta)$ has been assumed to be constituted by the surface of a hyper-sphere centred in the current parameter vector estimate and having a suitable radius, the value of which can be taken dependent on some a-priori knowledge of parameter vector sensitivity function.

At each iteration a decision has to be taken whether or not to accept the new vector ξ in place of the current one θ_k. Acceptance of the new vector is made with probability

$$\min\left\{1, \exp\left(-\frac{J(\xi) - J(\theta_k)}{k_b \tau_k}\right)\right\}$$

where τ_k is the current value of temperature. Every descent is thus accepted, but it is also possible, even if at a limited extent, to perform also up-hill transitions, which may allow the algorithm to escape out of local minima.

The temperature is gradually, at each iteration, reduced according to a proper cooling schedule. Such cooling should be carried out quite slowly, in order to enable the algorithm to achieve equilibrium.It has been verified (Tiano and Blanke, 1997) that piecewise linear schedule is a simple and yet efficient law.

The algorithm stops when evidence has been reached that convergence has been achieved within the limits of a specified tolerance or when computing resources have been exhausted.

```
begin
    θ := initial solution θ0;
    τ := initial temperature τ0;
    while (stopping criterion is not satisfied) do
      begin
        while (not yet in equilibrium) do
          begin
            ξ := random vector selected in S(θ);
            ΔJ := J(ξ) − J(θ);
            P := min{1, e^(−ΔJ/(k_b τ))};
            φ := random generation uniform in [0, 1];
            if φ ≤ P then θ_k := ξ;
          end
        τ_k := updated temperature f(J(θ));
      end
    output of optimal solution
end
```

Fig.2 : Outline of SA algorithm

Genetic Algorithms

Genetic algorithms (Goldberg, 1989) exploit an analogy with natural evolution of biological species. In natural evolution each species searches for beneficial adaptations in an ever changing environment. As species evolves, its new attributes are encoded in chromosomes of individual members. This information changes owing to the combination and exchange of chromosomic material during breeding as well as under the influence of random mutation. According to this innovative approach, optimization problems can be described by simple bit strings, which correspond to chromosomes, and by transformation laws operating on the strings. The basic structure of a GA is carried out along the following steps :

1) Generation of an initial population
2) Assessment of initial population
3) Selection of population
4) Recombination for new population
5) Mutation of new population
6) Assessment of new population
7) Stopping criteria

In this case a set of possible solutions of the optimization problem is represented by a population, the basic characteristics of which are encoded by means of binary strings, the length of which

determines the accuracy of the final solution. The representation of continuous parameter vectors is obtained by approximating them, after a suitable rescaling, by equivalent integer variables. The generation of a new solution, as in SA algorithm, is carried out in a random way by means of the three separate activities of population selection, recombination and mutation. Solutions which survive do so in order to serve as progenitors for a new generation. It is in the recombination phase that the algorithm attempts to create new solution with improved characteristics. The purpose of mutation is to prevent from an irreversible loss of genetic information and hence to maintain diversity within the population. Like the SA method, a GA does not use derivative information. It is just necessary to be supplied with a fitness value for each member of each population.

The efficiency of a GA is highly dependent on a number of parameters , essentially represented by the population size, the crossover probability and the mutation probability, (Grefenstette, 1986).

Unlike SA, which is essentially a sequential algorithm, GA searches from one population of solutions to another one, and is therefore particularly indicated for implementation on parallel computers.

4. Identification results

The above described identification have been tested out by using simulated UUV data. Such data have been obtained by a complete six degrees of freedom non linear model. A diving manoeuvre has been simulated, where a step input has been actuated by the port and starboard hydroplanes δ_p and δs, in such a way that $\delta_p = -\delta s$ and the vertical bow and stern thrusters $T_t = T_s = 0$.

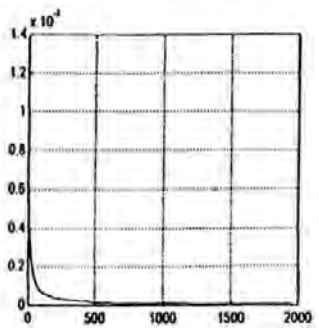


Fig.3 : Identification cost function versus iterations for SA

In this case the number of parameters to be estimated is equal to 10, since $\theta_{11} = \theta_{12} = \theta_{13} = \theta_{14} = 0$. Furthermore, it was assumed that all the four components of UUV state vector, i.e. heave velocity, pitch rate, depth and pitch were observed with a small amount of measurement errors.

SA method has been first tested, which has supplied quite good results. As it can be noticed in Fig.3, the identification cost function reduction is almost monotonic and, as shown in Fig. 4, the parameter vector convergence is quite regular. Different rates of convergence exhibited by different parameters are related to different values for the sensitivity functions. The agreement between the predicted and measured state variables can be appreciated in Fig.5, where, from top to bottom, heave velocity, pitch rate, depth and pitch angle are shown.

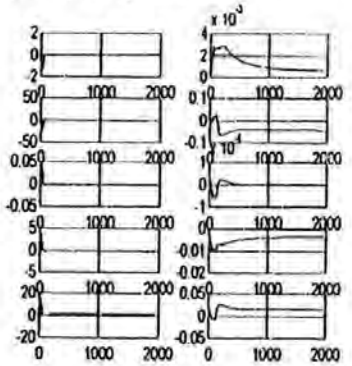


Fig.4 : Parameter vector estimate for SA algorithm

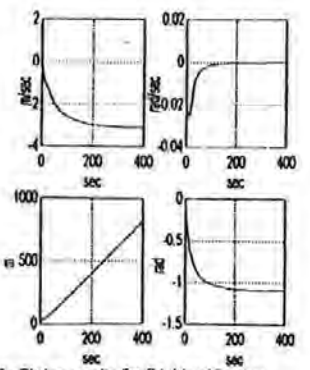


Fig.5 : Fitting results for SA identification

A GA identification algorithm has also been implemented, with a choice for the initial population of 80 members, a chromosome bit length of 20, a crossover probability of 0.7 and a mutation probability of .001.

The results obtained indicate a comparable performance in terms of convergence rate, as shown in Fig. 6, where the cost function of the optimal solution is plotted together with the mean cost value achieved by the first 4 best population members.

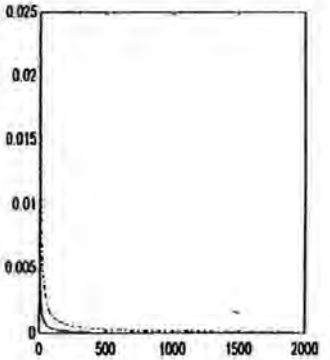


Fig.6 :Evolution of GA algorithm

It should be noticed, however, that computation time for GA identification is much longer than with SA identification. More extensive simulations, to be conducted also on parallel computers, are necessary in order to achieve a better evaluation of the performances of the two algorithms.

5. Conclusions

A novel identification approach which is based on two random search global minimization algorithms, i.e. Simulated Annealing and Genetic Algorithm, has been applied to the linearized dynamics in the diving plane of an UUV. The data used for testing such identification algorithms were obtained by complete six degrees of freedom non linear simulation model. Both identification algorithms have proved to be quite efficient in terms of capability of reaching the global minimum, with a better performance in terms of computation time of SA algorithm.

Acknowledgement

The authors gratefully acknowledge the financial support offered by MURST and the British Council in 1996 for this research on identification methods applied to marine vehicles.

References

E. Aarts and J. Korst (1989)
"Simulated Annealing and Boltzmann machines", John Wiley & Sons, New York

M.A. Abkowitz (1980)
"System identification techniques for ship manoeuvring trials", Proc. Symposium on Control Theory and Navy Applications, Monterey (USA), 337-393.

M. Caccia, G. Indiveri, A.Tiano, G. Veruggio (1997).
"Experimental comparison of identification methods for an open-frame ROV", MCMC '97, Brijuni (Croatia), September 10-12 1997.

T.I. Fossen (1994)
"Guidance and Control of Oceans vehicles", John Wiley & Sons, UK

D.E. Goldberg (1989)
"Genetic algorithms in search, optimization and machine learning", Addison Wesley, Reading, MA.

J.J. Grefenstette (1987).
"Optimization of Control Parameters for Genetic Algorithms", IEEE Trans. Systems Man and Cybernetics SMC-16,122-128.

L.Ljung (1987)
"System Identification: Theory for the User", Prentice Hall, London,

A.Tiano, M. Blanke (1997) (in press)
"Multivariable Identification of Ship Steering and Roll Motions", Transactions of Institute of Measurement and Control.

B.A. White, B.A. Stacey, Y. Patel, C. Bingham (1994).
"Robust Control Design of an ROV", Technical Report 102/94, The Royal Military College of Science, Cranfield, U.K.

D.R. Yoerger, J.J. Slotine, M. Grosenbaugh, D.M. DeLonga (1987).
"Dynamics and control of autonomous vehicles", Proc. 5th International Symposium on Unmanned Untethered Submersible Technology, 381-395..

Fuzzy Yaw Autopilots for Unmanned Underwater Vehicles Tuned Using Artificial Neural Networks

R. SUTTON* and P. J. CRAVEN*

* Marine Technology Division, Institute of Marine Studies, University of Plymouth, Drake Circus, Plymouth PL4 8AA, UK.

Abstract

This paper describes the application of neuro-fuzzy techniques in the design of autopilots for controlling the yaw dynamics of an unmanned underwater vehicle. Autopilots are designed using an adaptive-network-based fuzzy inference system (ANFIS), a simulated annealing tuning methodology, and a fixed fuzzy rule-based approach. To describe the yaw dynamic characteristics of an unmanned underwater vehicle a realistic simulation model is employed. Results are presented which demonstrate the superiority of the ANFIS approach. It is concluded that the approach offers a viable alternative method for designing such autopilots.

1. Introduction

Some of the earliest developments in unmanned underwater vehicle (UUV) technology can be attributed to the cable-controlled underwater recovery vehicle design and construction programme instigated by the US Navy in 1958. In 1963 one of these craft was used in the search for the ill-fated USS *Thresher* which tragically sank off the New England coast in 1400 fathoms of water. Later, another was used to help recover the US Navy hydrogen bomb lost off the coast of Palomares, Spain, in 1966.

Notwithstanding those successes and the accompanying publicity, the commercial potential of UUVs was not recognised until the discovery of offshore oil and gas in the North Sea. More specifically, remotely operated vehicles (ROVs) began and continue to be used extensively throughout the offshore industry. Whereas, both in the naval and commercial sectors, autonomous underwater vehicle (AUV) usage was limited.

Even so, more recently, interest in the possible use of both types of vehicle has been heightened. This has been prompted by the needs of the offshore industry to operate and explore in extreme depths in a continuously hostile environment and the requirements of navies to have low-cost vehicles capable of undertaking covert surveillance missions and performing mine laying and disposal operations. This revival is also coupled with the current and ongoing advances being made in control engineering and artificial intelligence techniques.

The dynamic characteristics of UUVs present a control design problem which classical linear design methodologies cannot accommodate easily. Fundamentally, UUV dynamics are non-linear in nature and are subject to a variety of disturbances such as varying drag forces, vortex effects and currents. Therefore they offer a challenging task in the development of suitable algorithms for motion and position control in the six degrees of freedom in which such craft operate, and are required to be robust in terms of disturbances rejection and varying vehicle speeds and dynamics.

It should be noted that the term 'unmanned underwater vehicle' as used here is a generic expression to describe both an AUV and an ROV. An AUV being a marine craft which fulfils a mission or task without being constantly monitored and supervised by a human operator, whilst an ROV is a marine vessel that requires instructions from an operator via a tethered cable or an acoustic link.

Previous studies into the development of UUV control strategies have been undertaken using advanced control engineering concepts such as $H\infty$ [1], sliding mode [2] and adaptive [3] theories. These investigations have achieved limited success owing either to the simplification of the problem or to the control scheme lacking robustness.

Artificial intelligence approaches are now also being introduced into the design process. Autopilots formulated using fuzzy logic [4, 5] and artificial neural network (ANN) methods [6, 7] have been reported and shown to be endowed with commendable robustness properties. Encouraged by such results, this paper considers the development of a course-keeping autopilot based on the innovative neurofuzzy methodology of Jang [8] known as the adaptive-network-based fuzzy inference system (ANFIS), which was successfully employed to produce a control strategy for the classical inverted pendulum problem.

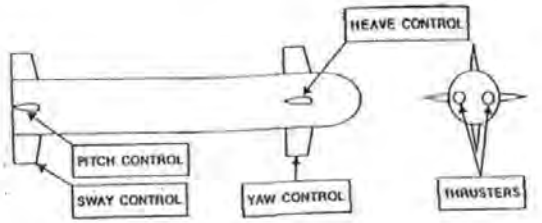


Figure 1 Control authority of the UUV

With the ANFIS approach, implementation of the controller design differs in form from the more traditional ANN in that it is not fully connected, and not all the weights or nodal parameters are modifiable. Essentially, the fuzzy rule base is encoded in a parallel fashion so that all the rules are activated simultaneously so as to allow network training algorithms to be applied. As in Jang's original work, here a back propagation algorithm is used to optimize the fuzzy sets of the premises in the ANFIS architecture and a least squares procedure is applied to the linear coefficients in the consequent terms. However, for this study a new cost function is introduced. For performance assessment purposes, comparisons are made with a fuzzy controller whose premises are tuned using a simulated annealing algorithm [9] and a fixed fuzzy rule based autopilot.

2. Modelling the UUV Dynamics

Figure 1 shows the complete control authority of the UUV model. However, it should be noted that for this study the upper and lower canards are the only surfaces used to control its yaw dynamics. Dimensionally, the model represents an underwater vehicle which is 7 m long, 1 m in diameter and has a displacement of 3600 kg.

The equation of motion describing the dynamic behaviour of the vehicle in the lateral plane is as follows [10]:

$$E\dot{x} = Fx + Gu \tag{1}$$

where:

$$E = \begin{bmatrix} (m - Y_{\dot{v}}) & -Y_{\dot{r}} & 0 & -(Y_p + mZ_G) & 0 \\ -N_{\dot{v}} & (I_z - N_{\dot{r}}) & 0 & -N_p & 0 \\ 0 & 0 & 1 & 0 & 0 \\ -(K_{\dot{v}} + mZ_G) & -K_{\dot{r}} & 0 & (I_x - K_p) & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$F = \begin{bmatrix} Y_v U & (Y_r - m)U & 0 & Y_\phi U & 0 \\ N_v U & N_r U & 0 & N_\phi U & 0 \\ 0 & 1 & 0 & 0 & 0 \\ K_v U & (K_r + mZ_G)U & 0 & K_\phi U & -mgBG \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$G = \begin{bmatrix} Y_{\delta_{UR}} U^2 & Y_{\delta_{UL}} U^2 & 0 & 0 & 1 & 1 \\ N_{\delta_{UR}} U^2 & N_{\delta_{UL}} U^2 & \ell_0 & -\ell_a & \ell_a & -\ell_a \\ 0 & 0 & 0 & 0 & 0 & 0 \\ K_{\delta_{UR}} U^2 & K_{\delta_{UL}} U^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$u = \begin{bmatrix} \delta_{UR} \\ \delta_{UL} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

and the state variables are V, R, ψ, P, and ϕ. For the interested reader, a nomenclature for the UUV parameters can be found in Appendix A. To implement equation (1) use is made of an UUV MATLAB/Simulink simulation model supplied by the Defence Research Agency (DRA), Sea Systems Sector, Winfrith. The model having been validated against standard DRA non-linear hydrodynamic code using tank test data and an experimentally derived set of hydrodynamic coefficients from the Southampton Oceanography Centre's AUTOSUB vehicle. In addition, the MATLAB/Simulink model structure also takes into account the dynamic behaviour of the canard actuators by describing them as first order lags with appropriate limiters.

As can be seen in equation (1) the roll cross-coupling dynamics are included. However, control of the roll channel is not considered here.

3. Neurofuzzy Autopilot Design

As mentioned above, the fuzzy controller design used in this study is based on the ANFIS. Functionally, there are almost no constraints on the membership functions of an adaptive network except piecewise differentiability. Structurally, the only limitation on network configuration is that it should be of feed-forward type. Due to these minimal restrictions, the adaptive network's applications are immediate and immense in various areas.

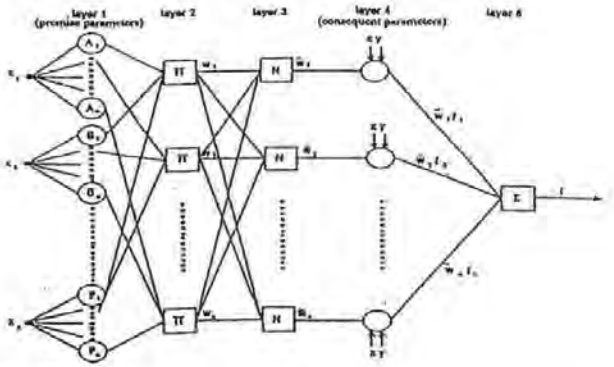


Figure 2 The ANFIS architecture

If it is assumed that the fuzzy inference system under consideration has multiple inputs and one functional output (f) then the fuzzy rule-based algorithm may be represented in the first order Sugeno form as shown below [11]:

Rule 1: If x is A_1 and y is B_1 then $f_1 = p_1x + q_1y + r_1$

Rule 2: If x is A_2 and y is B_2 then $f_2 = p_2x + q_2y + r_2$

⋮

Rule n: If x is A_n and y is B_n then $f_n = p_nx + q_ny + r_n$

The corresponding ANFIS architecture being shown in Figure 2.

The node functions in the same layer are of the same function family as described by the following:

Layer 1: Every ith node in this layer is an adaptive node with a node output defined by:

$$O_{1,i} = \mu_{A_i}(x) \tag{2}$$

where x is the input to the general node and A_i is the fuzzy set associated with this node. In other words, outputs of this layer are the membership values of the premise part. Here the membership functions for A_i can be any appropriate parameterised membership functions. Here A_i is characterised by the generalised bell function:

$$\mu_{A_i}(x) = \frac{1}{1 + \left[\left(\frac{x - c_i}{a_i}\right)^2\right]^{b_i}} \tag{3}$$

where $\{a_i, b_i, c_i\}$ is the parameter set. Parameters in this layer are referred to as *premise parameters*.

Layer 2: Every node in this layer is a fixed node labelled Π, which multiplies the incoming signals and outputs the product or T-norm operator result, e.g.

$$O_{2,i} = w_i = \mu_{A_i}(x) \times \mu_{B_i}(y), \quad i = 1, 2 \tag{4}$$

Each node output represents the *firing strength* of a rule. (In fact, any other T-norm operators that perform the fuzzy AND operation can be used as the node function in this layer).

Layer 3: Every node in this layer is a fixed node labelled N. The ith node calculates the ratio of the ith rules' firing strength to the sum of all rules' firing strengths:

$$O_{3,i} = \bar{w}_i = \frac{w_i}{w_1 + w_2}, \quad i = 1, 2 \tag{5}$$

For convenience, outputs of this layer are called *normalised firing strengths*.

Layer 4: Every ith node in this layer is an adaptive node with a node function:

$$O_{4,i} = \bar{w}_i f_i = \bar{w}_i (p_i x + q_i y + r_i) \tag{6}$$

where $\bar{w}_i$ is the output of layer 3 and $\{p_i, q_i, r_i\}$ is the parameter set. Parameters in this layer are referred to as *consequent parameters*.

Layer 5: The single node in this layer is labelled $\sum_i$ which computes the overall output as the summation of incoming signals:

$$O_{5,i} = \text{overall output} = \sum_i \bar{w}_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \tag{7}$$

Thus an adaptive network that has exactly the same function as a Sugeno fuzzy model may be constructed.

4. Simulated Annealing Tuned Autopilot Structure

The structure of the simulated annealing tuned autopilot is similar to that described in Section 3 and depicted in Figure 2 for the ANFIS architecture. However, there are dissimilarities. In this case, the nodes in Layer 4 are static and therefore are not modifiable as shown in Figure 3. Also during the tuning process, input data are only fed forward through the network in order to generate an error function. The simulated annealing algorithm is then applied to optimise the premises.

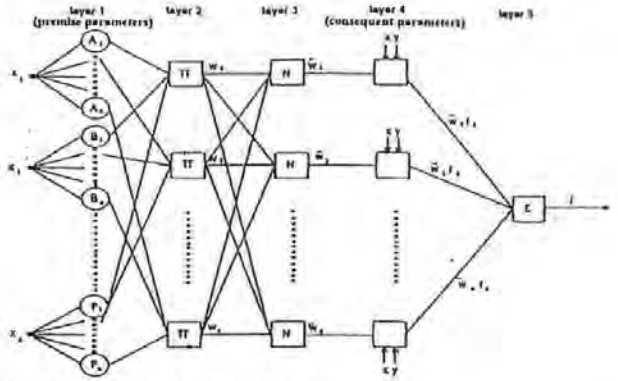


Figure 3 Simulated annealing tuned autopilot structure

5. Training Algorithms

5.1 The hybrid learning rule

This learning rule was based upon the hybrid learning rule of Jang. The system is simulated using the dynamic model and data is collected across a trajectory. This training data is used to compare the system trajectory with the desired trajectory, and so form the error measure to be used for training of the adaptive network parameters. The error measure chosen was the integral square of heading error over time (ITSE) with a rudder square component added to ensure efficient control effort:

$$E = \sum_t [(\psi_d - \psi_a)^2 + \rho(\delta)^2]t. \tag{8}$$

The parameters to be altered are the fuzzy parameters of both the premise and consequent layers. The hybrid learning rule employs the backpropagation method to update the fuzzy premise parameters and the recursive least squares method to update the fuzzy consequent parameters.

Writing the premise membership function of equation (3) as:

$$\mu_{ij}(x) = \frac{1}{1 + \left[\left(\frac{x - c_{ij}}{a_{ij}}\right)^2\right]^{b_{ij}}} \tag{9}$$

then equation (9) now represents the jth membership function on the ith input universe of discourse, where a_{ij} governs the width of set, b_{ij} governs the flatness of the bell function and c_{ij} is the centre of the set on the ith input universe of discourse. Therefore the learning rule for a general parameter may be described as follows:

$$\Delta\alpha_{ij} = -\eta \cdot \sum_{n=1}^{P} \frac{\partial E_n}{\partial O_{1n}} \cdot \frac{\partial O_{1n}}{\partial \alpha_{ij}}$$

$$= -\eta \cdot \sum_{n=1}^{P} \frac{\partial E_n}{\partial O_{1n}} \cdot \frac{\partial O_{1n}}{\partial O_{1i}} \cdot \frac{\partial O_{1i}}{\partial \alpha_{ij}} \tag{10}$$

where η is the learning rate, E_n is the error measure, P is the number of samples in the trajectory, and O_1 is the output of layer 1. If the function $O_1 = f(\alpha_{ij})$ is differentiable then $\partial O_{1i}/\partial\alpha_{ij}$ is a straightforward calculation. This was the motivation for choosing the set functions described by equation (9).

The main difficulty is in the calculation of $\partial E_n/\partial O_{1n}$. Considering the UUV model as the final layer in the adaptive network this calculation becomes simple for this layer:

$$\frac{\partial E_n}{\partial O_{1n}} = \frac{\partial}{\partial O_{1n}} \left[\sum_{n=1}^{P} (T_n - O_{1n})^2 t + \rho(O_{4n})^2\right] \tag{11}$$

$$= \sum_{n=1}^{P} -2(T_n - O_{1n})t \tag{12}$$

There are no adaptable parameters in the vehicle model layer. The next layer, layer 4, is the one that produces the defuzzified output. The computation of $\partial E_n/\partial O_{4n}$ uses a backpropagation of $\partial E_n/\partial O_{5n}$:

$$\frac{\partial E_n}{\partial O_{4n}} = \sum_{m=1}^{M(5)} \frac{\partial E_n}{\partial O_{5n}^m} \cdot \frac{\partial O_{5n}^m}{\partial O_{4n}} \tag{13}$$

where $\#(S)$ is the number of nodes in layer S. Hence

$$\frac{\partial E_n}{\partial O_{1n}} = \frac{\partial E_n}{\partial O_{1n}} \cdot \frac{\partial O_{1n}}{\partial O_{1n}} \tag{14}$$

as $\#(5) = 1$. Now $\partial O_{1n}/\partial O_{1n}$ may be written as:

$$\frac{\partial O_{1n}}{\partial O_{1n}} = \frac{d\psi}{d\delta_n} \tag{15}$$

whereby the function relating ψ to δ_n is non-linear and the derivative (or Jacobian) is approximated by:

$$\frac{\partial O_{1n}}{\partial O_{1n}} = \frac{O_1(n) - O_1(n-1)}{O_1(n) - O_1(n-1)}. \tag{16}$$

The only layer to be adapted using the back-propagation method is the first layer. Hence continuing the above process for each layer the following learning rules for each individual parameter within layer 1 are determined:

$$\Delta a_{ij} = -\eta \cdot \sum_{n=1}^{P} \frac{\partial E_n}{\partial O_{2n}} \cdot \frac{\partial O_{2n}}{\partial O_{1n}} \times \left[\frac{2b_{ij} a_{ij}^{(2b_{ij}-1)} (x - c_{ij}) a_{ij}^{2b_{ij}} (x - c_{ij})^{2b_{ij}}}{\left(a_{ij}^{2b_{ij}} (x - c_{ij})^{2b_{ij}} + (x - c_{ij})^{2b_{ij}} a_{ij}^{2b_{ij}}\right)^2} \right] \tag{17}$$

$$\Delta b_{ij} = -\eta \cdot \sum_{n=1}^{P} \frac{\partial E_n}{\partial O_{2n}} \cdot \frac{\partial O_{2n}}{\partial O_{1n}} \times \left[\frac{2a_{ij}^{2b_{ij}} (x - c_{ij})^{2b_{ij}} a_{ij}^{2b_{ij}} (x - c_{ij})^{2b_{ij}} \ln \left[\left| \frac{a_{ij}}{x - c_{ij}} \right| \right]}{\left(a_{ij}^{2b_{ij}} (x - c_{ij})^{2b_{ij}} + (x - c_{ij})^{2b_{ij}} a_{ij}^{2b_{ij}}\right)^2} \right] \tag{18}$$

$$\Delta c_{ij} = -\eta \cdot \sum_{n=1}^{P} \frac{\partial E_n}{\partial O_{2n}} \cdot \frac{\partial O_{2n}}{\partial O_{1n}} \times \left[\frac{-2b_{ij} a_{ij}^{2b_{ij}} (c_{ij} - x)^{(2b_{ij}-1)} a_{ij}^{2b_{ij}} (-(x - c_{ij}))^{2b_{ij}}}{\left(a_{ij}^{2b_{ij}} (-(x - c_{ij}))^{2b_{ij}} + (c_{ij} - x)^{2b_{ij}} a_{ij}^{2b_{ij}}\right)^2} \right] \tag{19}$$

It is given that if an adaptive network's output is linear in some of the networks parameters, then these linear parameters can be identified by the well documented least-squares method. Considering the case of one network output

$$\text{output} = F(\vec{I}, S) \tag{20}$$

where $\vec{I}$ is the vector of input variables and S is the set of parameters. If there exists a function H such that the composite function $H \cdot F$ is linear in some of the elements of S then these elements can be identified by the least-squares method. More formally, if the parameter set S can be decomposed into two sets

$$S = S_1 \oplus S_2 \tag{21}$$

(where $\oplus$ represents direct sum) such that $H \cdot F$ is linear in the elements of S_2 then upon applying H to equation (20) yields

$$H(\text{output}) = H \cdot F(\vec{I}, S) \tag{22}$$

which is linear in the elements of S_2. Now given values of elements of S_1, P training data can be collected for input into equation (22) which yields the matrix equation:

$$A\vartheta = B \tag{23}$$

where ϑ is an unknown vector whose elements are parameters in S_2. This equation represents the standard linear least-squares problem and the best solution for ϑ, which minimises $\|A\vartheta - B\|^2$, is the least-squares estimator (LSE) ϑ^*:

$$\vartheta^* = (A^T A)^{-1} A^T B \tag{24}$$

where A^T is the transpose of A and $(A^T A)^{-1} A^T$ is the pseudo-inverse of A if $A^T A$ is non-singular. The recursive LSE formula can be employed by letting the ith row vector of matrix A defined in equation (23) be a_i^T and the ith element of B be b_i^T; then ϑ can be calculated iteratively as follows:

$$\vartheta_{i+1} = \vartheta_i + S_{i+1} a_{i+1} (b_{i+1}^T - a_{i+1}^T \vartheta_i), \tag{25}$$

$$S_{i+1} = S_i - \frac{S_i a_{i+1} a_{i+1}^T S_i}{1 + a_{i+1}^T S_i a_{i+1}}, \tag{26}$$

$$i = 0, 1, \ldots, P-1$$

where the least-squares estimator ϑ^* is equal to ϑ_P. The initial conditions needed to bootstrap equations (25) and (26) are $\vartheta_0 = 0$ and $S_0 = \gamma I$ where γ is a positive large number and I is an identity matrix of dimension $M \times M$.

Consequently the gradient descent method and the least-squares method have been combined to update the parameters in an adaptive network. Each epoch consists of a forward pass in which inputs are presented and the matrices A and B are calculated and the consequent parameters are updated via the recursive least-squares method. Additionally each epoch consists of a backward pass in which the derivative of the error measure with respect to each nodes output is propagated from the output to the input of the network architecture. At the end of the backward pass the parameters of the premise layer are updated by the gradient descent method.

5.2 Simulated annealing

The main problematic aspect of gradient descent based learning algorithms for optimisation problems, such as backpropagation, is the tendency for these methods to spend long periods of time in the neighbourhood of poor or sub-optimal local minima on the error hypersurface. A technique which can be employed to overcome this

shortcoming is simulated annealing which was first introduced by Kirkpatrick *et al.* [9].

Simulated annealing is a very efficient random search method for global minimisation [12, 13]. This method is based on an analogy between the global minimisation problem and that of determining the lowest energy state of a physical system.

Kirkpatrick *et al.* [9] adapted an algorithm taken from the statistical mechanics field for converging to one of many possible cooled or low energy states. Energies of this algorithm are described by a Boltzman probability distribution such that the probability of any given energy E is an exponentially decreasing function of E. Thus, if a new matrix of parameters θ, which have been perturbed from an initially assumed solution by a randomly generated amount, lead to an improved performance of the system under consideration, then they are accepted and the process is repeated. However, if this new matrix leads to a worsened performance of the system it may be occasionally accepted with probability $P(\theta)$ such that:

$$P(\theta) = \exp\left[\frac{-E(\theta)}{kT}\right] \tag{27}$$

where $E(\theta)$ is the energy associated with the state θ, k is the Boltzman's constant and T is a temperature parameter.

For a thermodynamic system, it can be demonstrated both by theoretical arguments and experimental verification that the most effective strategy for obtaining a global minimum energy state requires the temperature to be cooled slowly. Indeed, provided the cooling process is performed sufficiently slowly, the system will by-pass locally stable states to reach one which is a global minimum. Thus, in analogous systems, the temperature T is allowed to decay during training according to the following equation:

$$T = \frac{T_0}{1 + an} \tag{28}$$

where T_0 is the initial temperature, a is a constant which governs the decay rate and n is the training epoch.

Hence, simulated annealing may be considered to consist of three distinct phases:

(i) a random search step,

(ii) a minimisation stage, and

(iii) a stopping rule.

The random search step is basically the iterative generation of random matrices in a domain $S(\theta_k)$, constituted by neighbouring matrices associated to the current matrix θ_k by:

$$\theta_k = \begin{bmatrix} \theta_k^{11} & \ldots & \theta_k^{1n} \\ \vdots & \vdots & \vdots \\ \theta_k^{m1} & \ldots & \theta_k^{mn} \end{bmatrix} \tag{28}$$

$$\theta_k \in \Re^n$$

The minimisation stage consists of applying a local minimisation routine to some of the sampled matrices. Whilst the stopping rule terminates the algorithm provided there is sufficient evidence that the global minimum has been detected within the limits of a specified accuracy or some explicit iteration number is reached.

In summary, the simulated annealing algorithm can be expressed as follows

1. Generate set of initial parameters and simulate system.
2. Make random changes to the parameters and re-simulate the system.
3. If performance improved then retain changes and re-apply.
4. If performance degraded then compute probability of accepting poorer parameters according to equations (27) and (28).
5. Generate random number in the range 0–1 and compare with probability computed at 4. If random number less than accept poorer parameters, otherwise reject.
6. Re-simulate and return to 3 until convergence.

6. Fixed Fuzzy Rule Based Autopilot Design

When in operation such an autopilot uses fuzzy rules to interpret its input data and to generate an appropriate control output. Within the context of an UUV autopilot and its internal structure the rules may take the form:

If yaw error (ψ_e) is negative and yaw rate $(\dot{\psi})$ is positive then canard demand is $f(\psi_e, \dot{\psi})$.

Where, the terms 'negative' and positive' are fuzzy sets and canard demand is some function of ψ_e and $\dot{\psi}$.

By defining universes of discourse for yaw error (ψ_e) and yaw rate $(\dot{\psi})$ as E and CE and describing the output in the Sugeno first order form, such rules may be expressed as:

$$\text{If } E_i \text{ and } CE_i \text{ then } Z_i = f(E_i, CE_i)$$

Where the fuzzy subsets E_i and CE_i are:

$$E_i = (\psi_e, \mu_{Ei}(\psi_e)) \subset E$$

$$CE_i = (\psi_e, \mu_{CEi}(\dot{\psi})) \subset CE$$

and ψ_e and $\dot{\psi}$ are elements of the appropriate universes of discourse with membership functions of $\mu_{Ei}(\psi_e)$ and $\mu_{CEi}(\dot{\psi})$ respectively.

Thus, in general, the N rules contained within the algorithm of the fixed fuzzy rule based autopilot may be expressed as:

Rule 1: If E_1 and CE_1 then $Z_1 = f(E_1, CE_1)$ else

Rule 2: If E_2 and CE_2 then $Z_2 = f(E_2, CE_2)$ else

⋮

Rule N: If E_N and CE_N then $Z_N = f(E_N, CE_N)$

In order to elicit the canard demand output (δ_d) then:

$$w_i = E_i(\psi_e) \wedge CE_i(\dot{\psi}) \tag{30}$$

$$\delta_d = \frac{\sum_i w_i Z_i}{\sum_i w_i} \tag{31}$$

7. Results and Discussion

The previous sections have discussed the development of three nine rule Sugeno type fuzzy autopilots. Firstly, the hybrid learning algorithm of Jang was applied to the task of tuning the antecedent and consequent parameters of a fuzzy autopilot. To account for some form of control effort reduction in the resulting fuzzy autopilot, a revised cost function (equation (8)) was employed. Secondly, the simulated annealing algorithm was applied to the task of tuning the antecedent parameters of a fuzzy autopilot only, whilst the consequent parameters remained fixed as equally spaced singletons. Finally, a fixed rule base fuzzy autopilot was decribed. This design was included as a means of compuring and assessing the performance of the two neurofuzzy tuning methods.

In order to adapt the fuzzy parameters of the autopilots, the ANFIS and simulated annealing autopilots were encoded as adaptive network architectures. Tuning of the network parameters took place over a series of positive and negative course changes of 40°, at a surge velocity of 7.5 knots. Time intervals of 60 seconds were allowed between consecutive course changing demands to ensure that the UUV translational and rotational motions had stabilised, and thus each course change was applied at similar initial conditions. This method was considered effective and necessary to ensure rule base symmetry.

As mentioned previously, the cost function employed during ANFIS tuning of the fuzzy autopilot incorporated a weighting parameter ρ to allow a compromise between error and control effort minimisation to be achieved. This value was varied during the tuning procedure to obtain varying degrees of compromise. The value used throughout these results gave a control effort weighting of 0.158 within the cost function.

Resulting from this tuning regime the 7.5 knot, the ANFIS autopilot was taken as:

if ψ_e is N and $\dot{\psi}$ is N then $\delta = -1.4619\psi_e - 0.3922\dot{\psi} + 0.6559$

if ψ_e is N and $\dot{\psi}$ is Z then $\delta = -0.4916\psi_e - 0.8833\dot{\psi} - 0.0502$

if ψ_e is N and $\dot{\psi}$ is P then $\delta = -0.5074\psi_e - 0.8987\dot{\psi} - 0.6972$

if ψ_e is Z and $\dot{\psi}$ is N then $\delta = +0.4542\psi_e - 0.1090\dot{\psi} - 0.7879$

if ψ_e is Z and $\dot{\psi}$ is Z then $\delta = 0.0000\psi_e + 0.0000\dot{\psi} + 0.0000$

if ψ_e is Z and $\dot{\psi}$ is P then $\delta = -0.4542\psi_e - 0.1090\dot{\psi} - 0.7879$

if ψ_e is P and $\dot{\psi}$ is N then $\delta = -0.5074\psi_e - 0.8987\dot{\psi} + 0.6972$

if ψ_e is P and $\dot{\psi}$ is Z then $\delta = -0.4916\psi_e - 0.8833\dot{\psi} + 0.0502$

if ψ_e is P and $\dot{\psi}$ is P then $\delta = -1.4619\psi_e - 0.8922\dot{\psi} - 0.6559$

Again it should be noted that the only parameters for adaption within the simulated annealing tuned autopilot were the antecedent parameters and thus the syntax for the final fuzzy autopilot was the same as the fixed rule based fuzzy autopilot:

if ψ_e is N and $\dot{\psi}$ is N then $\delta = +25.00$

if ψ_e is N and $\dot{\psi}$ is Z then $\delta = +18.75$

if ψ_e is N and $\dot{\psi}$ is P then $\delta = +12.5$

if ψ_e is Z and $\dot{\psi}$ is N then $\delta = +6.25$

if ψ_e is Z and $\dot{\psi}$ is Z then $\delta = 0.00$

if ψ_e is Z and $\dot{\psi}$ is P then $\delta = -6.250$

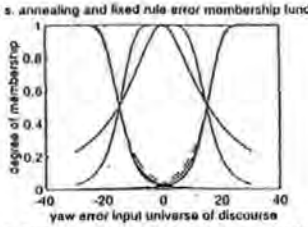
if ψ_e is P and $\dot{\psi}$ is N then $\delta = -12.50$

if ψ_e is P and $\dot{\psi}$ is Z then $\delta = -18.75$

if ψ_e is P and $\dot{\psi}$ is P then $\delta = -25.00$

By using simplified fuzzy if–then rules of this form the difficulty experienced in assigning appropriate linguistic terms to the nonfuzzy consequents is avoided. Indeed it can be proven that under this form of fuzzy if–then rule the resulting fuzzy inference system has unlimited approximation power to match any non-linear functions arbitrarily well.

ANFIS, ψ_e annealing and fixed rule error membership functions



ANFIS, $\dot{\psi}_e$ annealing and fixed rule derivative membership functions

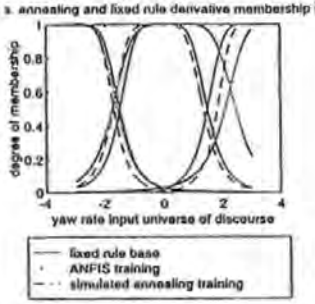


Figure 4 Input fuzzy sets for the three autopilots

Given sufficient training time the resulting input fuzzy sets for all three autopilots were as shown in Figure 4. Note the non-symmetrical nature of the tuned input fuzzy sets over the fixed fuzzy sets. This was probably due to computer truncation errors arising during the training process, and the approximate nature of the initial conditions required to bootstrap the calculation of the sequential least squares estimate for the ANFIS tuned input membership functions.

A qualitative assessment of the autopilot responses was provided by the UUV models responses to a series of random course changes as shown in Figure 5. Although Figure 5 does not provide conclusive evidence of the ANFIS autopilot's superior performance over the simulated annealing and fixed rule base autopilots, it is apparent that the hybrid learning algorithm of the ANFIS technique leads to faster, more accurate responses. Indeed it would seem that the ANFIS tuned autopilot is somewhat more effective at the course changing task than the remaining two autopilots with improved response times and only minor increases in overshoot as a consequence. Further experiments into the selection of the control effort weighting parameter for the ANFIS cost function show that a suitable compromise can be achieved between course changing response time and canard activity.

Testing each autopilot designed at 7.5 knots over the UUV speed envelope provided suitable insight into the robustness of each controller. Figure 6 depicts the yaw responses of each autopilot to a 40° course changing manoeuvre at 5, 7.5 and 10 knots.

At 5 knots the effectiveness of the canard control surfaces is significantly reduced due to the diminished hydrodynamic forces acting on them. Intuitively one would expect increased rise times as a consequence of this. During the simulations both the ANFIS and simulated annealing tuned autopilots produced good responses to the set points of the validation track at all operating speeds. At 10 knots the response times of each autopilot were significantly reduced, often at the expense of increased overshoots and in general more oscillatory behaviour. The ANFIS tuned autopilot responded faster at all three operating speeds. Both the neurally tuned autopilots faired exceptionally well with no evidence of steady state errors or unstable behaviour over the whole speed envelope of the UUV, thus showing good autopilot robustness to forward speed variation.

On the basis of the qualitative performances the ANFIS tuned autopilot was deemed the best at the required course changing manoeuvres due to its faster response. Although the overshoots of the ANFIS autopilot were greater than those of the simulated annealing autopilot, they were considered well within acceptable limits. The ANFIS autopilot also demonstrated marginally less oscillatory behaviour.

Earlier the qualitative performance of each fuzzy autopilot was discussed. This section addresses the performances of each autopilot in a quantitative manner. As a means of measuring the accuracy and sudder activity of a given autopilot, the integral square error (ISE) for the yaw error (ψ_e) and the canard demand (δ_e) are employed:

$$\psi_e = \int_{t_1}^{t_1} (\psi_d - \psi_a)^2 \, dt \tag{32}$$

$$\delta_e = \int_{t_1}^{t_1} (\delta_d - \delta_a)^2 \, dt \tag{33}$$

where ψ_d and δ_d represent desired yaw angle and canard demand, and ψ_a and δ_a represent actual yaw angle and canard demand respectively. To assess the speed of response of the control system the rise time (T_R) was calculated for each fuzzy autopilot, and the percentage peak overshoot ($M_p(t)$) was calculated to assess the oscillatory nature of each response. Here rise time is taken as the time to reach 99% of the desired 40° course change, i.e. 39.6°, and the peak overshoot is measured as a relative percentage of the 40° course change demand.

As the training took place at 7.5 knots, the robustness of each fuzzy autopilot was assessed by testing at training speed ±50%. Thus Table

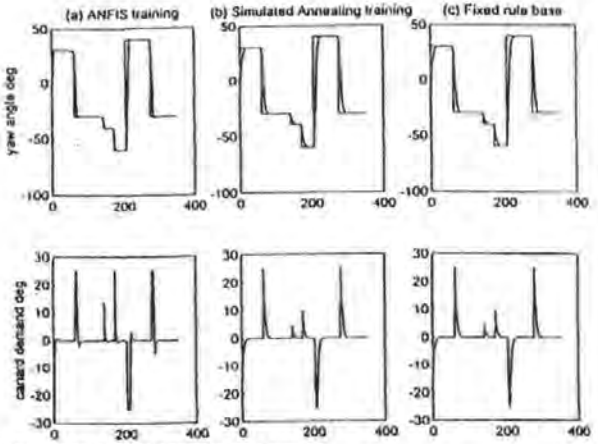


Figure 5 Yaw responses and canard demands for the autopilots.

I contains figures pertaining to 5, 7.5 and 10 knots.

Summarised in Table I are the results for the three fuzzy autopilots. When operating at 7.5 knots it appears the autopilot designed using the ANFIS technique was considerably more accurate than those of the simulated annealing tuned and fixed rule base autopilots. However, the minimum rudder demand was exercised by the fixed rule base autopilot, the activity being 2.38% less than the simulated annealing tuned autopilot. This highlights the fact that the simulated annealing algorithm has reduced course changing error by 3.96% at the expense of a 2.38% increase in canard activity. As suggested in the previous section the canard demands of the ANFIS tuned autopilot increased over the training period but remained well within the limits of acceptability. Indeed the canard demands produced by ANFIS training were much sharper than those of simulated annealing tuning.

Again, at 5 knots the autopilot developed using simulated annealing was 2.77% more accurate than the fixed rule base fuzzy autopilot at the expense of an increased canard activity of 3.53%. Additionally, the ANFIS tuned autopilot was approximately 31.1% more accurate but again demanded increased canard activity, of approximately 87.6%. The canard responses of the ANFIS autopilot were deemed acceptable at this operating speed as the periods of canard saturation did not lead to unstable UUV behaviour.

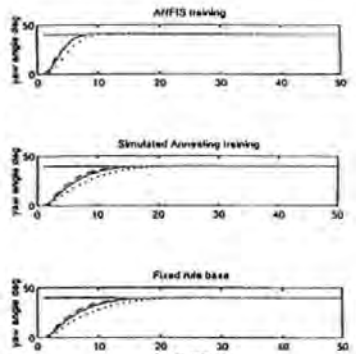


Figure 6 Robustness testing of autopilots for yaw control over a speed range. Key: ··· 5 knots, —— 7.5 knots, – – – 10 knots.

Finally, the increased effectiveness of the canard control surfaces at the higher operating speed of 10 knots led to reduced periods of canard saturation over the larger course changing manoeuvres. The simulated annealing autopilot performed well at this operating speed showing an improvement in accuracy of 7.39% corresponding to a reduction in canard activity of 1.89%. Again the ANFIS autopilot was deemed the best with the most impressive rise times and quite acceptable overshoots.

Table 1 Performance assessment of the autopilots

UUV Model	ANFIS autopilot				Simulated annealing autopilot				Fixed rule autopilot			
	ψ_t deg^2	δ_t deg^2	T_R secs	$M_P(t)$ %	ψ_t deg^2	δ_t deg^2	T_R secs	$M_P(t)$ %	ψ_t deg^2	δ_t deg^2	T_R secs	$M_P(t)$ %
5 knots	83.18	33.86	9.78	2.86	117.39	18.71	19.98	0.61	120.73	18.05	19.10	0.05
7.5 knots	59.29	20.98	7.79	1.90	85.46	11.33	15.29	0.01	88.98	11.08	15.91	0.02
10 knots	46.02	13.90	7.51	1.32	66.58	7.79	14.30	0.01	74.46	7.94	13.53	0.02

8. Conclusions

This paper has discussed the tuning of fuzzy autopilots for yaw control of an UUV. By encoding the fuzzy autopilots as adaptive network architectures, fuzzy parameters can be tuned using neural network techniques. The resulting autopilots thus remain purely fuzzy as parameter tuning is conducted off-line. From the results presented it may be concluded that the ANFIS approach provides the best autopilot design solution.

Acknowledgements

The authors wish to thank the Sea Systems Sector, DRA, Winfrith, for supplying the UUV model and their continuing support throughout this work.

References

1. Cowling, D. 'Full range autopilot design for an unmanned underwater vehicle', Proceedings of 1996 IFAC 13th Triennial World Congress, San Francisco, USA, Vol. Q, pp. 339–344.
2. Cristi, R., Papoulias, F. A. & Healey, A J. 'Adaptive sliding mode control of autonomous underwater vehicles in the dive plane', *IEEE Journal of Oceanic Engineering*, Vol. 15, No. 3, 1990, pp. 152–160.
3. Forsen, T. I. & Fjellstad, O.-E. 'Robust adaptive control of underwater vehicles', Proceedings of 3rd IFAC Workshop on Control Applications in Marine Systems, Trondheim, Norway, 10–12 May, 1995, pp. 66–74.
4. Fairbrother, H. N. R., Stacey, B. A. & Sutton, R. 'A self-organising controller for an ROV', Proceedings of IEE Control '91, Edinburgh, UK, 1991, pp. 499–504.
5. Smith, S. M., Rae, G. J. S., Anderson, D. T. & Shein, A. M. 'Fuzzy logic control of an autonomous underwater vehicle', Proceedings of 1st IFAC International Workshop on Intelligent Autonomous Vehicles, Southampton, UK, 18–21 April 1993, pp. 318–323.
6. Fujii, T. & Ura, T. 'Development of motion control system for AUV using neural nets', Proceedings of AUV 90, Washington, USA, 1990, pp. 81–86.
7. Yuh, J. 'A neural network controller for underwater robotic vehicles', *IEEE Journal of Oceanic Engineering*, Vol. 15, No. 3, 1990, pp. 161–166.
8. Jang, J. S. R. 'ANFIS: adaptive network-based fuzzy inference system', *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 23, No. 3, 1993, pp. 665–685.
9. Kirkpatrick, S., Gelatt, C. & Vecchi, M. 'Optimisation by simulated annealing', *Science*, 220, 1983, pp. 671–680.
10. Marshfield, W. B. 'Submarine data set for use in autopilot research', Technical Memorandum, DRA/MAR TM (MTH) 92314, DRA Haslar, April 1992.
11. Sugeno, M. (Ed). *Industrial applications of fuzzy control*, North Holland, The Netherlands, 1985.
12. Aarts, E. & Korst, J. *Simulated annealing and Boltzmann machines*, John Wiley and Sons, New York, 1989.
13. Collins, N. E., Eglese, R. W. & Golden, B. L. 'Simulated annealing: An annotated bibliography', *American Journal of Mathematical and Management Sciences*, 8, 1988, pp. 209–308.

Appendix A: Nomenclature of the UUV Equation Parameters

E, F, G	State equation matrices
m	Mass of UUV
P, R	Angular velocity components of rolling and yawing
U, V	Velocity components of surge and sway
ϕ, ψ	Angles of roll and heading
I	Moment of inertia
K, N	Moment components
Y	Force components
B	Centre of buoyancy
G	Centre of mass
K_{UP}	Dimensional hydrodynamic coefficients
N_{UUv_w} etc	Non-dimensional hydrodynamic coefficients
ℓ_ϕ	Roll moment arm length
ℓ_ψ	Yaw moment arm length
δ_{UP}, δ_{LO}	Upper and lower canard inputs

Control Strategies for Unmanned Underwater Vehicles

Paul J. Craven, Robert Sutton and Roland S. Burns

(University of Plymouth)

In recent years, both the offshore industry and the navies of the world have become increasingly interested in the potential operational usage of unmanned underwater vehicles. This paper provides a comprehensive review of a number of modern control approaches and artificial intelligence techniques which have been applied to the autopilot design problem for such craft.

1. INTRODUCTION. Although Bourne can be credited with producing the first conceptual design for a submarine in 1578, the first one built was constructed in 1620 by Van Drebbel. Nevertheless, it was not until 1776 that a submarine was specifically launched to take part in naval operations. Bushnell's submarine the *Turtle* was designed to destroy the Royal Navy men-of-war which were participating in naval blockades during the American War of Independence. Fortunately for the British fleet the attacks by the human-powered *Turtle* were unsuccessful. The *Turtle's* single crew member blamed the ineffectiveness of the assaults on the inability to lay 150-pound charges against the hulls of the ships owing to their reputed copper sheathing. In actual fact, the British warships were not sheathed. A more probable explanation has been postulated by Coverdale and Cassidy,[1] who propose it was due to the crew member being physically exhausted and affected by the build-up of unacceptable carbon dioxide levels in the vessel by the time it reached an intended target. Reader *et al.*[2] light-heartedly suggest that this may have been the initial impetus for the search for unmanned underwater vehicles (UUVs)! Clearly, since those pioneering days, manned submarine technology has advanced dramatically. However, the common potential weakness throughout their evolution has been the reliance on humans to perform operational tasks.

Some of the earliest developments in UUV technology can be attributed to the cable-controlled underwater recovery vehicle design and construction programme instigated by the US Navy in 1958. In 1963, one of these craft was used in the search for the ill-fated USS *Thresher* which tragically sank off the New England coast in 1400 fathoms of water. Later, another was used to help recover the US Navy hydrogen bomb lost off the coast of Palomares, Spain, in 1966. Notwithstanding those successes and the accompanying publicity, the commercial potential of UUVs was not recognised until the discovery of offshore oil and gas in the North Sea. More specifically, remotely operated vehicles (ROVs) began and continue to be used extensively throughout the offshore industry. However, both in the naval and commercial sectors, autonomous underwater vehicle (AUV) usage

was limited. Even so, more recently, interest in the possible use of both types of vehicle has been heightened. This has been prompted by the needs of the offshore industry to operate and explore in extreme depths in a continuously hostile environment and the requirements of navies to have low-cost vehicles capable of undertaking covert surveillance missions and performing mine laying and disposal operations. This revival is also coupled with the current and ongoing advances being made in control engineering and artificial intelligence techniques.

The dynamic characteristics of UUVs present a control system design problem which classical linear design methodologies cannot accommodate easily. Fundamentally, UUV dynamics are non-linear in nature and are subject to a variety of disturbances such as varying drag forces, vorticity effects and currents. Therefore they offer a challenging task in the development of suitable algorithms for motion and position control in the six degrees of freedom in which such craft operate, and are required to be robust in terms of disturbance rejection and varying vehicle speeds and dynamics. It should be noted that the term 'unmanned underwater vehicle' as used here is a generic expression to describe both an AUV and an ROV, an AUV being a marine craft which fulfils a mission or task without being constantly monitored and supervised by a human operator, whilst an ROV is a marine vessel that requires instructions from an operator *via* a tethered cable or an acoustic link.

The purpose of this paper is to review a number of modern approaches which have been adopted to control the dynamic behaviour of UUVs with an emphasis being made on artificial intelligence techniques. In the forthcoming text several references will be made to the 'robustness qualities' of certain controllers. Therefore it is felt an explanation of this phrase is in order. During the design stage of an algorithm, some form of optimised mathematical model which describes the dynamics of the vehicle to be controlled will be employed. When in actual operation, and over a period of time, the characteristics of the plant will change from those for which the control system was originally designed. Also within a class of vehicles, differences in their performances will exist. Hence, a control system is said to have good robustness qualities provided it can cope with plant uncertainties whilst possessing noise and disturbance rejection properties.

2. CLASSICAL AND MODERN TECHNIQUES

2.1. *Proportional-integral-derivative control.*

The first autopilots to be called proportional (P) controllers were employed in the period 1930–50 and employed a heading error signal which was then used to adjust the steering mechanism of a ship. These controllers had no method of reducing overshoot and thus caused transient oscillations in the actual heading of the ship. The 1950s saw the introduction of the derivative (D) term which improved the stability of the controlled vessel. Around the same period the integral (I) term was also implemented to produce counter thrust to external disturbances, this controller being based on classical PID control theory. The control action u_c produced by the PID controller is given by:

$$u_c = K_p\left[e(t) + \frac{1}{T_i}\int e(t) + T_d\frac{de(t)}{dt}\right] \qquad (1)$$

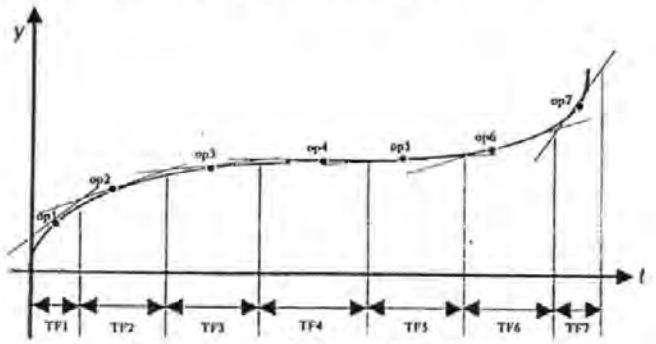


Fig. 1. Linearisation of a non-linear function

where K_p is the proportional gain of the controller, T_i is the integral time constant, T_d is the derivative time constant and $e(t)$ is the error signal.

The transfer function of the controller, in the Laplace domain, then becomes:

$$\frac{U_c(s)}{E_s(s)} = K_p\left[1 + T_d s + \frac{1}{T_i s}\right]. \qquad (2)$$

When controlling underwater vehicles, the ideal requirement is to have high proportional gain in order to ensure rapid response to error and effective removal of steady state errors; however, these requirements tend to reduce the stability of the system and so a high derivative gain is also demanded. This allows these requirements to be met. The PID control strategy depends upon the availability of an accurate linear representation of the relevant non-linear vehicle dynamics. Such models take the form of transfer functions in the Laplace or z-domains (Eqn (3)), representing the input/output relationship to be controlled.

$$G(z) = \frac{\sum_{i=0}^{n} a_i z^{-i}}{\sum_{i=0}^{n} b_i z^{-i}}. \qquad (3)$$

In Fig. 1 a non-linear relationship is represented. At specific positions on this relationship, known as operating points (OPs), the dominant non-linear dynamics can be linearised to give a transfer function (TF) at that point. Provided the operating points are chosen judiciously, it is possible to approximate the non-linear function as a series of linear transfer functions across the whole operating range.

For each of these operating regions where the linear relationship generally holds true, a PID controller, a PI controller or a PD controller can be designed to meet the required specifications. Methods to perform this are well established.[3–5] However, the further the system diverges from the operating point, the more likely it is that the non-linearities of the system will dominate

Additionally, controllers based on this format are not able to prevent the high-frequency rudder movements that often arise. In 1953 Motora[6] applied a low-pass filter to the output signal to prevent rudder oscillation but this, it was suggested, would represent a loss in stability. Taylor[7] reports that PID controllers have historically been employed in ship course-keeping modes due to the manual complexity of the aforementioned tuning process when on full-scale sea trials. Da Cunha *et al.*[8] employ this technique as a PI controller where the derivative term is neglected. This is in fact a popular approach and has been utilised in many marine control applications, for example by Hsu *et al.*[9]

UUV dynamics are such that it is usually very difficult to define clearly the dominant characteristics in order to derive a transfer function which is representative of the system under consideration.

2.2. *Gain-scheduling control.* One method often used as a means of adjusting the gain terms of a linear controller to suit the current operating range is gain scheduling. Depending on the current operating conditions, the gain terms of the controller will be adjusted to provide the best controller performance. To aid transition between operating regions, an interpolating function is often used. The structure of a typical gain-scheduled controller is shown in Fig. 2 where a

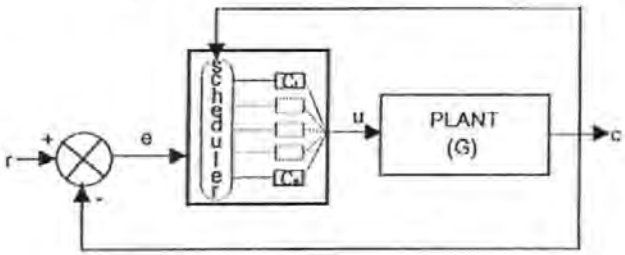


Fig. 2. A gain-scheduled controller

scheduling algorithm monitors the operating condition of the system and selects the controller parameters ($C_1 \ldots C_n$) best able to produce satisfactory performance. If designed correctly the algorithm will also ensure a bumpless transfer between the operating points.

The MARIUS (marine utility vehicle system) is an example of a successful implementation of a gain-scheduled controller in an UUV.[10] This work documents four fundamental steps to the implementation of such a control strategy:

(i) Linearise the plant about a finite number of points.
(ii) Design linear controllers around each operating point.
(iii) Interpolate the parameters of the linear controllers of step (ii) to achieve adequate performance of the linearised closed-loop systems at all points where the plant is expected to operate. The interpolation is performed according to an external scheduling vector and the resulting family of linear controllers is referred to as a *gain scheduling controller*.

The linear controllers outlined in step (ii) often take the form of PID controllers which are therefore time-consuming to develop. This technique requires thorough research into suitable controllers for each operating point and is not well documented among UUV control techniques, although a velocity alogorithm for the subsequent implementation of a non-linear gain-scheduling controller has been reported.[11]

2.3. *Adaptive control.* The past twenty-five years have seen a considerable amount of research in the application of control techniques to the problem of course-keeping and manoeuvring of marine vehicles, particularly in the field of autopilots which adapt to dynamic and environmental changes, and consequently update the controller's parameters to cope with these disturbances. The popularity of adaptive techniques concerns the poorly known hydrodynamic coefficients of the vehicle as well as the inherent non-linearities usually involved. Indeed, the majority of successful UUV control studies can be seen to include some form of adaptive control strategy. A comprehensive outline of adaptive control techniques and a brief review of the historical developments is given in reference 12. Additionally, Astrom and Wittenmark[13] report the implementation of an adaptive autopilot for the ship course-keeping task, based on a PID algorithm, the *Steermaster 2000*.

2.3.1. *Model reference adaptive control.* The model reference adaptive control (MRAC) technique is one of the main approaches to adaptive control. The desired performance of the system is given by a reference model. A feedback loop allows an error measure to be computed between the output of the system and the reference model. Thus, based upon the error measure, the parameters of the controller are adjusted to reduce this error measure. MRAC techniques were first designed to control the servo problem in deterministic continuous-time systems. Figure 3 shows a typical MRAC scheme.

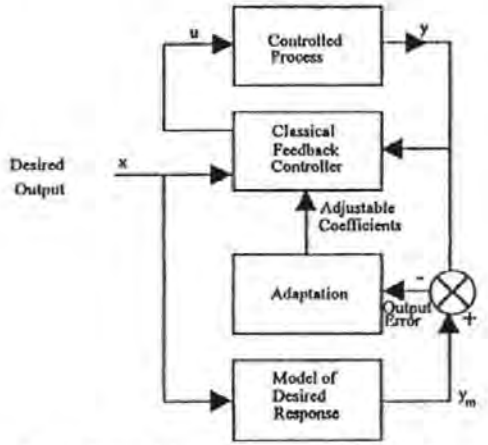


Fig. 3. The model reference adaptive control scheme

The key problem with this approach is in the determination of the adjustment mechanism in order that a stable system is achieved. The MIT rule (Eqn (4)) was developed for the original adjustment of parameters in the MRAC system.

$$\frac{d\theta}{dt} = -\gamma e \frac{de}{d\theta}. \tag{4}$$

Van Amerongen[14] applied the MRAC approach to the problem of ship course-keeping and manoeuvring. This process employed a cost function (Eqn (5)) which was chosen to enable the derivation of optimal controller gains, *via* the use of a reference model (for comparison purposes).

$$J = \int_0^T (e^2 + \lambda_1 \psi + \lambda_2 \delta^2)\, dt. \tag{5}$$

This method was based on the assumptions that the modelled process was linear and external disturbances could be disregarded. Thus, obvious criticisms of this approach are the linearity assumptions concerning the vehicle's dynamics, and the technique is only really viable when external conditions can be considered unimportant, which is almost never for a marine application. In the surveyed literature this technique is scarcely mentioned with respect to UUV control strategies. Shimmin and Lucas[15] document the technique and note the instability of such systems when the adaptation method relates controller parameter adjustments to system errors.

Da Cunha *et al.*[9] designed an adaptive position controller for an ROV, based on a recently developed output feedback variable structure control algorithm, which was given the acronym VS-MRAC. The performance of the technique is evaluated by initially using simulation models and then full-scale sea trials utilising an actual ROV. Results show that this technique of position control consistently outperforms the conventional PI control technique, by providing more accurate position tracking and disturbance rejection.

2.3.2. *Indirect adaptive control.* Farrell and Clauberg[16] report the implementation of an indirect adaptive control system on-board the AUV *Sea Squirt*. This system is designed in two layers, a standard adaptive layer and a 'supervisory logic' layer (to control the behaviour of the adaptive layer), as shown in Fig. 4. Although this technique provides convergence of the modelling parameters towards their optimum tracking performance values, it takes no account of the varying mission-to-mission modularity and dynamics that such vehicles often encounter, thus suggesting the incorporation of some form of learning control strategy to model/estimate these variations. It is felt that this addition may also provide a means by which the vehicle can compensate for variations in hydrodynamics, effected by velocity variations.

2.4. *Self-tuning control.* Self-tuning controllers can be used for processes with time-varying dynamics, by using a derived model of the process and environment to adjust the coefficients of the controller in order to satisfy a desired closed-loop system performance. A key publication by Astrom and Wittenmark[13] states that self-tuning can be applied to a controller if, initially, control of the system is

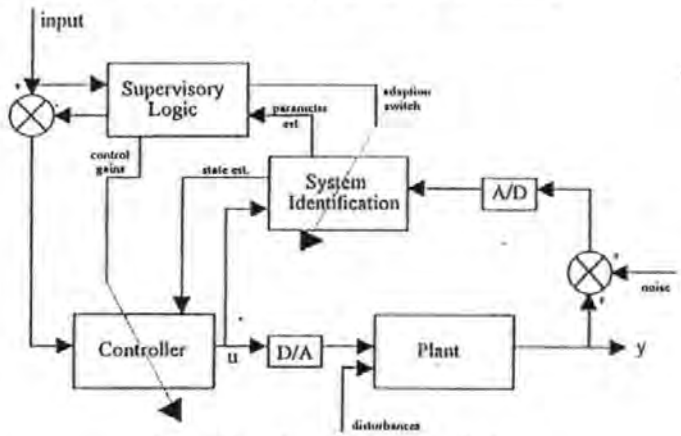


Fig. 4. The indirect adaptive control scheme

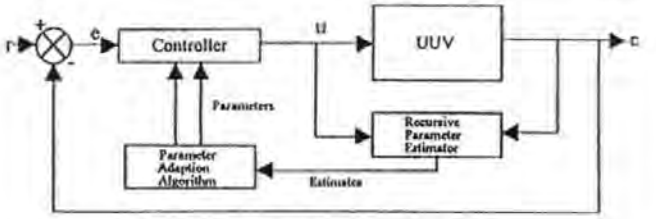


Fig. 5. Explicit self-tuning control system

by adjusting the coefficients of the controller by means of a second, slower recursive loop to achieve better control. Typically, algorithms have been implemented for adaptive ship steering and the dynamic positioning of drill ships.[17]

Triantafyllou and Grosenbaugh[18] applied a multi input–multi output (MIMO) self-tuning controller to the difficult problem of automatic guidance of an AUV by manipulating thruster outputs to produce the desired translational and yaw velocities. The self-tuning controller was used due to the lack of an accurate mathematical model of the AUV's open loop dynamics.

2.4.1. *Explicit self-tuning control.* An explicit self-tuning control scheme is shown in Fig. 5. In this control law all unknown dynamics are characterised by a time constant and steady-state gain between each velocity output and thruster input. The open-loop system can therefore be approximated by a square m by m matrix with a first order lag if, and only if

$$G_A^{-1}(z) = (z-1)B_0 + B_1 \tag{6}$$

for some $m \times m$ matrices B_0 and B_1 where $|B_0| \neq 0$. In the case of $m = 1$, this

It was established by Owens and Chotai,[19] using a PI controller for this type of system, that for minimum phase systems with fast enough sampling rates, the closed-loop system becomes decoupled into m separate single input-single output (SISO) loops. It can also be shown that reference signals will be tracked with zero steady-state errors.

The self-tuning mechanism works by using the thruster input/velocity output data to update the model at each sampling instant. The model is of the form:

$$\overset{v}{y}(t+1) = Ay(t) + Bu(t) \tag{7}$$

where $\overset{v}{y}(t+1)$ is the vector of predicted velocity outputs. The matrices A and B of Eqn (7) are used along with the closed-loop poles to calculate the PI control law, which takes the form:

$$B_0 = B^{-1} \tag{8}$$

$$B_1 = B^{-1}(I-A). \tag{9}$$

Finally the thruster inputs are determined and applied and the entire cycle is repeated. The α_i factors in the cost function (Eqn (10)) can be used to discount measurements that are known to be spurious or to give less weight to values of the model output when the algorithm is 'tuning-in' after a large change in the real system.

The versatility of this control scheme is demonstrated in Katebi and Byrne,[20] where it was employed to provide adaptation capability to a ship autopilot in adverse weather conditions and was customised to produce minimum variance to low-frequency heading variations and resistance to steering, again using the cost function of Eqn (10):

$$J = \frac{1}{n}\sum_{i=1}^{n} \alpha_i \|y(i) - \overset{v}{y}(i)\|^2 \tag{10}$$

that is, the weighted sum of the squared errors between the real system and model responses. For time-varying dynamics, Yuh *et al.*[21] employed an adaptive alogrithm to control the pitch of an underwater robotic vehicle. From the results obtained, it was found that, as the parameters of the adaptive controller had been originally derived from rough estimates of the system model, its initial performance was poor but improved significantly after a few iterations. However, it was also found that the stability of the system cannot be guaranteed for unmodelled system dynamics.

Goheen and Jeffreys[22] implemented explicit 'one-shot self-tuning' in the *Seapup* and *PAP104* underwater vehicles, whereby first-order lags for the sway, yaw and surge velocities are used together with a PI controller. The resulting controller displays the expected robustness of PI control but does not adequately account for the inherent non-linearity of the UUVs, due to the underlying linearity of such self-tuning controllers. The one-shot controller is not an adaptive strategy either and, as such, cannot account for the time-varying dynamics of the UUV.

2.4.2. *Implicit self-tuning control.* An implicit self-tuning controller identifies the parameters of the system directly and then uses these data in the control law

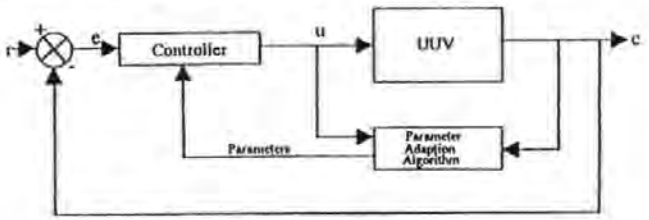


Fig. 6. Implicit self-tuning control system

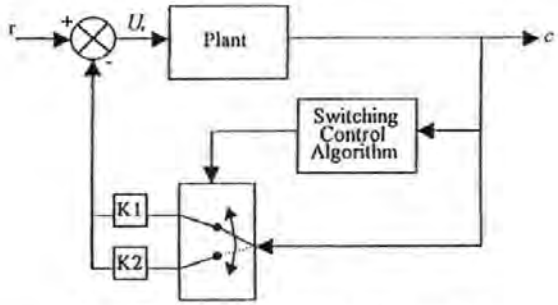


Fig. 7. A schematic of a sliding mode controller

of the system characteristics. Figure 6 shows the schematic diagram of an implicit self-tuning control system. The implicit self-tuning controller alorithms of Clarke and Gawthorp[23] incorporated the controller output into a cost function for variance minimisation of performance objectives. This approach was also employed by Lim and Forsythe[24] in their design of an autopilot applied to ship control. The expression for the cost function output in order to minimise the expected variance is Eqn (11):

$$J = E\{\|y(t+1) - r(t)\|^2 + \lambda u(t)^2\} \tag{11}$$

where y is the actual output; r is the desired output; u is the control input; $E\{*\}$ is the form of the variance to be minimised; and λ is a constant factor which is a compromise between the control action and reference tracking against adaptation speed and is determined empirically.

Therefore, the generalised output Θ:

$$\Theta(t+1) = \sum_{i=0}^{ny} F_i y(t-i) + \sum_{j=0}^{nr} H_j r(t-j) + \sum_{k=0}^{nu} G_k u(t-k) \tag{12}$$

is identified and the control law follows directly, hence the implicit nature is that of:

$$u(t) = G_0^{-1}\left[\sum^{ny} F_i y(t-i) + \sum^{nr} H_j r(t-j) + \sum^{nu} G_k u(t-k)\right] \tag{13}$$

as described in the paper by Goheen and Jeffreys,[25] who showed that second-order models ($ny = nr = nu = 2$) produced the best closed-loop response with the *Seapup* simulation data, the reason being that the second-order pitch and roll dynamics couple in other modes, unless the centre of rotation is chosen as the origin. This situation very rarely occurs in a UUV due to loading and weight distribution along the vehicles axes. This application somewhat omitted the finer points of the controller implementation.

Explicit self-tuning has the advantage of being computationally easier to implement as concerns the process algorithms than the implicit method; however, the implicit method does not require the same degree of control knowledge by the operator to determine the correct closed-loop pole positions to meet the performance criteria.

2.5. *Sliding mode control.* A non-linear strategy that has been extensively applied to the UUV control problem is that of the sliding mode controller. A switching control law transforms the state trajectory of the plant onto a user-chosen sliding surface in the state space, thus providing a technique that is robust to parametric uncertainty. Figure 7 shows a schematic of a sliding mode controller.

The reader is referred to Cristi *et al.*,[26] who state that any UUV description based on a set of differential equations can only be approximate in its nature and therefore there are uncertainties in the model. This calls for a robust input u, of the form:

$$u = u + \bar{u} \tag{14}$$

where u is determined on the basis of the nominal model and $\bar{u}$ compensates for deviations from ideal performance due to uncertainties.

Sliding control theory has been developed to apply to a large class of non-linear systems.[27,28] The only restriction on the choice of sliding surface is that it has to be associated with stable dynamics; that is, the following applies:

$$s(x(t)) = 0, \quad \text{for all } t > t \Rightarrow \lim_{t \to 0} x(t) = 0 \tag{15}$$

for any initial conditions $x(t_0)$. The choice of a linear sliding surface being:

$$s(x) = s^T x \tag{16}$$

for some vectors, $s^T x$ allows the use of pole placement techniques in the design of the non-linear controller. Using the defined Lyapunov function,[26] the sliding surface $s(x) = 0$ is reached in a finite time by the condition:

$$\sigma\dot{\sigma} = -\eta_0^2(x)|\sigma(x)| \text{ or} \tag{17}$$

$$\dot{\sigma} = -\eta_0^2(x)\,\text{sign}\,(s). \tag{18}$$

The dynamic matrix of the model and Eqn (16) are combined to obtain:

$$s^T(Ax + bu + f) = -h_0^2(x)\,\text{sign}\,(s) \tag{19}$$

By knowing a bound h on the non-linearity for all conditions of x, the state described in Eqn (16) is satisfied by choosing the control input:

$$u = -(s^T b)^{-1} s^T Ax - \eta^2 (s^T b)^{-1}\,\text{sign}\,(s) \tag{20}$$

As mentioned previously, due to the uncertainties in modelling a UUV, it is important to recognise that the feedback law u is composed of two parts. The first:

$$\hat{u} = -(s^T b)^{-1} s^T Ax \tag{21}$$

is a linear feedback law based on the nominal model, whereas the second,

$$\bar{u} = -h^2 (s^T b)^{-1}\,\text{sign}\,(s) \tag{22}$$

is a non-linear feedback law, with its sign alternating between plus and minus according to which side of the sliding plane the system is currently located. Since u has to change its sign as the system crosses $\sigma(x) = 0$, the sliding surface has to be a hyperplane; that is, the sliding surface dimension has to be one less than the state space. u is also largely responsible for driving the system onto, and keeping it on, the sliding plane $\sigma(x) = 0$ (where $u = 0$ as well). Provided that the gain has been chosen sufficiently large, u can provide the robustness required to handle random disturbances and unmodelled dynamics without compromise. This is achieved by designing the linear feedback law to ensure that the system has the desired dynamics on the sliding plane.

Yoerger and Slotine[29] develop and apply a sliding mode controller to an ROV and document their simulation results on the experimental autonomous vehicle (EAVE). Although the authors reported successful implementation and control of EAVE they neglected the effects of pitch in their simulations, even though the heave and pitch channels are known to have quite influential cross-coupling effects.

Also Fossen,[30] and Fossen and Satagun[31] report the implementation of multivariable sliding mode controllers to the positioning of an ROV. Simulation results demonstrate the controller's ability to achieve robustness to parameter uncertainty. Healey and Lienard[32] have used this approach to control the speed, yaw and dive channels of an AUV individually. This work was then extended to develop a combined channel autopilot for the AUV. Results show robust performance for each of the individually controlled channels at low speed, and robust control in the combined autopilot for acceleration up to the chosen operational speed.

Trebi-Ollennu *et al.*[33] provide a review of four robust multivariable control designs, including input–output linearisation control with sliding mode depth control for an ROV. This technique is reported to result in a very robust controller but requires raw estimation of the bounds on the parametric uncertainties, which in itself is a non-trivial task.

2.6. *H-infinity (H_∞) robust control.* The UUV operating environment is varied; the speed of such a vehicle may vary, payloads may be increased or decreased and the underlying mathematical models are inherently uncertain. Classical and optimal types of controllers are designed around a specific set of environmental conditions; the performance consequently degrades as these factors vary.

Robust control addresses these problems. It guarantees, given actuator limitations, a minimum level of performance and stability for a specified operation envelope, not only in terms of disturbances which impinge on the

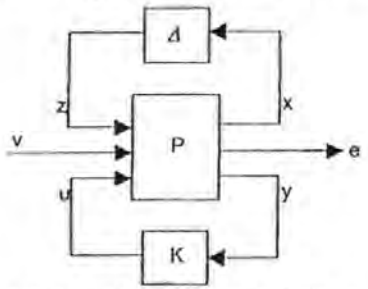


Fig. 8. H Infinity robust control scheme

mathematical representations of the UUV system. This method is embodied by the μ-synthesis procedure, essentially, an iterative process for the design of H_∞ controllers such that the closed loop adheres to the specified performance and stability criteria.

Given that G is a matrix representation of the plant and K is the matrix describing the controller, let the matrices T (the complementary sensitivity), S (sensitivity) and C (control sensitivity), be defined as in reference 34:

$$T = GK(I+GK)^{-1} \tag{23}$$

$$S = (I+GK)^{-1} \tag{24}$$

$$C = K(I+GK)^{-1}. \tag{25}$$

Figure 8 depicts the schematic control scheme, where u is the control signal, and v represents disturbance and noise inputs, y physical quantities, e error signals, and x and z the uncertainty inputs/outputs. P is the nominal plant and Δ the block-diagonal representations of uncertainty, environmental and mathematical.

If P is partitioned as shown in Eqn (26)

$$P = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} \tag{26}$$

then let M denote the closed-loop function mapping v to e; this is known as the lower fractional transformation (Eqn (27)):

$$M = P_{11} + P_{12}\,K(I-P_{22}\,K)^{-1}\,P_{21} = F(P, K) \tag{27}$$

The H_∞ optimisation problem is then to minimise Eqn (27) over all stabilising and realisable controllers, the constraints being defined, dependent upon engineering constraints, by weighting functions.[35] Provided that the following H_∞ norm inequalities are satisfied, then robust stability and performance are assured. Here γ is a search variable and the weightings are described below:

$$\|\gamma W_p S\|_\infty < 1 \tag{28}$$

$$\|\gamma W_d T\|_\infty < 1 \tag{29}$$

$$\|\gamma W_c C\|_\infty < 1 \tag{30}$$

where W_p is a weight matrix reflecting the frequency locations where the desired

within the mathematical models and, W_c depicts the restrictions on regions of operation of the servomechanisms.

Using the structured singular value, μ approach[36] a less conservative measure of robustness may be calculated. If the controller, K, is absorbed into the plant, P, and provided that Δ has a block-diagonal structure and is normalised, then partitioning

$$\begin{pmatrix} e \\ x \end{pmatrix} = Q \begin{pmatrix} v \\ z \end{pmatrix} = \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{bmatrix} \begin{pmatrix} v \\ z \end{pmatrix}. \tag{31}$$

Then for robustness, the μ is defined as Eqn (32), and must remain less than unity, that is,

$$\mu(Q_{11}(j\omega)) = \frac{1}{\min\,(\bar{\sigma}(\Delta(j\omega)),\ \det\,(I-Q_{11}(j\omega)\,\Delta(j\omega)) = 0)}. \tag{32}$$

White *et al.*[37] successfully applied the loop shaping H_∞ technique to the ROV depth control problem. Unlike other H_∞ control methods this technique concerns shaping the open loop dynamics of the ROV as opposed to the closed loop transfer function. Loop shaping was performed *via* two weighting functions, W_1 and W_2 which modified the open loop systems inputs and outputs respectively, thus achieving the desired loop shape for the system, namely; high gain at low frequencies, low gain at high frequencies, and some bandwidth and crossover frequency which yields desired gain and phase margins. Subsequent loop shaping designs emphasise the effects of different choices of W_1 and W_2 on the required closed loop design specifications. The resulting ROV controller's performance was assessed to variations in forward speed of ± 50 percent (5–15 knots) and perturbations in the pitch and heave coefficients of ± 20 percent.

The main difficulty with this application lies with finding a combination of weighting matrices that yields a controller which demanded inputs within the saturation limits of hydroplane and thruster actuators.

3. FUZZY LOGIC APPROACHES. Classical linear control system design methods are adequate to control linear systems but are often notably lacking in robustness when the system to be controlled exhibits characteristics of non-linearity, time dependence and extreme complexity. However, human operators still manage to control dynamical systems which display such characteristics. The emergence of fuzzy logic enabled the vagueness of human language to be mathematically quantified. Consequently, the control decisions of an experienced plant operator could be formulated into an algorithm to control the desired plant. Such an approach may therefore be capable of controlling an UUV very successfully.

3.1. *Generic structure.* When operating as a controller it uses fuzzy rules to interpret its input data and to generate an appropriate control output. Within the context of an UUV autopilot and its internal structure, the rules take the typical form:

If yaw error is positive small *and* yaw rate is positive big *then* rudder demand is zero.

Whilst fuzzy sets are usually illustrated as continuous functions, for implementation purposes they are often in a quantized form. By defining discrete universes of discourse for quantized error (e), quantized yaw rate (ce) and output (u) as E, CE and U respectively, such rules may be expressed as:

$$\text{If } E_i \text{ and } CE_i \text{ then } U_i$$

Where the fuzzy subsets E_i, CE_i and U_i are:

$$E_i = (e, \mu_{Ei}(e)) \subset E$$
$$CE_i = (ce, \mu_{CEi}(ce)) \subset CE$$
$$U_i = (u, \mu_{Ui}(u)) \subset U$$

and e, ce and u are elements of the appropriate discrete universes of discourse with membership functions of $\mu_{Ei}(e)$, $\mu_{CEi}(ce)$ and $\mu_{Ui}(u)$ respectively.

Thus, in general, the N rules contained within the algorithm of the fuzzy autopilot may be expressed as:

$$R_1: \text{If } E_1 \text{ and } CE_1 \text{ then } U_1 \text{ else}$$
$$R_2: \text{If } E_2 \text{ and } CE_2 \text{ then } U_2 \text{ else}$$
$$\cdots$$
$$R_N: \text{If } E_N \text{ and } CE_N \text{ then } U_N$$

which can be summarised in the fuzzy relation:

$$R = R_1 \cup R_2 \cup \cdots \cup R_N = \bigcup_{i=1}^{N} (E_i \times CE_i \times U_i). \tag{33}$$

As shown by Pedrycz,[38] Eqn (33) may be expressed in the following form:

$$R(e_i, ce_j, u_k) = \max [E_i(e_i) \vee CE_j(ce_j) \vee U_i(u_k)], \quad 1 \leqslant i \leqslant N. \tag{34}$$

To enable the fuzzy autopilot to operate for any given input use is made of the computational rule of inference:

$$U = (E \times CE) \circ R. \tag{35}$$

Thus, Eqn (35) may be written as:

$$U(u_k) = \bigvee_{ce_i \in C_E} [E(e_i) \wedge CE(ce_i) \wedge R(e_i, ce_j, U_k)]. \tag{36}$$

In order to elicit a deterministic value from the resultant fuzzy control output set, the centre of area method could be employed:

$$u_0 = \frac{\sum_{i=1}^{M} u_i U(u_i)}{\sum_{i=1}^{M} U(u_i)}. \tag{37}$$

Control systems which are based on fuzzy logic are extremely robust to [illegible] with the inherent non-linearities of the

plant under consideration. They, therefore, offer the potential to control UUVs in an effective manner.

3.2. *Fixed rule-based fuzzy control.* One particular implementation of fuzzy logic of interest here is the study conducted by Deffitetto,[39] who applied fuzzy logic to the depth and pitch control of an UUV. A fixed-rule base is used containing 14 rules for depth, pitch and ballast control. Good overall performance is achieved, although simulation results are obtained at a forward speed that is considered too slow for cross-coupling effects between the yaw and pitch channels to be influential. This approach does possess the advantage that rules can be easily modified as they take a linguistic form allowing an insight into the control strategy.

Smith *et al.*[40] applied fixed rule-based fuzzy controllers simultaneously to the channels yaw, heave and pitch of the *Ocean Voyager* AUV. Two problems were addressed: (i) the low level control problem of developing a control system which could reliably and efficiently manoeuvre the AUV, and (ii) the high level problem of docking the AUV in a confined space. Promising results were achieved using this approach although a more detailed analysis and comparison of the fuzzy controller to more conventional methods was not forthcoming.

3.3. *Fuzzy self-organising control.* In some circumstances it may be difficult to obtain a clear set of fuzzy rules which describe the controller action required, particularly in the case of non-linear, time-varying processes. To overcome this problem, fuzzy self-organising controllers (FSOC) have been developed,[41–46] which generate their own fuzzy rule-base by continual performance feedback, thus assessing the rule bases effectiveness. A schematic of a self-organising fuzzy controller is shown in Fig. 9.

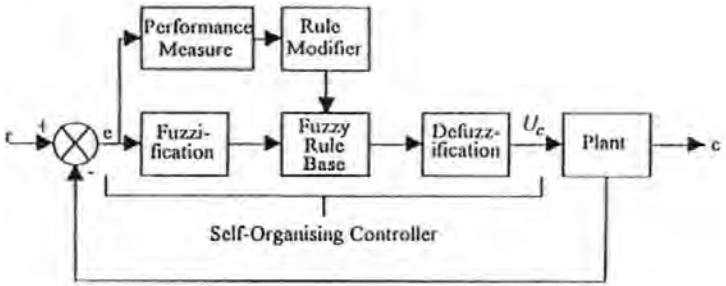


Fig. 9. The self-organising fuzzy controller

Essentially the architecture of a FSOC is similar to that of a fixed rule-based fuzzy controller but has the added refinement of a higher hierarchical level superimposed upon it. The hierarchical structure consists of three elements, namely, an appropriate performance index (PI), a simplified model of the plant and a rule modification algorithm.

The learning mechanism in this control system uses the values of error and change in error to initiate any improvements to the rules driven by the [illegible]

Assuming the plant is in an undesired state, the PI relates the measured values to an implied rule correction (dr) to the rule modification algorithm *via* the simplified plant model. In the case of a single-input, single-output system, the simplified model can be a sign change of unity, whereas for a multi-input, multi-output system, a matrix of its steady-state gains will suffice.

Clearly the above is a simplified description of a FSOC. A detailed exposition of a FSOC, adapted to form the basis of a ship autopilot, can be found in Sutton and Jess.[47] The actual fuzzy algorithm in the autopilot operates as described in the previous section with the exception that the compositional rule of inference is interpreted differently. Recalling Eqn (35) that is:

$$U = (E \times CE) \circ R. \tag{38}$$

As previously shown, normally '$\circ$' denotes the max–min product; however, Yamazaki[48] has found that better control responses result from using the max–max product. Thus, for this autopilot Eqn (36) is rewritten as:

$$U(u_k) = \bigvee_{e_i \in E,\, ce_j \in CE} [E(e_i) \wedge CE(ce_j) \vee R(e_i, ce_j, u_k)]. \tag{39}$$

Farbrother and Stacey[49] developed and applied a fuzzy logic fixed-rule base controller to the yaw channel of an ROV. This study provide encouraging results, but achieved limited success. This was due to the changing dynamics and external disturbances (such as noise) when applied to mine-counter measures. Encouraged by these simulations, and those of Sutton and Jess, Fabrother *et al.*[50] have developed and applied a self-organising fuzzy logic controller to the same problem. Consequently, the controller achieved a much more robust performance in the presence of external disturbances.

4. ARTIFICIAL NEURAL NETWORK METHODS. The artificial neural network (ANN) is a biologically inspired computing technique that, in its simplest form, is a fully connected structure of basic units which are themselves based on the McCulloch and Pitts[51] neuron model shown in Fig. 10. This model forms the

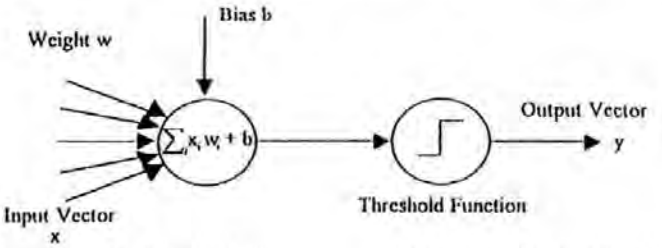


Fig. 10. The McCulloch and Pitts neuron model (after McCulloch and Pitts)[52]

basis for the early perceptron learning algorithm[52] and the later, and now widely known, backpropagation feedforward algorithm of Rumelhart and McClelland[53] for training the multilayer perceptron (MLP). With an ability to approximate non-

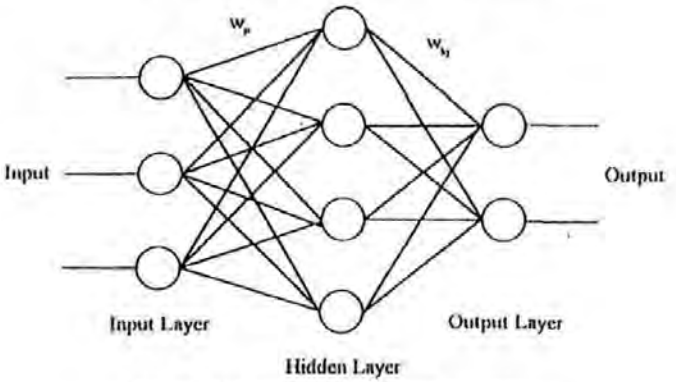


Fig. 11. The feedforward multilayer perceptron

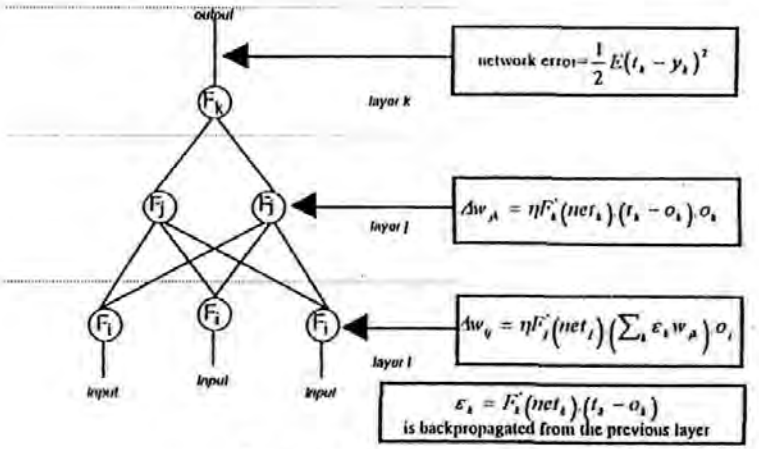


Fig. 12. Error backpropagation

classification and recognition problems,[54,55] for which more conventional techniques would have been less tractable. Figure 11 illustrates the form of the feedforward, fully connected multilayer perceptron.

Here the input vector $\mathbf{x}$ is mapped to the output vector $\mathbf{y}$ *via* the nodes in the hidden layer j and the weighted connections w_{ji} and w_{kj}:

$$\mathbf{y} = f(\mathbf{x}, w_{ji}, w_{kj}) \quad \mathbf{x} \in R^n, \mathbf{y} \in R^m. \tag{40}$$

There are three methods generally used in order to train neural networks, they

Supervised learning depends upon a target being available for each input pattern to compare to the actual output of the network. Reinforcement learning does not require a target output to be available but only a cost function signal to indicate whether changes in weight connections provide better or worse performance. Unsupervised learning is used when no target pattern and no cost function are easily available to perform training and requires that the network itself has the ability to recognise common features across the range of input patterns and modifies its internal state to model the features found in the training data.

One commonly used method of supervised learning is the aforementioned backpropagation (BP) rule. For each set of input data there is a corresponding output set, thus enabling the computation of an error measure between actual and desired output data sets on presentation of an input data set. The alteration of weights and biases within the ANN is therefore possible. The BP algorithm aims to alter the ANN weights and biases so that progression is made in the direction of the greatest rate of change of error reduction. To allow this behaviour, a function based upon the derivative of the error at the output of the previous layer is backpropagated through the ANN on completion of each training epoch or iteration. This principle is highlighted in Fig. 12. Further details can be found in reference 55.

4.1. *Applications.* Owing to the ability of ANNs to represent non-linear mappings of systems for which the underlying rules are unknown, they have been applied extensively to the control of marine vehicles. In the majority of applications to date, ANNs have been employed as a robust controller, where the network is generated through a period of learning and reinforcement. Usually the network is then frozen at this point because to perform on-line calculations generally requires a large amount of computer power. This problem has limited most controllers to date to be non-adaptable once a suitable level of control has been achieved.

Yuh,[56] in his paper on the application of an ANN controller for an AUV, describes the application of two and three layer architectures to the problem of trajectory control. The two-layered architecture is seen to provide a limited degree of success, proving unreliable in the advent of unknown vehicle dynamic situations and environmental uncertainties. A three-layered network, conversely, gave much more robust performances but was insufficiently documented as concerns simulation results to make satisfactory conclusions on its overall performance.

Waldock *et al.*[57] used a BP algorithm to train architectures of differing structure to control a UUV to follow the terrain of the sea bed. The chosen architecture, based on the smallest sum squared error after testing, was the single hidden layer three-neuron feedforward network. Initial training was then improved upon by using an alopex algorithm to find a minimum global error solution.

Johnson[58] performed a similar study again using differing architectures, but using a chemotaxis algorithm over the commonly employed BP algorithm. The better overall control than a classical PID controller with which it was compared in terms of reduced thruster revolutions.

An example of a neural network controller for an AUV which uses an on-line learning technique to update nodal weights has been designed and implemented by Venugopal *et al.*[59] A four-layered neural network is used which is trained by the BP algorithm. This implementation differs from many on-line strategies in that it employs a gain factor (proportional to the inverse of the Jacobian of the dynamics) which adapts to changes in the vehicle's dynamics in order that the controller accounts for dynamical change in its control action. Simulation results are divided into three distinct categories: (i) maintaining a desired pitch, (ii) maintaining a desired heading and (iii) maintaining a desired depth.

In case (i), the controller (beginning from a random start) soon achieves a good control action. Perturbations are introduced into the vehicle's forward speed dynamics to assess the performance of the adaptive gain network. Better simulation results were achieved when the learning rate of the network was increased, suggesting that the ANN was better equipped to generalise to disturbances in vehicle dynamics at a higher learning rate.

In case (ii), an increased learning rate provided faster convergence to the desired heading but also increased oscillatory behaviour about the desired heading. However, good simulation results were achieved and, as time increased, the on-line learning capabilities of the ANN controller reduced the oscillatory behaviour. The vehicle's parameters were again disturbed and it was noted that the higher learning rate again improved the controller's ability to adapt to varying dynamics.

Finally in case (iii), the controller achieved adequate control action even at a small learning rate.

It is obvious from this study that the learning rate is a critical feature of the ANN controller's performance in the event of varying vehicle dynamics. A criticism of this study is the lack of attention devoted to this important feature. For example, could the learning rate be increased incrementally, perhaps less initially to avoid oscillatory behaviour and then in larger increments as on-line learning improves controller performance in latter stages? Also the paper mentions the applicability of the chosen architecture and technique to MIMO controller design, but no simulations were even mentioned. Obviously it is a difficult task to choose an appropriate ANN learning rate for a MIMO controller which will optimise the learning rate of all three degrees of freedom simultaneously. However, this paper presents an alternative approach to the usual ANN AUV implementations and provides encouraging results for SISO control of the vehicle in the presence of varying vehicle dynamics.

Another interesting application of an ANN control strategy for a UUV is that given by Ishii *et al.*[60] Again this is an adaptive on-line control scheme called 'imaginary training' applied to a 'self-organising neural-net controller' (SONCS) which attempts to improve or shorten the training times involved in the learning phase of the control scheme. The SONCS consists of real world ANN and imaginary world ANN sections which are connected to form an adaptive link. In the real

imaginary world ANN section the imaginary training takes place based upon simulated state variables which are calculated without knowledge of actual simulation data. The imaginary training is executed independently of the actual operations of the AUV *via* an identification ANN. The controller of the real world part adjusts its network weights based on the adjusted values obtained in the imaginary world part. This ANN is chosen for its specific structure and consequent ability to identify dynamical systems *via* a forward model network.

To assess the effectiveness of the imaginary training, the proposed scheme was implemented on the *Twin-Burger* AUV for the yaw channel only. The resulting simulations show that the SONCS can control the heading of the AUV sufficiently well, even in the event of noise corruption in the teaching period. Also the controller shows good performance in the event of dynamical changes to the vehicle and its environment. One criticism proposed is the relatively infrequent sampling of the teaching data from the state variables for the adaptation of the weights in the imaginary training controller network. It is thought that a vehicle with a relatively small time constant such as an AUV will be very susceptible to environmental and dynamical peturbations and thus 20 s per sample of data is seen as a rather long time period in which to let the teaching of the real world controller weights remain 'untaught', especially in the initial stages of learning by the controller.

Yuh and Lakshmi[61] document a multilayered ANN controller which estimates the control error using a 'critic' or punish/reward system. The most appropriate architecture is again investigated in the initial stages of the paper. A three-layered ANN controller is chosen and compared using three learning algorithms: the BP algorithm, the parallel recursive prediction error algorithm (PRPE) and the modified parallel recursive prediction error algorithm (MPRPE).

The critic equation upon which controller network weights are adapted is a function of the actual velocity vector and the desired velocity vector, where the desired velocity vector is computed *via* the desired position vector, actual position vector and sampling period. The equation is then based upon a one-step performance measure of the vehicle position and velocity.

Case studies performed on the ANN controller were considered in the lateral plane; that is, in the yaw, surge and sway degrees of freedom. Initial tests investigated the performance of each learning algorithm whilst keeping the vehicle parameters constant. Resulting graphs indicate that the BP and MPRPE algorithms have good learning ability whilst the PRPE algorithm produces consistently high values of mean squared error in horizontal plane control. Results obtained using the MPRPE learning algorithm to train the ANN controller show that on-line training is effective and can adequately cope with the addition of random noise and also the effects of varying vehicle dynamics and parameters.

This study, although somewhat taking into account some of the cross-coupling effects of such a vehicle, does not provide any decoupled results in this plane against which to compare these simulations. Thus no comparisons can be made in terms of controller performance as concerns reduced effectiveness in any particular channel arising due to any cross-coupling effects.

research concerned with the unification of the two intelligent approaches of neural networks and fuzzy logic to produce intelligent neuro-fuzzy controllers. The aim of this union is to retain the generalisation, robustness and non-linear mapping ability of ANNs whilst allowing the utilisation of both linguistic and numerical data *via* the introduction of fuzzy logic. Thus a controller designed using a neuro-fuzzy approach has an immediate advantage over either an ANN or fuzzy logic controller.

The structure of neuro-fuzzy controllers is invariably such that no previous knowledge of the modelled process is required for the ANN to identify the existing input/output mapping. (It should be noted however that prior knowledge of the modelled process should improve the training time of such controllers.) It is thus possible that a controller can be designed which is applicable to other non-linear dynamic situations, even if knowledge of process dynamics is unknown.

Training of a neuro-fuzzy controller is usually based on one of two methods, either gradient descent as with the BP algorithm or reinforcement learning, although some applications have employed combinations of both methods. The hierarchical structure of the controllers means that the learning process usually consists of the ANN converging on an optimal set of fuzzy parameters, these parameters often taking a Sugeno form as in the work of Taylor.[7] Taylor[7] suggests that using an ANN to imitate a fuzzy controller and thus producing a 'black box' form of controller is inferior to the aforementioned method whereby knowledge is retained of process dynamics.

The adaptive network-based fuzzy inference system (ANFIS) was designed and implemented by Jang.[62] This approach uses an ANN which differs from most in that not all nodes are connected through weighted links. This has the consequence that not all the nodes are modifiable with respect to their weights. Jang has implemented the ANFIS in dynamic control of an inverted pendulum problem, whereby an approximation to the Jacobian matrix is used for backpropagation of the overall system error as opposed to the network error, thus creating a specialised learning approach to neuro-fuzzy control.

Jang and Sun[63] give a review of fundamental and advanced developments in neuro-fuzzy synergisms for modelling and control. In this review the basic concepts of fuzzy logic and adaptive networks are discussed as a prelude to discussions on the ANFIS architecture and its superiority over the BP algorithm. To conclude the review, a number of design techniques are given for neural and fuzzy controllers, and the common problems encountered in their implementation.

One such technique is 'BP through time and real time recurrent learning', a scheme whereby the controller and plant simulation blocks are replaced by two adaptive ANNs which cascade to form a large single network. The parameters to be adjusted in this process are the fuzzy controller parameters in the ANFIS. This BP through time approach has been employed by Nguyen and Widrow[64] to control a tractor-trailer vehicle in a confined zone operating environment.

Another implementation of neuro-fuzzy control by Jang and Gulley[65] is that

conditions so that a first-order Sugeno fuzzy model becomes a gain scheduler that switches between several sets of feedback gains. The MATLAB fuzzy logic toolbox contains Jang's implementation of this technique applied to the inverted pendulum system, where the scheduling variable is the pole length and the control action is the smooth switching between three sets of feedback gains.

A recent application of neuro-fuzzy control is given by Tao and Burkhardt,[66] who employ such a scheme to control a flame process. In order to control the supply of the input variables gas and oxygen a neural network-based fuzzy logic controller is implemented through a personal computer. The optimal burning state is hoped to be achieved by the fine-tuning of the fuzzy parameters. Prior expert knowledge is incorporated into the control action through the fuzzy logic. Two methods of network training were adopted; supervised learning and reinforcement learning. Supervised learning was employed in one instance when it was assumed that training data were available; reinforcement learning was otherwise used. This paper provides the reader with an alternative example of an application of neuro-fuzzy control. Although no results are presented the system is said to be able to control the flame to find its optimum state.

Taylor[7] provides a review of neural fuzzy algorithms for control applications. This work highlights the clear distinction between two classes of learning algorithm, those of supervised learning and those of unsupervised or reinforcement learning.

Jang[67] has successfully applied the more computationally inexpensive temporal difference methods of Sutton[68] in conjunction with his own ANFIS architecture to control a specified pole balancing on a moving cart problem. This approach falls into the supervised learning category due to the use of temporal backpropagation. Thus, as such, the method relies on *a priori* knowledge of the underlying model of the pole problem. This approach is seen to be capable of balancing the pole on a cart which moves unrestrictedly after only one controller parameter set adjustment, the parameter antecedents and consequents being initially set to zero and to cover the input space respectively. The controller was also seen to be robust to variations in pole lengths and initial conditions.

Barto and Anderson[69] have applied a reinforcement algorithm to the balancing pole problem, whereby the reinforcements are used to update the weights of an ANN, the action selection network (ASN) which has a partly connected architecture and is derived from a set of fuzzy conditional statements. An action evaluation network (AEN) is also used to generate a reinforcement signal based on the effectiveness of the previous control action. The learning aim of the ASN is to increase the output from the AEN, thus providing more improvements in control action. This controller required 13 rules in its fuzzy rule base to balance the pole. Results showed good robustness to rule omission or degradation. This method also requires a model for off-line training or that a failure signal is generated, in on-line training, by the plant.

Albus[70] designed and implemented the cerebellar model articulation controller (CMAC) architecture, an associative ANN which employs piecewise 'constant' basis functions, the number of which is determined by the designer *via* a

fuse together these basis functions to cover the input space. The network output can consequently be written as a function of the sum of the individual weights of the network which are determined from a supervised training scheme such as a least mean squares law.

If the CMAC network is used to interpret a fuzzy controller, as in Pedrycz,[38] the network consists of the following layers:

(i) A sensory layer – provided with inputs of fuzzy sets 'error' and 'change in error'.
(ii) An association layer – this consists of logical AND nodes and is used to aggregate the individual signals of the Sensory Layer.
(iii) A post association layer – this consists of logical OR neurons used to summarise the AND aggregates above.
(iv) A defuzzification layer – transforming the results of all three above layers into one single valued output.

The learning scheme is by reinforcement and a single scalar value is used to determine a group of connections in the network. Also the amount of *a priori* knowledge necessary to construct or design the controller is reduced.

An alternative neural fuzzy control strategy, the differential competitive learning (DCL) algorithm, was proposed by Kosko[71] and co-workers,[72,73] whereby individual neurons in the network architecture are in competition with each other. Fuzzy rules containing multiple antecedents are decomposed and then reformulated to produce more fuzzy rules which can be considered as the unions of individual decomposed rules, where fuzzy sets are defined as quantisation vectors of membership function values. These are termed fuzzy associative memories (FAM) and combine to produce the fuzzy inference system, each FAM representing a particular area of the input/output space. The aim of DCL is to cluster the quantisation vectors and consequently generate fuzzy rules. The NN architecture has an input layer and a 'competition layer' connected together by a weight matrix and an intra-weight matrix in the competition layer which has positive diagonal elements and negative non-diagonal elements to excite the neurons. If a neuron becomes unexcited then learning will occur, caused by the weight changes' dependence on the change in competing neurons output. Thus, many fuzzy rules may be defined but only those having a number of quantisation vectors assigned to them are used. Kosko[71] demonstrates successful controller performance using the DCL method which is not reliant on *a priori* knowledge of the plant model.

Future applications of neuro-fuzzy techniques to the control of marine vehicles are expected, mainly due to the fact that such techniques do not require a model of the process dynamics to produce a control action.

Sutton *et al.*[74] have investigated the use of ANNs in the design of fuzzy autopilots for controlling the yaw dynamics of a modern Royal Navy Warship model. A network is chosen based on the ANFIS network and is trained using the BP, alopex,[75] chemotaxis[76] and simulated annealing[77] algorithms. The input fuzzy sets for the autopilot are chosen as yaw error and yaw rate. Simulation results are [illegible] conventional multilinear autopilot. Overall, it was

found that the autopilot trained using the simulated annealing algorithm performed the best with respect to overshoot, rise time and the integral square error of rudder angle and yaw error.

6. CONCLUDING REMARKS. It has been shown that when attempting to design a controller for a UUV, the control engineer is faced with several difficult problems. Thus, whilst many of the present generation of control systems installed within UUVs perform satisfactorily within given specifications, their overall effectiveness is limited.

Traditionally, control system designs have been variants of the analogue PID controller. The main shortcoming of such designs has been the requirement for manual adjustment of the controller's parameters to compensate for changes in the craft's environment, but these settings are rarely optimal for the UUV. The adjustments necessary for current and payload variations are time consuming. Consequently, there has been a growth of interest in autopilots which can automatically adapt themselves.

Adaptive control techniques for autopilot designs have, *via* the use of suitable cost functions in the optimisation algorithms, enabled more efficient use of vehicle actuators in the event of environmental changes and varying vehicle dynamics. Due to the use of a cost function, usually minimising the rudder activity and heading error when performing yaw changing manoeuvres, adaptive strategies do not always reduce the fuel consumption of the vehicle in all sea states. This inherent inability to measure the sea states effect upon the vehicle suggests that more advanced techniques must be developed for UUV autopilots. H_∞ has enabled the design of optimal controllers in the presence of significant uncertainties within the UUV model without the need for on-line identification of the vehicle's dynamics, and these controllers have been shown to be robust in operation.

Fuzzy logic control systems are inherently robust to non-linear time varying plant but remain reliant upon a rule base. Indeed, the self-organising fuzzy logic controller develops its own rule base but requires some initial performance criterion. Such approaches have proved to be very successful at controlling UUVs.

The fusion of fuzzy logic and neural network control methodologies offers a means by which the inherently robust and non-linear nature of the fuzzy controller can be combined with the powerful learning abilities of the neural network. Although there are examples of such fusions as applied to ship autopilot designs, little attention has been given to the design of UUV autopilots using these techniques. Consequently, the use of neuro-fuzzy approaches to control the dynamic behaviour of UUVs could offer significant technological advances in the field of UUV autopilot design and thus provide an excellent research area.

REFERENCES

[1] Coverdale, A. and Cassidy, S. (1987). The use of scrubbers in submarines. *Journal of Naval Engineering*, pp. 528–544.

[2] Reader, G. T., Walker, G. and Hawley, J. G. (1989). Non-nuclear powerplants for AUVs. *Proceedings of the 6th International Symposium on Untethered Submersible Technology*, University of New Hampshire, pp. 88–99.

[3] Golten, J. and Verwer, A. (1991). *Control System Design and Simulation*. McGraw Hill, New York.

[4] Ogata, K. (1990). *Modern Control Engineering*, 2nd ed. Prentice-Hall International Ltd.

[5] Kuo, B. (1987). *Automatic Control Systems*, 5th ed. Prentice-Hall International Editions.

[6] Motora, S. (1953). On the automatic steering and yawing of ships in rough seas. *Journal of the Society of Naval Architects*, Japan. Volume 94.

[7] Taylor, S. D. H. (1995). The design and tuning of fuzzy autopilots using artificial neural networks. *M.Phil. thesis*, University of Plymouth/RNEC.

[8] Da Cunha, J. P. V. S., da Costa, R. R. and Hsu, L. (1991). Design of a high-performance variable structure position control of ROVs. *ISOPE 91, Proceedings of the First Offshore and Polar Engineering Conference*, Edinburgh, UK.

[9] Hsu, L., Costa, R. R., Lizarralde, F. C., Da Cunha, J. P. V. S., Scieszko, J. L., Romanov, A. V., Junior, D. W. and Sant'anna, A. C. (1994). Underwater vehicle dynamic positioning based on a passive arm measurement system. *Proceedings of the Second International Advanced Robotics Programme Workshop on Mobile Robots for Subsea Environments*, Monterey.

[10] Egeskov, P., Bjerrum, A., Pascoal, A., Silvestre, C., Aage, C. and Wagner Smitt, L. (1994). Design, construction and hydrodynamic testing of the AUV *MARIUS*, *Proceedings of the 1994 AUV Conference*, Cambridge, Massachusetts.

[11] Kaminar, I., Pascoal, A. M., Khargonekar, P. P. and Thomson, C. (1993). A velocity algorithm for the implementation of gain-scheduled nonlinear controllers. *Proceedings of the Second European Control Conference*, pp. 787–792.

[12] Harris, C. J. and Billings, S. A. (1981). *Self-tuning and Adaptive Control*. IEE Engineering Series 15. Peter Peregrinus Press Ltd, UK and New York.

[13] Astrom, K. J. and Wittenmark, B. (1989). *Adaptive Control*. Addison-Wesley Publishing Company, Boston.

[14] Van Amerongen, J. (1982). Adaptive steering of ships: a model reference approach to improved manoeuvring and economical course-keeping. *Ph.D. thesis*, Delft University of Technology, Holland.

[15] Shimmin, D. W. and Lucas, J. (1993). Control strategies for underwater vehicle autopilots. *IEE Colloquium on Control and Guidance of Underwater Vehicles*. Digest no. 1993/232.

[16] Farrell, J. and Clauberg, B. (1993). Issues in the implementation of an indirect adaptive control system. *IEEE Journal of Oceanic Engineering* 18 (3).

[17] Boulton, D. B. (1985). The development of multi-variable control algorithms for a dynamically positioned ship. *Proceedings of the IEE Control 85*, Cambridge, UK, pp. 151–156.

[18] Triantafyllou, M. S. and Grosenbaugh, M. A. (1991). Robust control for underwater vehicle systems with time delays. *IEEE Journal of Oceanic Engineering* 16 (1), 146–151.

[19] Owens, D. H. and Chotai, A. (1983). Robust controller design for linear dynamic systems using approximate models. *Proceedings of the IEE, Part D* 130 (2).

[20] Katebi, M. R. and Byrne, J. C. (1988). LQG adaptive ship autopilot. *Transactions of the Institute of Measurement and Control* 10 (4), 187–191.

[21] Yuh, J., Fox, J. S. and Lakshmi, R. (1990). Control and optimal sensing in underwater robotic vehicles (URVs). *Proceedings of the IEEE Oceans 90*, Washington, D.C., USA, pp. 88–93.

[22] Goheen, K. R. and Jeffreys, E. R. (1989). Hardware and software considerations for a self-tuning controller. *Proceedings of the DOI*, Brighton, UK.

[23] Clarke, D. W. and Gawthorp, P. J. (1975). Self-tuning controller. *Proceedings of the IEE, Part D* 122 (9), 929–934.

[24] Lim, C. C. and Forsythe, W. (1983). Autopilot for ship control. *Proceedings of the IEE, Part D* 130 (6), 281–294.

[25] Goheen, K. R. and Jeffreys, E. R. (1990). Multivariable self-tuning autopilots for autonomous and remotely operated vehicles. *IEEE Journal of Oceanic Engineering* 15 (3).

[26] Cristi, R., Papoulias, F. A. and Healey, A. J. (1990). Adaptive sliding mode control of autonomous underwater vehicles in the dive plane. *IEEE Journal of Oceanic Engineering* 15 (3).

[27] Slotine, J. J. E. (1983). Tracking control of non-linear systems using sliding surfaces. *Ph.D. thesis*, Department of Aeronautics and Astronautics, MIT, Cambridge, MA.

[28] Slotine, J. J. E. (1985). The robust control of robot manipulators. *International Journal of Robotics Research* 4 (2), 49–64.
[29] Yoerger, D. R. and Slotine, J. J. (1985). Robust trajectory control of underwater vehicles. *IEEE Journal of Oceanic Engineering* 10 (4).
[30] Fossen, T. I. (1991). Non-linear modelling and control of underwater vehicles. *Dr. Ing. thesis*, Norwegian Institute of Technology, Trondheim.
[31] Fossen, T. I. and Satungen, S. (1991). Adaptive control of non-linear systems: a case study of underwater robotic systems. *Journal of Robotic Systems* 8, 393–412.
[32] Healey, A. J. and Lienard, D. (1993). Multivariable sliding mode control for diving and steering of unmanned underwater vehicles. *IEEE Journal of Oceanic Engineering* 18 (3).
[33] Trebi-Ollennu, A., King, J. and White, B. A. (1995). A study of robust multivariable control designs for remotely operated vehicle depth control systems. *IEE Colloquium on Control and Guidance of ROVs*, Digest no. 124.
[34] Sharif, M. T., Roberts, G. N. and Sutton, R. (1996). Final experimental results of full-scale fin/rudder roll stabilisation sea trials. *Control Engineering Practice* 4 (3), 377–384.
[35] Maciejowski, J. M. (1989). *Multivariable Feedback Design*. Addison-Wesley.
[36] Doyle, J. C. (1982). Performance and robustness analysis for structured uncertainty. *Proceedings of the IEEE, Decision and Control*.
[37] White, B. A., Stacey, B. A., Patel, Y. and Bingham, C. (1994). Robust control design of an ROV. *Technical Report*, Contract No. DRA/PD/107/94. The Royal Military College of Science, Cranfield University.
[38] Pedrycz, W. (1993). *Fuzzy Control and Fuzzy Systems*, 2nd ed. Research Studies Press, Taunton, UK.
[39] Dellitetto, P. (1995). Fuzzy logic for depth control of UUVs. *IEEE Journal of Oceanic Engineering* 20 (1).
[40] Smith, S. M., Rae, G. J. S., Anderson, D. T. and Shein, A. M. (1993). Fuzzy logic control of an autonomous underwater vehicle. *Proceedings of the First International Workshop on Autonomous Underwater Vehicles*, Southampton, UK, pp. 318–323.
[41] Shao, S. (1988). Fuzzy self-organising controller and its application for dynamic processes. *Journal of Fuzzy Sets and Systems* 26, 151–164.
[42] Daley, S. and Gill, K. F. (1986). A design study of a self-organising fuzzy logic controller. *Proceedings of the Institution of Mechanical Engineers*, Part C, 1, 200, 59–69.
[43] Mandic, N. J., Scharf, E. M. and Mamdani, E. H. (1985). Practical application of a heuristic fuzzy rule-based controller to the dynamic control of a robot arm. *IEE Proceedings, Part D* 132 (4), 190–201.
[44] Tanscheit, R. and Scharf, E. M. (1988). Experiments with the use of a rule-based self-organising controller for robotics applications. *Fuzzy Sets and Systems* 26, 195–214.
[45] Procyk, T. J. and Mamdani, E. H. (1979). A linguistic self-organising process controller. *Automatica* 15, 15–30.
[46] Farbrother, H. N. R. (1991). Aspects of remotely operated vehicle control – a review. *RNEC Control Department Research Report*, RNEC-RR-91025.
[47] Sutton, R. and Jess, I. M. (1991). A design study of a self-organising fuzzy autopilot for ship control. *Proceedings of the IMechE, Part I* 205, 35–47.
[48] Yamakazi, T. (1982). An improved algorithm for a self-organising controller. Ph.D. thesis, University of London.
[49] Farbrother, H. N. R. and Stacey, B. A. (1990). Fuzzy logic control of an ROV. *Modelling and Control of Marine Craft* (Pomzaujani, M. and Roberts, G., eds.), pp. 193–210.
[50] Farbrother, H. N. R., Stacey, B. A. and Sutton, R. (1991). A self-organising controller for an ROV. *Proceedings of IEE Control 91*, Edinburgh, pp. 499–504.
[51] McCulloch, W. S. and Pitts, W. (1943). A logical calculus of the ideas imminent in nervous activity. *Bulletin of Mathematical Biophysics* 5, 115–133.
[52] Rosenblatt, F. (1962). *Principles of Neurodynamics*. Sparten Books, New York.
[53] Rumelhart, D. E. and McClelland, J. L. (1986). *Parallel Distributed Processing: Explorations*

[54] Sejnowski, T. and Rosenberg, C. R. (1987). Parallel networks that learn to pronounce English text. *Complex Systems*, pp. 145–168.
[55] Beale, R. and Jackson, T. (1990). *Neural Computing: An Introduction*. IOP Publishing Ltd, Bristol, UK.
[56] Yuh, J. (1990). A neural network controller for underwater robotic vehicles. *IEEE Journal of Oceanic Engineering* 15 (3), 161–166.
[57] Waldock, M. I., Roberts, G. N. and Sutton, R. (1995). Terrain following control of an unmanned underwater vehicle using artificial neural networks. *IEE Colloquium on Control and Guidance of Remotely Operated Vehicles*, London.
[58] Johnson, C. (1995). Depth and yaw control of an unmanned underwater vehicle using neural networks. *M.Phil. thesis*, University of Plymouth.
[59] Venugopal, K. P., Sudhakar, R. and Pandya, A. S. (1992). On-line learning control of autonomous underwater vehicles using feedforward neural networks. *IEEE Journal of Oceanic Engineering* 17 (4).
[60] Ishii, K., Fujii, T. and Ura, T. (1993). An on-line adaption method in a neural network based control system for AUVs. *IEEE Journal of Oceanic Engineering* 20 (3).
[61] Yuh, J. and Lakshmi, R. (1993). An intelligent control system for remotely operated vehicles. *IEEE Journal of Oceanic Engineering* 18 (1).
[62] Jang, J. S. R. (1993). ANFIS: adaptive network-based fuzzy inference system. *IEEE Transactions on Systems, Man and Cybernetics* 23, 665–685.
[63] Jang, J. S. R. and Sun, C. T. (1995). Neuro-fuzzy modelling and control. *Proceedings of the IEEE* 83 (3).
[64] Nguyen, D. H. and Widrow, B. (1990). Neural networks for self learning control systems. *IEEE Control Systems Magazine*, pp. 18–23.
[65] Jang, J. S. R. and Gulley, N. (1995). *The Fuzzy Logic Toolbox for Use with Matlab*. The Mathworks Inc., Natick, MA, USA.
[66] Tao, W. and Burkhardt, H. (1994). Application of fuzzy logic and neural network to the control of a flame process. *IEE Intelligent Systems Engineering*, publication no. 395.
[67] Jang, J. S. R. (1993). *Self-learning Fuzzy Controllers Based on Temporal Backpropagation*. Department of Electrical Engineering and Computer Science, MIT, Cambridge, MA, USA.
[68] Sutton, R. S. (1988). Learning to predict by the methods of temporal differences. *Machine learning*, vol. 3, pp. 9–44. Kluwer Academic Publishers, Dordrecht.
[69] Barto, A. G. and Anderson, C. W. (1985). Structural learning in connectionist systems. *Proceedings of the Seventh Conference of the Cognitive Science Society*, pp. 43–53.
[70] Albus, J. S. (1975). A new approach to manipulator control: the cerebellar model articulation controller (CMAC). *Transactions of the ASME, Journal of Dynamic Systems, Measurement, and Control*, pp. 220–227.
[71] Kosko, B. (1992). *Neural Networks and Fuzzy Systems*. Prentice Hall International.
[72] Pacini, P. J. and Kosko, B. (1992). Adaptive fuzzy systems for target tracking. *Intelligent Systems Engineering*, pp. 3–21.
[73] Kong, S. G. and Kosko, B. (1992). Adaptive fuzzy systems. *IEEE Transactions on Neural Networks* 3, 2–9.
[74] Sutton, R., Taylor, S. D. H. and Roberts, G. N. (1996). Neuro-fuzzy techniques applied to a ship autopilot design. This *Journal* 49, 410.
[75] Unnikrishnan, K. P. and Venugopal, K. P. (1994). *Alopex: A Correlation-Based Learning Algorithm for Feedforward and Recurrent Neural Networks*, downloaded from the Joint Academic Network.
[76] Koshland, D. E. (1980). Bacterial chemotaxis in relation to neurobiology. *Annual Review of Neuroscience* 3, 43–75.
[77] Kirkpatrick, S., Gelatt, C. and Vecchi, M. (1983). Optimization by simulated annealing. *Science* 220, 671–680.

Paul J. CRAVEN*
Mirosław KWIESIELEWICZ**
Robert SUTTON*

IDENTIFICATION OF UNDERWATER VEHICLE INVERSE MODEL USING RECURRENT NEURAL NETWORK AND GENETIC ALGORITHMS***

1. Introduction

This paper concerns a control of autonomous underwater vehicle yawing with a controller of a neural network structure. In general in order to adjust neural controller parameters two sets of training data are necessary: controller input trajectories set and controller output trajectories set over a given time interval. Whilst the set of input trajectories can be obtained rather easy, the set of output trajectories should be often evaluated basing on a set of a control system output trajectories, in our case a set of yaw responses over a given time interval [4]. To evaluate the controller output trajectories set we propose the inverse model approach based on the continuos recurrent neural network which can be viewed as a set of non-linear state equations.

Recent investigations concerning underwater vehicle inverse model identification using the recurrent neural network show [3,4] that the backward propagation approach applied to adjust the network parameters does not give satisfactory results due to trapping into local minima and the algorithm instabilities. To overcome these problems a genetic algorithm is used.

All calculation concerning this study were made using the MATLAB/Simulink environment.

2. Underwater vehicle yaw control structure

In the study the yawing of the autonomous underwater vehicle (AUV) with a neural network controller is considered (Fig. 1).

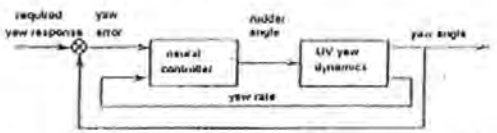


Fig. 1. The control system structure with the neural network controller
Rys. 1. Struktura układu sterowania z regulatorem neuronowym

* University of Plymouth, Institute of Marine Studies, Drake Circus, Plymouth PL4 8AA, Devon

The yaw dynamics is described by a set of non-linear state equations and implemented in MATLAB/Simulink environment, supplied by the Defence Research Agency (DRA), Sea Sector, Winfrith [1,2].

3. Inverse model of the auv yaw dynamics

In order to identify the AUV inverse model a dynamic neural network is proposed which can be described by the following system of coupled differential equations [5,6,7]:

$$T_i \frac{dx_i}{dt} = -x_i + \sigma(s_i) + u_i, \quad i = 1,\ldots,n, \tag{1}$$

$$s_i = \sum_{j=1}^{n} w_{ij} x_j, \quad i = 1,\ldots,n, \tag{2}$$

where n is the number of dynamic neurons, x_i is the state of the ith dynamic neuron, s_i is the total input to the ith unit, T_i is the time constant of unit i, w_{ij} are weights, u_i are inputs to the system and σ is an arbitrary differentiable function, e.g. a sigmoidal function: $\sigma(\xi) = (1 - e^{-\xi})^{-1}$.

Such a network can easily be implemented in the MATLAB environment. Training of the network (1)-(2) leads to weight and time constant adjustments. The inverse model tuning procedure is shown in Fig. 2.

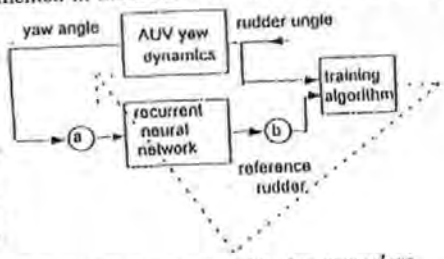


Fig. 2. The inverse model tuning procedure
Rys. 2 Algorytm strojenia modelu odwrotnego

There are three main procedures to train the network (1)-(2): forward/backward technique [5], backpropagation through time [8] and forward propagation [8]. Only the forward/backward technique can be implemented directly in the MATLAB environment, however recent investigation show that this technique does not give satisfactory results due to trapping int local minima and algorithm instabilities [3,4]. Therefore a genetic algorithm approa is assumed.

4. Numerical results

For the inverse model identification of the AUV yaw dynamics the recurrent neu network (1)-(2) with 10 dynamics neurons was assumed. A genetic algorithm was us to minimise the energy function:

$$E = \frac{1}{2}\int_{t_0}^{t_k} \left(y(t) - d(t)\right)^2 dt,$$

where y is the actual and d is the desired trajectory of the network over the time interval $[t_0, t_1]$.

Input and output trajectories were collected over the time interval [0,100] seconds basing on the AUV yaw dynamics implemented in MATLAB/Simulink environment, however for the inverse model identification only the trajectories over the time interval [0,20] seconds were used. The numerical results for one input-output data set are shown in Fig. 3.

After 160 genetic generations a value of the energy function was decreased to 0.025.

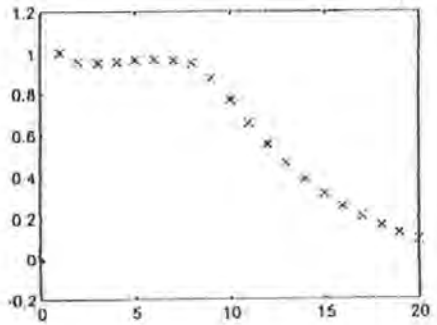


Fig. 3. Numerical results for the inverse model
- rudder reference trajectory
x rudder actual trajectory

Rys. 3. Wyniki obliczeń dla modelu odwrotnego
- trajektoria odniesienia dla steru
x aktualna trajektoria dla steru

5. Remarks and Comments

In neural network based control systems a main difficulty arises in providing a training data set for a desired controller output. In practice sometimes it is easier to give desired trajectories for the control system output. In this paper the inverse model approach was recalled in order to evaluate a controller output trajectory referred to a control system output trajectory, basing on the warship yaw dynamics inverse model in form of a continuous recurrent neural network. Recent approaches to the problem of the inverse model identification [3,4] using such kind of the network have shown that the application of the gradient method [8,9] led to trapping in local minima and causing the algorithm instabilities. To overcome these difficulties the genetic based method was used for the network parameters tuning. Numerical results show that it is possible to obtain the inverse model of the warship dynamics using this approach. The method proposed can be applied for any plant with a neural network controller.

References

1. Cowling D., Corfield, S.J,: *Control functions for autonomous underwater vehicles on-board command and control systems*. IEE Colloquium on Control and Guidance of Remotely Operated Vehicles, London, June 1995: 3/1-3/8.
2. Cowling, D.: *Full range autopilot design for an unmanned underwater vehicle*. Proceedings of 13th IFAC World Congress, Vol. Q, San Francisco, USA, July 1996: 339 -344.
3. Craven P.J., Kwiesielewicz M. and Sutton R.: *Non-linear system identification using a recurrent neural network*. Proceedings of 16th IASTED International Conference 'MODELLING, IDENTIFICATION AND CONTROL', February 17 -19th, Innsbruck, Austria, 1997, pap. No. 258-161.
4. Craven P.J., Kwiesielewicz M. and Sutton R.: *Dynamic system control using ANFIS architecture based controller*. Materiały XV Ogólnopolskiej Konferencji „Polioptymalizacja i Komputerowe Wspomaganie Projektowania":69-76.
5. Pearlmutter B. A.: *Learning state space trajectories in recurrent neural networks*. Tech. Report. CMU-CS-88-191, School of Computing Science, Carnegie Mellon University, Pittsburgh, USA, 1988.
6. Pearlmutter B. A.: *Learning state space trajectories in recurrent neural networks*. Neural Computation 1, 1989: 263-269.
7. Pineda F.: *Generalization of backpropagation to recurrent neural networks*. Physical Review Letters, Vol. 19, No. 59, 1987: 2229-2232.
8. Zbikowski R., Gawthrop P.J.: *A survey of neural networks for control*. in K. Warwik, G.W. Irwin and K.J. Hunt, Neural Networks for Control Systems, IEE Control Engineering Series, 46, Peter Peregrinus, 1992: 31-50.

Summary

The paper concerns an autonomous underwater vehicle yawing control with a neural network controller. In order to evaluate a reference controller output an inverse model approach based on a continuous recurrent neural network is used. A genetic algorithm is applied to tune the network parameters. Numerical results are included.

IDENTYFIKACJA MODELU ODWROTNEGO POJAZDU PODWODNEGO Z WYKORZYSTANIEM RECURENCYJNEJ SIECI NEURONOWEJ I ALGORYTMÓW GENETYCZNYCH

Streszczenie

Rozważa się zagadnienie sterowania zmianą kursu dla autonomicznego pojazdu podwodnego z wykorzystaniem regulatora neuronowego. W celu obliczenia wzorcowego sygnału sterującego stosuje się podejście oparte o model odwrotny obiektu, w postaci ciągłej rekurencyjnej sieci neuronowej. Do strojenia parametrów tej sieci stosuje się algorytm genetyczny. Załącza się wyniki obliczeń.

Referat recenzowany

A NEURAL NETWORK BASED FUZZY AUTOPILOT DESIGN FOR AN AUTONOMOUS UNDERWATER VEHICLE

Paul J Craven and Robert Sutton

Institute of Marine Studies, University of Plymouth, Drake Circus, Plymouth, PL4 8AA, UK
(E-mail : pcraven@plymouth.ac.uk)

Abstract: This paper describes the application of a neural network approach to the tuning of a fuzzy autopilot for course-changing control of an autonomous underwater vehicle (AUV). The autopilot is encoded as an adaptive network-based architecture. To describe the yaw dynamics of the AUV a sophisticated simulation model is employed. Results are presented which highlight the course-changing ability and robustness of the tuned autopilot over the same autopilot prior to neurofuzzy tuning. It is concluded that the chosen tuning method offers a viable approach to the development of effective AUV yaw autopilots.

Keywords: neural, tuning, fuzzy, autopilot, autonomous underwater vehicle.

1. INTRODUCTION

The running costs of manned submersibles and support ship platforms for remotely operated vehicles are becoming increasingly high. As a consequence in order to reduce financial overheads, considerable interest is being shown in the development and construction of autonomous underwater vehicles (AUVs) to undertake such tasks as ocean surveying for geological and biological purposes, pipeline inspection, explosive ordnance disposal and covert surveillance. In order for this type of vehicle to be truly autonomous, it is necessary for it to possess a reliable and robust onboard navigation, guidance and control (NGC) system. A key element of the NGC system is the control subsystem which is responsible for maintaining the vehicle on course. Several advanced control engineering concepts including H_∞, sliding mode, adaptive and generalized predictive control theories are being employed in the design of course-changing autopilots and have met with varying degrees of success.

Artificial intelligence (AI) approaches are now also being introduced into the design process. Autopilots formulated using fuzzy logic and artificial neural network (ANN) methods have been reported and shown to be endowed with commendable robustness properties. Encouraged by such results, this paper considers the development of a course-changing autopilot based on the innovative neurofuzzy methodology known as the adaptive network-based fuzzy inference system (ANFIS), (Jang, 1993).

With the ANFIS approach implementation of the controller design differs in form from the more typical ANN in that it is not fully connected, and not all the weights or nodal parameters are modifiable. Essentially the fuzzy rule base is encoded in a parallel fashion so that all the rules are activated simultaneously so as to allow network training algorithms to be applied. As in Jang's original work, a backpropagation algorithm is used to tune the fuzzy sets of the antecedents in the fuzzy rule base and a recursive least squares algorithm tunes the consequent coefficients with respect to stipulated input-output training data. For performance assessment purposes, the evolved ANFIS autopilot is compared to the initial untuned fuzzy autopilot.

To describe the yaw dynamics of a UUV use is made of a MATLAB/Simulink model supplied by the Defence Evaluation and Research Agency (DERA), Sea Systems Sector, Winfrith. The model having been validated against standard DERA non-linear hydrodynamic code using tank test data and an experimentally derived set of hydrodynamic coefficients from the Southampton Oceanography Centre's AUTOSUB vehicle. Details of the AUV simulation package can be found in Craven *et al.* (1997).

2. NEUROFUZZY AUTOPILOT DESIGN

As mentioned above, the fuzzy controller tuning performed within this work is based on the ANFIS. Functionally, there are almost no constraints on the membership functions of an adaptive network except piecewise differentiability.

Structurally, the only limitation on network configuration is that it should be of feed-forward type. Due to these minimal restrictions, the adaptive network's applications are immediate and immense in various areas.

If it is assumed that the fuzzy inference system under consideration has multiple inputs and one functional output (f) then the fuzzy rule-based algorithm may be represented in the first order Sugeno form as shown below:

Rule 1:If x is A_1 and y is B_1 then $f_1 = p_1 x + q_1 y + r_1$
Rule 2:If x is A_2 and y is B_2 then $f_2 = p_2 x + q_2 y + r_2$
⋮
Rule n:If x is A_n and y is B_n then $f_n = p_n x + q_n y + r_n$

By encoding such a fuzzy rulebase as an adaptive network structure the resulting ANFIS architecture can be taken as that of Figure 1. Consequently, node functions in the same layer are of the same function family as described by the following:

Layer 1: Every *ith* node in this layer is an adaptive node with a node output defined by:

$$O_{1,i} = \mu_{A_i}(x) \qquad (1)$$

where x is the input to the general node and A_i is the fuzzy set associated with this node. In other words, outputs of this layer are the membership values of the premise part. Here the membership functions for A_i can be any appropriate parameterized membership functions. Here A_i is characterized by the generalized bell function of equation (2), where $\{a_i, b_i, c_i\}$ is the parameter set. Parameters in this layer are referred to as *premise parameters*.

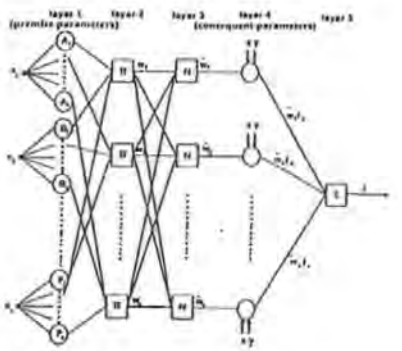


Fig. 1. The adaptive network architecture

$$\mu_{A_i}(x) = \frac{1}{1 + \left[\left(\frac{x - c_i}{a_i}\right)^2\right]^{b_i}} \qquad (2)$$

Layer 2: Every node in this layer is a fixed node labelled Π, which multiplies the incoming signals and outputs the product or T-norm operator result, e.g.

$$O_{2,i} = w_i = \mu_{A_i}(x) \times \mu_{B_i}(y), \quad i = 1,\ldots,n \quad (3)$$

Each node output represents the *firing strength* of a rule. (In fact, any other T-norm operators that perform the fuzzy AND operation can be used as the node function in this layer).

Layer 3 : Every node in this layer is a fixed node labelled N. The *ith* node calculates the ratio of the *ith* rules' firing strength to the sum of all rules' firing strengths:

$$O_{3,i} = \bar{w}_i = \frac{w_i}{\sum w_i}, \quad i = 1,\ldots,n \qquad (4)$$

For convenience, outputs of this layer are called *normalized firing strengths*.

Layer 4 : Every *ith* node in this layer is an adaptive node with a node function:

$$O_{4,i} = \bar{w}_i f_i = \bar{w}_i (p_i x + q_i y + r_i) \qquad (5)$$

where $\bar{w}_i$ is the output of layer 3 and $\{p_i, q_i, r_i\}$ is the parameter set. Parameters in this layer are referred to as *consequent parameters*.

Layer 5 : The single node in this layer is labelled $\sum$, which computes the overall output as the summation of incoming signals:

$$O_{5,1} = \text{overall output} = \sum_i \bar{w}_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \qquad (6)$$

Thus an adaptive network that has exactly the same function as a Sugeno fuzzy model may be constructed.

3. THE HYBRID LEARNING RULE

This learning rule was based upon the hybrid learning rule of Jang. The system is simulated using the dynamic model and data is collected across a specified trajectory. The required data being of the form ψ_ε, $\dot{\psi}$ and δ_d (desired rudder), which itself is calculated via a simple Jacobian approximation of the form:

$$\delta_\varepsilon = \frac{\psi(t) - \psi(t-1)}{\delta_\varepsilon(t) - \delta_\varepsilon(t-1)} \qquad (7)$$

where δ_ε is actual canard demand. This training data is used to compare the system trajectory with the desired trajectory, and so form the error measure to be used for training of the adaptive network parameters. The discrete error measure chosen was the integral square of heading error over time (ITSE):

$$E = \sum_{s=1}^{M} \frac{\left[(\psi_d - \psi_s)^2\right]}{N} \qquad (8)$$

The parameters to be altered are the fuzzy parameters of both the premise and consequent layers. The hybrid learning rule employs the backpropagation method to update the fuzzy premise parameters and the recursive least squares method to update the fuzzy consequent parameters.

Writing the premise membership function of equation (2) more generally as:

$$\mu_{ij}(x) = \frac{1}{1 + \left[\left(\frac{x - c_{ij}}{a_{ij}}\right)^2\right]^{b_{ij}}} \qquad (9)$$

It can be shown that by continuing the backpropagation process through each layer the following learning rules for each individual parameter within layer 1 are determined:

$$\Delta a_{ij} = -\eta \cdot \sum_{s=1}^{P} \frac{\partial E_s}{\partial O_{1s}} \cdot \frac{\partial O_{1s}}{\partial a_{ij}} \cdot$$

$$\left[\frac{2b_{ij}a_{ij}^{(2b_{ij}-1)}\left(x-c_{ij}\right)^2 a_{ij}^{2b_{ij}}\left(x-c_{ij}\right)^{2b_{ij}}}{\left[a_{ij}^{2b_{ij}}\left(x-c_{ij}\right)^{2b_{ij}} + \left(x-c_{ij}\right)^{2b_{ij}} a_{ij}^{2b_{ij}}\right]^2}\right] \qquad (10)$$

$$\Delta b_{ij} = -\eta \cdot \sum_{s=1}^{P} \frac{\partial E_s}{\partial O_{1s}} \cdot \frac{\partial O_{1s}}{\partial b_{ij}} \cdot$$

$$\left[\frac{2a_{ij}^{2b_{ij}}\left(x-c_{ij}\right)^{2b_{ij}} a_{ij}^{2b_{ij}}\left(x-c_{ij}\right)^{2b_{ij}} \ln\left[\left|\frac{x-c_{ij}}{x-c_{ij}}\right|\right]}{\left[a_{ij}^{2b_{ij}}\left(x-c_{ij}\right)^{2b_{ij}} + \left(x-c_{ij}\right)^{2b_{ij}} a_{ij}^{2b_{ij}}\right]^2}\right] \qquad (11)$$

$$\Delta c_{ij} = -\eta \cdot \sum_{s=1}^{P} \frac{\partial E_s}{\partial O_{1s}} \cdot \frac{\partial O_{1s}}{\partial c_{ij}} \cdot$$

$$\left[\frac{-2b_{ij}a_{ij}^{2b_{ij}}\left(c_{ij}-x\right)^{(2b_{ij}-1)} a_{ij}^{2b_{ij}}\left(-\left(x-c_{ij}\right)\right)^{2b_{ij}}}{\left[a_{ij}^{2b_{ij}}\left(-\left(x-c_{ij}\right)\right)^{2b_{ij}} + \left(c_{ij}-x\right)^{2b_{ij}} a_{ij}^{2b_{ij}}\right]^2}\right] \qquad (12)$$

4. FIXED RULE BASED FUZZY AUTOPILOT DESIGN

When operating as an autopilot, a fuzzy controller uses fuzzy rules to interpret input data and to generate an appropriate control output. Within the context of an AUV autopilot the rules take the typical form:

If yaw error (ψ_ε) is negative and yaw rate ($\dot{\psi}$) is positive then canard demand is $f(\psi_\varepsilon, \dot{\psi})$, where the terms "negative" and "positive" are fuzzy sets and canard demand is some function of ψ_ε and $\dot{\psi}$.

By defining universes of discourse for ψ_ε and $\dot{\psi}$ as E and CE respectively, and describing the output in the first order Sugeno form, such rules may be expressed as:

If E_i and CE_i then $Z_i = f(E_i, CE_i)$

where the fuzzy subsets E_i and CE_i are:

$$E_i = (\psi_\varepsilon, \mu_{Ei}(\psi_\varepsilon)) \subset E$$
$$Ce_i = (\dot{\psi}, \mu_{CEi}(\dot{\psi})) \subset CE$$

and ψ_ε and $\dot{\psi}$ are elements of the appropriate universes of discourse with membership functions of $\mu_{Ei}(\psi_\varepsilon)$ and $\mu_{CEi}(\dot{\psi})$ respectively.

Thus, in general, the N rules contained within the algorithm of the fuzzy autopilot may be expressed as:

R_1 : If E_1 and CE_1 then $Z_1 = f(E_1, CE_1)$ else
R_2 : If E_2 and CE_2 then $Z_2 = f(E_2, CE_2)$ else
: : : :
: : : :
: : : :
R_N : If E_N and CE_N then $Z_N = f(E_N, CE_N)$

In order to elicit a canard demand output (δ_ε) then:

$$w_i = E_i(\psi_\varepsilon) \wedge Ce_i(\dot{\psi})$$

$$\delta_\varepsilon = \frac{\sum_{i=1}^{N} w_i Z_i}{\sum_{i=1}^{N} w_i} \qquad (13)$$

Control systems which are based on fuzzy logic are inherently nonlinear and thus often extremely robust to parametric uncertainties. They therefore offer the potential to control AUVs in an effective manner.

5. RESULTS AND DISCUSSION

The ANFIS regime has been developed and applied to the task of tuning a course-changing fuzzy autopilot for an AUV simulation model. This section discusses the performance of the ANFIS tuned autopilot in a qualitative and quantitative manner. Comparisons are made with the Sugeno style fuzzy autopilot prior to tuning with the ANFIS technique.

Tuning of the autopilot parameters took place over a series of positive and negative course changes of 40°, at a surge velocity of 7.5 knots. Sufficient time intervals were allowed between consecutive course-changing demands to enable the AUV translational and rotational motions to stabilise, and thus ensure that each course-change was applied at similar initial conditions. This method was considered effective and necessary to ensure rule base symmetry.

Resulting from this tuning regime, the 7.5 knot ANFIS autopilot was taken as:

If ψ_ε is N and $\dot{\psi}$ is N then
$\delta = -1.46\psi_\varepsilon - 0.89\dot{\psi} + 0.66$

If ψ_ε is N and $\dot{\psi}$ is Z then
$\delta = -0.49\psi_\varepsilon - 0.88\dot{\psi} - 0.05$

If ψ_ε is N and $\dot{\psi}$ is P then
$\delta = -0.51\psi_\varepsilon - 0.89\dot{\psi} - 0.69$

If ψ_ε is Z and $\dot{\psi}$ is N then
$\delta = -0.45\psi_\varepsilon - 0.11\dot{\psi} + 0.79$

If ψ_ε is Z and $\dot{\psi}$ is Z then
$\delta = 0.00\psi_\varepsilon - 0.00\dot{\psi} + 0.00$

If ψ_ε is Z and $\dot{\psi}$ is P then
$\delta = -0.45\psi_\varepsilon - 0.11\dot{\psi} - 0.79$

If ψ_ε is P and $\dot{\psi}$ is N then
$\delta = -0.51\psi_\varepsilon - 0.89\dot{\psi} + 0.69$

If ψ_ε is P and $\dot{\psi}$ is Z then
$\delta = -0.49\psi_\varepsilon - 0.88\dot{\psi} + 0.05$

If ψ_ε is P and $\dot{\psi}$ is P then
$\delta = -1.46\psi_\varepsilon - 0.89\dot{\psi} - 0.66$

The untuned Sugeno autopilot was taken as:

If ψ_ε is N and $\dot{\psi}$ is N then $\delta = +25.00$
If ψ_ε is N and $\dot{\psi}$ is Z then $\delta = +18.75$
If ψ_ε is N and $\dot{\psi}$ is P then $\delta = +12.50$

If ψ_e is Z and $\dot{\psi}$ is N then δ = + 6.25
If ψ_e is Z and $\dot{\psi}$ is Z then δ = 0.00
If ψ_e is Z and $\dot{\psi}$ is P then δ = - 6.25
If ψ_e is P and $\dot{\psi}$ is N then δ = -12.50
If ψ_e is P and $\dot{\psi}$ is Z then δ = -18.75
If ψ_e is P and $\dot{\psi}$ is P then δ = - 25.00

and thus the Sugeno linear output functions were simply taken as a fuzzy singleton spikes, equally spaced on the output universe of discourse. The input fuzzy sets for both the ANFIS tuned autopilot and the original untuned Sugeno fuzzy autopilot can be seen in Figure 2.

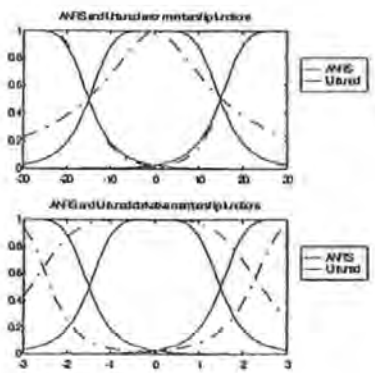


Fig. 2. The ANFIS and Untuned Input Fuzzy Sets.

A qualitative assessment of autopilot responsiveness was provided by the AUV simulation model's responses to a series of random course-changes of various magnitude, as shown in Figure 3. Such a track configuration was deemed necessary to assess the ability of the neurally-tuned autopilot to generalize to course-changes for which it had not been tuned. Figure 4 illustrates the corresponding canard demands to this particular track configuration.

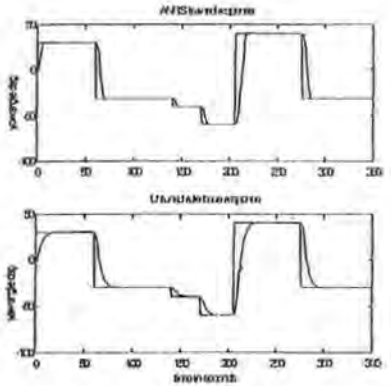


Fig. 3. Autopilot generalization track.

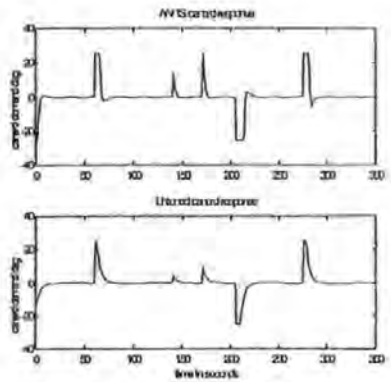


Fig. 4. Canard responses over autopilot generalization track.

The ANFIS tuned autopilot shows a superior course-changing response to that of the untuned autopilot, with faster rise times and no steady-state course error. Collectively, these results show the ability of the ANFIS tuned autopilot to generalize to large course-changing demands whilst applying a reasonable amount of control effort.

As a means of quantifying off-course error and course-changing control effort the following performance measures were adopted:

$$\psi_e = \int_0^t (\psi_d - \psi_a)^2 dt \qquad (14)$$

$$\delta_e = \int_0^t (\delta_d - \delta_a)^2 dt \qquad (15)$$

which represent the integral of squared error (ISE), where ψ_d and δ_d represent desired yaw angle and canard demand respectively, and ψ_a and δ_a represent actual yaw angle and canard demand respectively. Additionally, to assess the response speed of the AUV model and the oscillatory nature of each AUV response to a particular autopilot, figures pertaining to the rise time (T_r) and the percentage peak overshoot ($M_p(t)$) were collected. Rise time is considered here as the time to reach 99 per cent of the course-change demand and the percentage peak overshoot is calculated as a relative percentage of the course-change demand.

As the design process took place at 7.5 knots, the robustness of each autopilot was assessed by simulating AUV responses to a course-change of 40° at surge velocities of 5, 7.5and 10 knots. Thus Table 1 contains the results pertaining to these three AUV surge velocities. Additionally data are supplied for off-course error, canard effort, rise time and percentage peak overshoot.

When operating at 7.5 knots the autopilot tuned using the ANFIS technique was 33.37% more accurate than the benchmark Sugeno fuzzy autopilot. This illustrates that the ANFIS tuned autopilot produces a reduced off-course error for the 40° course-changing demand, as shown in Figure 5.

At 5 knots the effectiveness of the canard control surfaces is significantly diminished due to the reduced hydrodynamic forces acting on them. Intuitively one would anticipate more sluggish AUV response times as a consequence of this situation. Indeed this is borne out in the results of Table 1. The ANFIS tuned fuzzy autopilot proved to be 31.10% more accurate than the Sugeno fuzzy autopilots.

Conversely, the increased effectiveness of the canards at 10 knots lead to much sharper AUV responses. Figure 5 clearly illustrates the superior performance of the ANFIS tuned autopilot at 10 knots, with a reduction in off-course error of 38.19% over the Sugeno fuzzy autopilot.

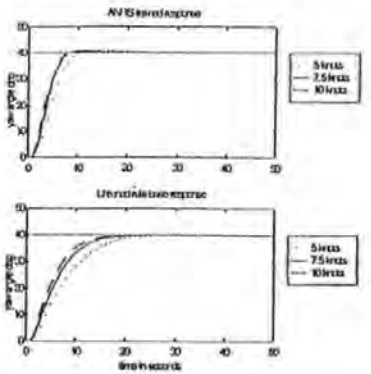


Fig. 5. Robustness of each autopilot to AUV surge parameter variations.

6. CONCLUSIONS

This paper has discussed the tuning of a fuzzy autopilot for course-changing control of an AUV using a neural network architecture and a hybrid learning algorithm. The resulting autopilots remain purely fuzzy as parameter tuning is conducted off-line and the network style implementation of the working controller is merely a convenience. From the results presented it may be concluded that the ANFIS approach provides a viable autopilot design solution.

7. ACKNOWLEDGEMENTS

The authors wish to thank the Sea Systems Sector, DERA, Winfrith, for supplying the AUV model and their continuing support throughout this study.

8. REFERENCES

Craven, P. J., R. Sutton and R.S. Burns. (1997). Intelligent Course-Changing Control of an Autonomous Underwater Vehicle. *Twelfth International Conference on Systems Engineering*, vol. 1, pp. 159-164.

Jang, J-S.R., (1993). ANFIS : adaptive network-based fuzzy inference system. *IEEE Transactions Systems, Man and Cybernetics*, vol. 23, no. 3, pp. 665-685.

Intelligent roll control of an autonomous underwater vehicle

R SUTTON, P J CRAVEN, and C M SOLOMON
Institute of Marine Studies, University of Plymouth, UK

SYNOPSIS

This paper describes the development of two roll control autopilots for use in an autonomous underwater vehicle. The autopilot designs are based on a fuzzy controller optimised using a genetic algorithm and on the neurofuzzy methodology known as the adaptive network-based fuzzy inference system. To describe the dynamics of the underwater vehicle use is made of a sophisticated mathematical model which is hosted in the MATLAB / Simulink environment. Owing to the control surface configuration of the model, severe cross-coupling occurs between the yaw and roll channels which results in unwanted roll motions occurring as a consequence of heading demands. Simulation results are presented and comparisons between the autopilot designs are made. It is concluded that both approaches offer viable alternative methods for designing such marine control systems.

1. INTRODUCTION

In the work reported herein, a realistic autonomous underwater vehicle (AUV) model is employed in a simulation study. Owing to the control surface configuration of the AUV model, severe cross-coupling occurs between the yaw and roll channels which causes an unwanted roll motion as a result of applied canard demands. This paper describes the development of two roll control autopilots based on a fuzzy controller which has been optimised using a genetic algorithm (GA) and on the innovative neurofuzzy methodology of Jang known as the adaptive network-based fuzzy inference system (ANFIS) [1].

To describe the dynamic behaviour of an AUV use is made of a MATLAB / Simulink model supplied by the Defence Evaluation and Research Agency (DERA), Sea Systems Sector, Winfrith. The AUV model having been validated against standard DERA non-linear hydrodynamic code using tank test data and an experimentally derived set of hydrodynamic coefficients from the Southampton Oceanography Centre's Autosub vehicle. In the interest of brevity, a description of the AUV dynamics will not be given here. Instead, the interested reader should refer to reference [2] for full details.

2. FUZZY RULE BASED AUTOPILOT DESIGN

2.1 Generic structure of the fuzzy autopilot

When in operation a fuzzy controller uses fuzzy rules to interpret its input data and to generate an appropriate control output. Within the context of an AUV autopilot and its internal structure, the rules may take the typical form:

> If roll is positive small and roll rate is positive big then hydroplane demand is zero.

Where, the terms 'positive small', 'positive big' and 'zero' are fuzzy sets.

Whilst fuzzy sets are usually illustrated as continuous functions, for implementation purposes they are often in a quantised form. By defining discrete universes of discourse for quantised roll (e), quantised roll rate (ce) and output (u) as E, CE and U respectively, such rules may be expressed as:

$$\text{If } E_i \text{ and } CE_i \text{ then } U_i$$

Where the fuzzy subsets E_i, CE_i and U_i are:

$$E_i = (e, \mu_{Ei}(e)) \subset E$$
$$CE_i = (ce, \mu_{CEi}(ce)) \subset CE$$
$$U_i = (u, \mu_{Ui}(u)) \subset U$$

and e, ce and u are elements of the appropriate discrete universes of discourse with membership functions of $\mu_{Ei}(e)$, $\mu_{CEi}(ce)$ and $\mu_{Ui}(u)$ respectively.

Thus, in general, the N rules contained within the algorithm of the fuzzy autopilot may be expressed as:

$$R_1 : \text{If } E_1 \text{ and } CE_1 \text{ then } U_1 \text{ else}$$
$$R_2 : \text{If } E_2 \text{ and } CE_2 \text{ then } U_2 \text{ else}$$
$$\vdots \quad \vdots \quad \vdots \quad \vdots$$
$$R_N : \text{If } E_N \text{ and } CE_N \text{ then } U_N$$

which can be summarised in the fuzzy relation:

$$R = R_1 \cup R_2 \cup \ldots\ldots \cup R_N = \bigcup_{i=1}^{N} (E_i \times CE_i \times U_i) \qquad (1)$$

$$R(e_i, ce_j, u_k) = \max [E_i(e_i) \vee CE_i(ce_j) \vee U_i(u_k)] \quad 1 \le i \le N \tag{2}$$

To enable the fuzzy autopilot to operate for any given input use is made of the computational rule of inference:

$$U = (E \times CE) \circ R \tag{3}$$

Thus, equation (3) may be written as:

$$U(u_k) = \bigvee_{e \in E,\ ce \in CE} \left[E(e_i) \wedge CE(ce_j) \wedge R(e_i, ce_j, U_k) \right] \tag{4}$$

In order to elicit a deterministic value from the resultant fuzzy control output set, the centre of area method is employed:

$$u_o = \frac{\sum_{i=1}^{M} u_i \, U(u_i)}{\sum_{i=1}^{M} U(u_i)} \tag{5}$$

2.2 Initial fuzzy autopilot design

Given the general shape and configuration of the control surfaces of the AUV, it was deemed appropriate to initiate the design study by using the fuzzy roll autopilot which was heuristically developed for a torpedo model by Sutton et al [4]. Thus, the fuzzy rule was taken as shown in Table 1.

Table 1 Initial fuzzy rule base

ϕ / $\dot{\phi}$ °/sec	NB	NM	NS	AZ	PS	PM	PB
NB	PB	PB	PM	PM	NS	NM	NM
NS	PB	PB	PM	AZ	NS	NM	NB
AZ	PB	PM	PS	AZ	NS	NM	NB
PS	PB	PM	PS	AZ	NM	NB	NB
PB	PM	PM	PS	NM	NM	NB	NB

The fuzzy sets being defined on the universes of discourse of roll (ϕ), roll rate ($\dot{\phi}$) and hydroplane (rudder) demand. Initially, triangular shaped fuzzy sets were chosen which were symmetrically and evenly spaced on the respective universes of discourse. The linguistic labels used to describe the fuzzy sets being: negative big (NB), negative medium (NM), negative small (NS), zero (AZ), positive small (PS), positive medium (PM) and positive big (PB).

±1°/sec and ±30° respectively.

2.3 Genetic algorithm concepts

In the past, fuzzy controllers have been designed using purely trial and error methods to obtain an optimised solution. Here a GA is used to optimise the consequent terms of the fuzzy rule base and then the shapes and positions of the fuzzy sets on their respective universes of discourse. Based on the Darwinian theory of genetic evolution in which nature tends to favour the survival of the stronger and fitter members of the population rather than the weaker ones, the GA is a global optimisation technique founded upon this natural selection principle. The GA employed in the study used standard operating functions which may be found in reference [5].

3. NEUROFUZZY AUTOPILOT DESIGN

As mentioned above, one of the controller designs used in this study is based on the ANFIS. Functionally, there are almost no constraints on the membership functions of an adaptive network except piecewise differentiability. Structurally, the only limitation on network configuration is that it should be of feed-forward type. Due to these minimal restrictions, the adaptive network's applications are immediate and immense in various areas.

If it is assumed that the fuzzy inference system under consideration has multiple inputs and one functional output (f) then the fuzzy rule-based algorithm may be represented in the first order Sugeno form as shown below:

Rule 1 : If x is A_1 and y is B_1 then $f_1 = p_1 x + q_1 y + r_1$

Rule 2 : If x is A_2 and y is B_2 then $f_2 = p_2 x + q_1 y + r_2$

: : : : : :

: : : : : :

Rule n : If x is A_n and y is B_n then $f_n = p_n x + q_n y + r_n$

The corresponding ANFIS architecture is shown in Figure 1.

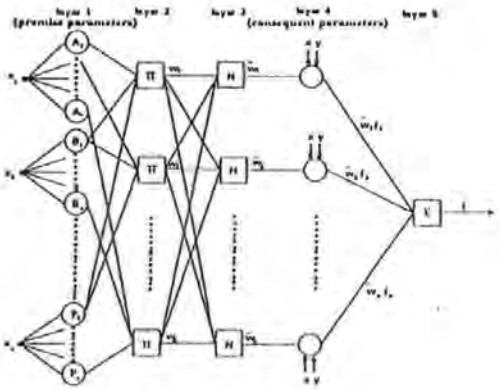


Fig 1 The adaptive network architecture

The node functions in the same layer are of the same function family as described by the following:

Layer 1: Every *ith* node in this layer is an adaptive node with a node output defined by:

$$O_{1,i} = \mu_{A_i}(x) \tag{6}$$

where x is the input to the general node and A_i is the fuzzy set associated with this node. In other words, outputs of this layer are the membership values of the premise part. Here the membership functions for A_i can be any appropriate parameterised membership functions. Here A_i is characterised by the generalised bell function:

$$\mu_{A_i}(x) = \frac{1}{1 + \left[\left(\frac{x - c_i}{a_i}\right)^2\right]^{b_i}} \tag{7}$$

where (a_i, b_i, c_i) is the parameter set. Parameters in this layer are referred to as *premise parameters*.

Layer 2: Every node in this layer is a fixed node labelled Π, which multiplies the incoming signals and outputs the product or T-norm operator result, eg.

$$O_{2,i} = w_i = \mu_{A_i}(x) \times \mu_{B_i}(y), \; i = 1, 2 \tag{8}$$

Each node output represents the *firing strength* of a rule. (In fact, any other T-norm operators that perform the fuzzy AND operation can be used as the node function in this layer.)

Layer 3: Every node in this layer is a fixed node labelled N. The *ith* node calculates the ratio of the *ith* rules' firing strength to the sum of all rules' firing strengths:

$$O_{3,i} = \bar{w}_i = \frac{w_i}{w_1 + w_2}, \quad i = 1, 2 \tag{9}$$

For convenience, outputs of this layer are called *normalised firing strengths*.

Layer 4: Every *ith* node in this layer is an adaptive node with a node function:

$$O_{4,i} = \bar{w}_i f_i = \bar{w}_i (p_i x + q_i y + r_i) \tag{10}$$

where $\bar{w}_i$ is the output of Layer 3 and (p_i, q_i, r_i) is the parameter set. Parameters in this layer are referred to as *consequent parameters*.

Layer 5: The single node in this layer is labelled $\sum$, which computes the overall output as the summation of incoming signals:

$$O_{5,1} = \text{overall output} = \sum_i \bar{w}_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \tag{11}$$

Thus an adaptive network that has exactly the same function as a Sugeno fuzzy model may be constructed.

3.1 Learning rules

Rewriting the premise membership function of equation (7) as:

$$\mu_{ij}(x) = \frac{1}{1 + \left[\left(\frac{x - c_{ij}}{a_{ij}}\right)^2\right]^{b_{ij}}} \tag{12}$$

then equation (12) now represents the *jth* membership function on the *ith* input universe of discourse.

Therefore the learning rule for a general parameter may be described as follows:

$$\Delta\alpha_{ij} = -\eta \sum_{n=1}^{P} \frac{\partial E_n}{\partial O^1_{1n}} \cdot \frac{\partial O_{1n}}{\partial \alpha_{ij}}$$

$$= -\eta \sum_{n=1}^{P} \frac{\partial E_n}{\partial O^{ij}_{2n}} \cdot \frac{\partial O^{ij}_{2n}}{\partial O^{ij}_{1n}} \cdot \frac{\partial O^{ij}_{1n}}{\partial \alpha_{ij}} \qquad (13)$$

Hence the learning rules for each individual parameter are:

$$\Delta a_{ij} = -\eta \sum_{n=1}^{P} \frac{\partial E_n}{\partial O^{ij}_{2n}} \cdot \frac{\partial O^{ij}_{2n}}{\partial O^{ij}_{1n}} \cdot \left[\frac{2b_{ij} a_{ij}^{(2b_{ij}-1)} \left(x - c_{ij}\right)^{2b_{ij}}}{\left\{ a_{ij}^{2b_{ij}} \left(x - c_{ij}\right)^{2b_{ij}} + \left(x - c_{ij}\right)^{2b_{ij}} a_{ij}^{2b_{ij}} \right\}^2} \right] \qquad (14)$$

$$\Delta b_{ij} = -\eta \sum_{n=1}^{P} \frac{\partial E_n}{\partial O^{ij}_{2n}} \cdot \frac{\partial O^{ij}_{2n}}{\partial O^{ij}_{1n}} \cdot \left[\frac{2a_{ij}^{2b_{ij}} \left(x - c_{ij}\right)^{2b_{ij}} a_{ij}^{2b_{ij}} \left(x - c_{ij}\right)^{2b_{ij}} \ln\left[\left|\frac{a_{ij}}{x - c_{ij}}\right|\right]}{\left\{ a_{ij}^{2b_{ij}} \left(x - c_{ij}\right)^{2b_{ij}} + \left(x - c_{ij}\right)^{2b_{ij}} a_{ij}^{2b_{ij}} \right\}^2} \right] \qquad (15)$$

$$\Delta c_{ij} = -\eta \sum_{n=1}^{P} \frac{\partial E_n}{\partial O^{ij}_{2n}} \cdot \frac{\partial O^{ij}_{2n}}{\partial O^{ij}_{1n}} \cdot \left[\frac{-2b_{ij} a_{ij}^{2b_{ij}} \left(c_{ij} - x\right)^{(2b_{ij}-1)} a_{ij}^{2b_{ij}} \left(-\left(x - c_{ij}\right)\right)^{2b_{ij}}}{\left\{ a_{ij}^{2b_{ij}} \left(-\left(x - c_{ij}\right)\right)^{2b_{ij}} + \left(c_{ij} - x\right)^{2b_{ij}} a_{ij}^{2b_{ij}} \right\}^2} \right] \qquad (16)$$

The fuzzy consequent parameters being updated using a recursive least squares method.

4. RESULTS AND DISCUSSION

Using the rule base shown in Table 1 resulted in an autopilot design which, when in operation, produced a reasonable reduction in the roll response, however, to achieve this the control actuators were almost permanently saturated. Clearly such control action cannot be tolerated. Whilst in operation the role of an autopilot is to minimise the induced roll as quickly as possible with the minimum of control effort. Thus, to encompass this performance criteria, the objective function (OF) for the GA was selected to take into account a measure of the roll and the hydroplane demand as follows:

$$OF = \frac{1}{T} \int_{t=0}^{t=T} (\phi_e^2 + 0.1\,\delta_d^2)\,dt \qquad (17)$$

where ϕ_e is the roll error and δ_d is the hydroplane demand.

As stated earlier, the first stage in the development of the fuzzy autopilot was to use the GA to optimise the consequent terms of the rule base. Employing a single point crossover function resulted in the rule base shown in Table 2.

Table 2 Final fuzzy rule base

ϕ / $\dot{\phi}$ °/sec	NB	NM	NS	AZ	PS	PM	PB
NB	PS	NS	PM	NM	PS	NS	PB
NS	NS	PB	PM	PB	NB	PB	PS
AZ	NS	PM	PM	AZ	NB	NM	PM
PS	AZ	NM	PM	NS	NM	PM	AZ
PB	NM	NM	NB	PM	NM	AZ	NM

The next stage in the development process involved using the GA to optimise the fuzzy set positions on their respective universes of discourse and produced the results shown in Figure 2.

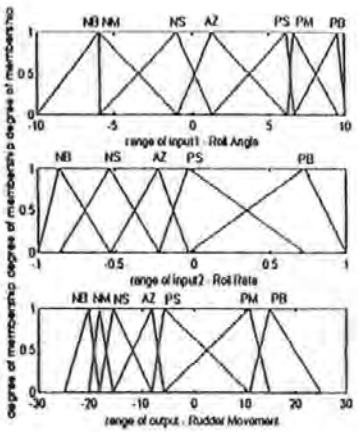


Fig 2 Fuzzy sets for the final fuzzy autopilot design

At first the lack of symmetry in the sets was a point for concern. Therefore adjustments were made to the set positions to discover whether symmetrical placings would produce better results. However, it was discovered that any alterations only degraded the performance of the

fuzzy roll autopilot. It was decided the lack of symmetry was linked to the non-linear nature of the system dynamics and therefore a necessary characteristic.

For the AUV travelling at 7.5 knots without any roll controller, the maximum roll induced as a consequence of a 40° yaw demand was 9.11°. However, under the same operating conditions, the AUV with the GA optimised fuzzy roll autopilot, the maximum roll was reduced to 1.04° as seen in Figure 3.

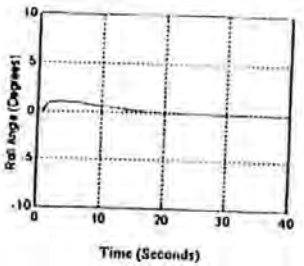


Fig 3 AUV roll response to a 40° yaw demand with the GA optimised fuzzy roll autopilot

With reference to the ANFIS roll autopilot, after experiencing an appropriate training regime, the rule base converged to the following:

If ψ_e is N and $\dot{\psi}$ is N then $\delta = 0.1671\ \psi_e - 0.0575\ \dot{\psi} - 0.01045$

If ψ_e is N and $\dot{\psi}$ is Z then $\delta = 0.05253\ \psi_e + 0.3766\ \dot{\psi} + 0.02821$

If ψ_e is N and $\dot{\psi}$ is P then $\delta = 0.1781\ \psi_e - 0.02979\ \dot{\psi} - 0.00695$

If ψ_e is N and $\dot{\psi}$ is P then $\delta = 0.0227\ \psi_e + 0.2005\ \dot{\psi} + 0.2688$

If ψ_e is Z and $\dot{\psi}$ is Z then $\delta = 0.002441\ \psi_e + 0.01659\ \dot{\psi} - 0.03516$

If ψ_e is Z and $\dot{\psi}$ is P then $\delta = -0.01542\ \psi_e - 0.0382\ \dot{\psi} + 0.1082$

If ψ_e is P and $\dot{\psi}$ is N then $\delta = -0.7792\ \psi_e + 0.1229\ \dot{\psi} - 0.05202$

If ψ_e is P and $\dot{\psi}$ is Z then $\delta = 0.1285\ \psi_e + 0.04216\ \dot{\psi} + 0.04187$

If ψ_e is P and $\dot{\psi}$ is P then $\delta = 0.251\ \psi_e + 0.02147\ \dot{\psi} + 0.02312$

In this case for the corresponding circumstances as above, the ANFIS roll autopilot produced a maximum roll angle of 1.14° as depicted in Figure 4.

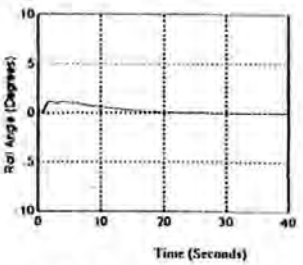


Fig 4 AUV roll response to a 40° yaw demand with the ANFIS roll autopilot.

5. CONCLUDING REMARKS

In conclusion, it can be stated that from the study described herein the viability of using a fuzzy roll autopilot which has been optimised by a GA has been demonstrated. Also the analytical approach adopted gives more confidence in the design to that developed exclusively by heuristic means. Equally so, the same comments can be made for the neurofuzzy approach used for the ANFIS roll autopilot. Both methodologies are shown to offer feasible means of designing roll autopilots for AUVs.

6. ACKNOWLEDGEMENT

The authors wish to thank the Sea Systems Sector, DERA, Winfrith, for supplying the AUV model and their continuing support for the work being undertaken.

7. REFERENCES

1. JANG, J.S.R. ANFIS: Adaptive network-based fuzzy inference system, IEEE Transaction on Systems, Man and Cybernetics, 1993, Vol 23, pp 665-685.

2. SUTTON, R. and CRAVEN, P.J. Fuzzy yaw autopilots for unmanned underwater vehicles using artificial neural networks, Underwater Technology. (To be published.)

3. PEDRYCZ, W. Fuzzy control and fuzzy systems, 1993, (Research Studies Press).

4. SUTTON, R., STACEY, B.A and JONES, A.R.T. Methods of reducing rudder induced roll in a three fin torpedo, Proceedings of AMST 90, Bradford, UK, 1990, pp 187-193 (Elsevier Applied Science).

5. GOLDBERG, D.E. Genetic algorithms in search, optimisation and machine learning, 1989 (Addison-Wesley Publishing Co.).

MULTI - INPUT SINGLE - OUTPUT AUTOPILOTS FOR AUTONOMOUS UNDERWATER VEHICLE CONTROL

Paul J Craven, Robert Sutton and Yong-Ming Dai

University of Plymouth, Drake Circus, Plymouth, UK, PL4 8AA
Tel: +44 (0)1752 232407 Fax: +44 (0)1752 232406 e-mail: pcraven@plymouth.ac.uk

KEYWORDS neurofuzzy, autopilot, tuning, course-changing, autonomous underwater vehicle

ABSTRACT

This paper describes the application of two neurofuzzy techniques to the tuning of fuzzy autopilots for course-changing control of an autonomous underwater vehicle. Autopilots are designed using an adaptive network-based fuzzy inference system (ANFIS) which performs parameter tuning via a gradient descent and a sequential least squares based hybrid algorithm, and a chemotaxis tuning methodology which relies upon a random search technique. To describe the yaw dynamic characteristics of an autonomous underwater vehicle a realistic simulation model is employed. Results are presented which demonstrate the superiority of the ANFIS approach. It is concluded that the approach offers a viable alternative method for designing such autopilots.

INTRODUCTION

Encouraged by the prospect of increasingly more ambitious autonomous underwater vehicle (AUV) mission scenarios considerable interest is now being shown in the development and construction of these craft to undertake tasks such as ocean surveying, pipeline inspection, explosive ordnance disposal and covert surveillance. In order that they can possess a suitable level of autonomy it is necessary for AUVs to possess reliable and robust onboard navigation, guidance and control (NGC) systems. A key element of the NGC system is the control subsystem which is responsible for maintaining the vehicle trajectory.

Due to the inherently nonlinear and coupled dynamics of AUVs, they present a formidable control problem. Generally, the amount of coupling is reduced by separating the system into non-coupled sub-systems. This method has been employed by various researchers and has met with varying degrees of success.

More recent control system designs have usually incorporated some form of artificial intelligence. Autopilots formulated using fuzzy logic (Smith et al., (1993)) and artificial neural network (ANN) methods (Yuh, (1990)) have been shown to be endowed with commendable robustness properties.

This paper concerns investigative work into fuzzy multi-input single-output (MISO) autopilots for course-changing control. More specifically, the use of an adaptive neural network tuning architecture for the encoding of Sugeno style fuzzy autopilots is discussed.

MODELING THE VEHICLE DYNAMICS

It should be noted that for this study the upper and lower canards are used to control the AUV's yaw dynamics. Dimensionally the model represents an AUV which is 7 m long, 1 m in diameter and has a displacement of approximately 3600 kg. The equation of motion describing the dynamic behaviour of the vehicle in the lateral plane is as follows:

$$E\dot{x} = Fx + Gu \qquad (1)$$

where :

$$E = \begin{bmatrix} (m - Y_{\dot{v}}) & -Y_{\dot{r}} & 0 & -(Y_{\dot{p}} + mZ_G) & 0 \\ -N_{\dot{v}} & (I_z - N_{\dot{r}}) & 0 & -N_{\dot{p}} & 0 \\ 0 & 0 & 1 & 0 & 0 \\ -(K_{\dot{v}} + mZ_G) & -K_{\dot{r}} & 0 & (I_x - K_{\dot{p}}) & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$u = [\,\delta_{\text{aft-upper}}, \delta_{\text{aft-lower}}, 0\ 0\ 0\ 0\,]^T$$

$$\begin{bmatrix} Y_{uv}U & (Y_{ur}-m)U & 0 & Y_{up}U & 0 \\ N_{uv}U & N_{ur}U & 0 & N_{up}U & 0 \\ 0 & 1 & 0 & 0 & 0 \\ K_{uv}U & (K_{ur}+mZ_G)U & 0 & K_{up}U & -mgBG \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

G =

$$\begin{bmatrix} Y_{\delta_{upper}}U^2 & Y_{\delta_{lower}}U^2 & 0 & 0 & 1 & 1 \\ N_{\delta_{upper}}U^2 & N_{\delta_{lower}}U^2 & \lambda_\psi & -\lambda_\psi & +\lambda_\psi & -\lambda_\psi \\ 0 & 0 & 0 & 0 & 0 & 0 \\ K_{\delta_{upper}}U^2 & K_{\delta_{lower}}U^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

and the state variables are V, R, ψ, P, ϕ. To implement equation (1) use is made of an AUV MATLAB / Simulink simulation model supplied by the Defence Evaluation and Research Agency (DERA), Sea Systems Sector, Winfrith, the model having been validated against DERA non-linear hydrodynamic code using tank test data and experimentally derived hydrodynamic coefficients from the Southampton Oceanography Centre's AUTOSUB vehicle. As can be seen in equation (1) the roll cross-coupling dynamics are included. However, control of the roll dynamics is not considered here.

NEUROFUZZY AUTOPILOT DESIGN

If it is assumed that the fuzzy inference system under consideration has multiple inputs (x_i) and one functional output (f) then the fuzzy rule-based algorithm may be represented in the first order Sugeno form as shown below:

Rule 1 : If x_1 is A_1 and x_2 is B_1 and ... and x_n is P_1
then $f_1 = p_1 x_1 + q_1 x_2 + ... + v_1$

Rule 2 : If x_1 is A_2 and x_2 is B_1 and ... and x_n is P_1
then $f_2 = p_2 x_1 + q_2 x_2 + ... + v_2$

Rule n : If x_1 is A_n and x_2 is B_n and ... and x_n is P_n
then $f_n = p_n x_1 + q_n x_2 + ... + v_n$

The corresponding neurofuzzy architecture is depicted Figure 1.

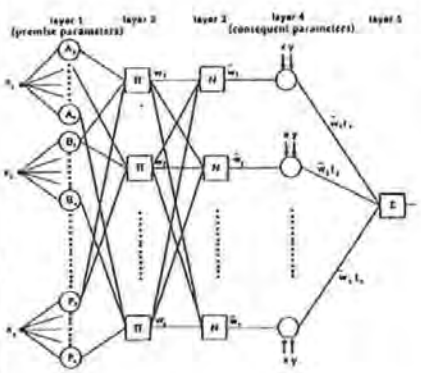


Figure 1. The adaptive neurofuzzy architecture

The node functions in each individual layer are of the same function family as described by the following:

Layer 1: Every *ith* node in this layer is an adaptive node with a node output defined by:

$$O_{1,i} = \mu_{A_i}(x_i) \qquad (2)$$

where x_i is the input to the node and A_i is the fuzzy set associated with the nodes pertaining to input x_i. In other words, outputs of this layer are the membership values of the premise part. The membership functions can be any appropriate parameterized shapes. Here $A_i, B_i, ..., P_i$ are characterized by the generalized bell function:

$$\mu_{A_i}(x) = \frac{1}{1 + \left[\left(\frac{x - c_i}{a_i}\right)^2\right]^{b_i}} \qquad (3)$$

where $\{a_i, b_i, c_i\}$ is the parameter set. Parameters in this layer are referred to as *premise parameters*.

$$\begin{bmatrix} Y_{\dot{v}v}U & (Y_{\dot{r}R}-m)U & 0 & Y_{\dot{r}P}U & 0 \\ N_{\dot{v}v}U & N_{\dot{r}R}U & 0 & N_{\dot{r}P}U & 0 \\ 0 & 1 & 0 & 0 & 0 \\ K_{\dot{v}v}U & (K_{\dot{r}R}+mz_G)U & 0 & K_{\dot{r}P}U & -mgBG \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$G =$$

$$\begin{bmatrix} Y_{UUu_c}U^2 & Y_{UUR}U^2 & 0 & 0 & 1 & 1 \\ N_{UUu_c}U^2 & N_{UUR}U^2 & \lambda_\phi & -\lambda_\phi & \lambda_\psi & -\lambda_\psi \\ 0 & 0 & 0 & 0 & 0 & 0 \\ K_{UUu_c}U^2 & K_{UUR}U^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

and the state variables are V, R, ψ, P, ϕ. To implement equation (1) use is made of an AUV MATLAB / Simulink simulation model supplied by the Defence Evaluation and Research Agency (DERA), Sea Systems Sector, Winfrith, the model having been validated against DERA non-linear hydrodynamic code using tank test data and experimentally derived hydrodynamic coefficients from the Southampton Oceanography Centre's AUTOSUB vehicle. As can be seen in equation (1) the roll cross-coupling dynamics are included. However, control of the roll dynamics is not considered here.

NEUROFUZZY AUTOPILOT DESIGN

If it is assumed that the fuzzy inference system under consideration has multiple inputs (x_i) and one functional output (f) then the fuzzy rule-based algorithm may be represented in the first order Sugeno form as shown below:

Rule 1 : If x_1 is A_1 and x_2 is B_1 and ... and x_n is P_1
then $f_1 = p_1 x_1 + q_1 x_2 + ... + v_1$

Rule 2 : If x_1 is A_2 and x_2 is B_1 and ... and x_n is P_1
then $f_2 = p_2 x_1 + q_2 x_2 + ... + v_2$

Rule n : If x_1 is A_n and x_2 is B_n and ... and x_n is P_n
then $f_n = p_n x_1 + q_n x_2 + ... + v_n$

The corresponding neurofuzzy architecture is depicted in Figure 1.

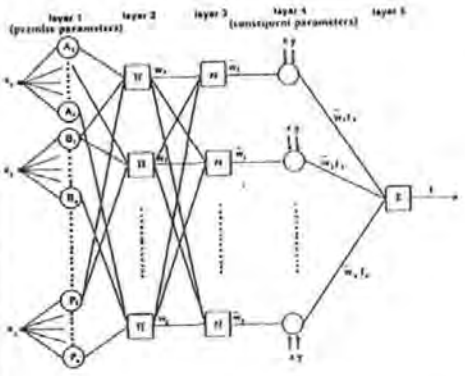


Figure 1. The adaptive neurofuzzy architecture

The node functions in each individual layer are of the same function family as described by the following:

Layer 1: Every *ith* node in this layer is an adaptive node with a node output defined by:

$$O_{1,i} = \mu_{A_i}(x_i) \qquad (2)$$

where x_i is the input to the node and A_i is the fuzzy set associated with the nodes pertaining to input x_i. In other words, outputs of this layer are the membership values of the premise part. The membership functions can be any appropriate parameterized shapes. Here $A_i, B_i, ..., P_i$ are characterized by the generalized bell function:

$$\mu_{A_i}(x) = \frac{1}{1+\left[\left(\frac{x-c_i}{a_i}\right)^2\right]^{b_i}} \qquad (3)$$

where $\{a_i, b_i, c_i\}$ is the parameter set. Parameters in this layer are referred to as *premise parameters*.

Layer 2: Every node in this layer is a fixed node labelled Π, which multiplies the incoming signals and outputs the product or T-norm operator result, e.g.

$$O_{2,i} = w_i = \mu_A(x_1) \times \mu_B(x_2) \times ... \times \mu_Z(x_n), \quad i = 1,2,...,n. \qquad (4)$$

Each node output represents the *firing strength* of a rule.

Layer 3: Every node in this layer is a fixed node labelled N. The *ith* node calculates the ratio of the *ith* rules' firing strength to the sum of all rules' firing strengths:

$$O_{3,i} = \bar{w}_i = \frac{w_i}{\sum_i w_i}, \quad i = 1,2,...,n. \qquad (5)$$

For convenience, outputs of this layer are called *normalized firing strengths*.

Layer 4: Every *ith* node in this layer is an adaptive node with a node function:

$$O_{4,i} = \bar{w}_i f_i = \bar{w}_i (p_i x_1 + q_i x_2 + ... + v_i) \qquad (6)$$

where $\bar{w}_i$ is the output of layer 3 and $\{p_i, q_i, ..., v_i\}$ is the parameter set. Parameters in this layer are referred to as *consequent parameters*.

Layer 5: The single node in this layer is labelled $\sum$ which computes the overall output as the summation of incoming signals:

$$O_{5,1} = \text{overall output} = \sum_i \bar{w}_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \qquad (7)$$

Thus an adaptive network that has exactly the same function as a Sugeno fuzzy model may be constructed. This work considers fuzzy autopilots of two input - single output form, the input variables being yaw error (degrees) and yaw rate (degrees per second) and the output variable being canard demand (degrees).

TRAINING ALGORITHMS

The Hybrid Learning Rule

This section discusses the hybrid learning rule of Jang (1993). The system is simulated using the dynamic model and data is collected across a trajectory. This training data is used to compare the system trajectory with the desired trajectory, and so form the error measure to be used for training of the adaptive network parameters. The error measure chosen was the integral square of heading error over time (ITSE):

$$E = \int_0^t (\psi_d - \psi_a)^2 dt \qquad (8)$$

The parameters to be altered are the fuzzy parameters of both the premise and consequent layers. The learning rule combines a backpropagation method to update the fuzzy premise parameters and a recursive least squares method to update the fuzzy consequent parameters in an adaptive network. Each epoch consists of a forward pass in which inputs are presented and the consequent parameters are updated via the recursive least-squares method, and a backward pass in which the derivative of the error measure with respect to each nodes' output is propagated from the output to the input of the network architecture. At the end of the backward pass the parameters of the premise layer are updated by a gradient descent method.

Chemotaxis Algorithm

The main disadvantage of backpropagation is the tendency for the search to become trapped in a local minimum of the error hypersurface. The more complex the network the more likely this is to happen as the error surface is increasingly multi-dimensional and therefore irregular, with more local minima into which the partially trained network can fall. An alternative is to use less guided methods to search the parameter space. Such random methods are virtually guaranteed to find a global solution but training times may be somewhat extended as the search pattern is completely random in nature.

The chemotaxis algorithm was inspired by observations of the movement of bacteria in a chemical environment, hence 'chemo' - chemical, and 'taxis' - movement [Koshland, (1980)]. In the presence of an irritant, bacteria would move randomly away in any direction in order to reduce the irritation, until this direction took them into an area where the irritation began to increase again. A new, random direction would then be chosen and if this again led to less irritation the bacteria would head in the new direction, otherwise another random direction would be tried. In time the bacteria would be located at the global minima, that is, furthest from the source of irritation. This behaviour may be transformed into a general search algorithm for an optimum sets of weights or parameters. The increase/decrease in irritation may be characterized by an increase/decrease in a suitable cost function for the optimization. By converting this better/worse information into a reinforcement signal according to:

$$r(t) = 1, \text{ better}$$
$$r(t) = 0, \text{ worse} \qquad (9)$$

the chemotaxis search algorithm is obtained, which may be classed as a reinforcement learning technique. The algorithm is summarized in Table 1.

1. Simulate the system with an initial set of parameters.
2. Generate some small random changes in the parameters and re-simulate the system.
3. If the system's performance has improved with the new set of parameters, retain the changes and re-apply. This is essentially assuming that the local cost function gradient will continue to be negative in the local area.
4. If the system performance has worsened, return to step 2.
5. Continue until the system has reached an acceptable level.

Table 1. The Chemotaxis Algorithm

Given sufficient training time, the algorithm should converge to a global minimum of the cost function, although given the random nature of the search a lengthy training period is often necessary.

The structure of the chemotaxis tuned autopilot is similar to that described in section 3 and depicted in Figure 1. During the tuning process, input data are passed forward through the network architecture to generate an error function. The chemotaxis algorithm is then applied to simultaneously search for a set of premise and consequent parameters which minimize the cost function of equation (8).

RESULTS AND DISCUSSION

In the previous sections the development of two Sugeno type fuzzy MISO autopilots has been discussed. The hybrid learning algorithm and the chemotaxis algorithm were applied to the task of tuning the premise and consequent parameters of a fuzzy autopilot by encoding the autopilot as an adaptive network architecture.

Tuning of the network parameters took place over a series of positive and negative course changes, determined *a priori* to stimulate a wide range of the vehicles yaw dynamics.

On completion of the 300 epoch tuning regime the resulting input fuzzy sets for the two autopilots were as shown in Figure 2. The original fuzzy sets are included to highlight the evolution of the sets during the tuning procedures. The non-symmetrical nature of the tuned fuzzy rulebases was considered mainly due to computer truncation errors arising during the training processes, and in the case of the ANFIS tuning regime due to the approximate nature of the initial conditions required to bootstrap the calculation of the sequential least squares estimate.

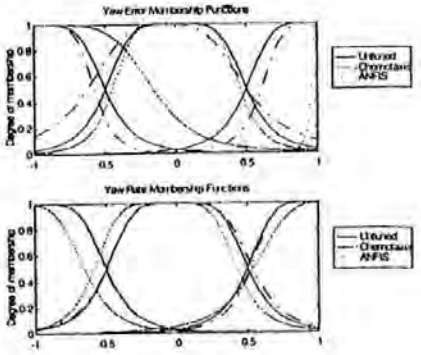


Figure 2. The tuned input fuzzy sets

The rulebase of the ANFIS tuned Sugeno style fuzzy autopilot was taken as:

If ψ_ε is N and $\dot{\psi}$ is N then $\delta = -0.487\,\psi_\varepsilon - 0.879\,\dot{\psi} - 0.029$

If ψ_ε is N and $\dot{\psi}$ is Z then $\delta = -0.489\,\psi_\varepsilon - 0.902\,\dot{\psi} + 0.001$

If ψ_ε is N and $\dot{\psi}$ is P then $\delta = -0.486\,\psi_\varepsilon - 0.896\,\dot{\psi} + 0.003$

If ψ_ε is Z and $\dot{\psi}$ is N then $\delta = -0.299\,\psi_\varepsilon - 0.703\,\dot{\psi} - 0.123$

If ψ_ε is Z and $\dot{\psi}$ is Z then $\delta = -0.488\,\psi_\varepsilon - 0.891\,\dot{\psi} + 0.004$

If ψ_ε is Z and $\dot{\psi}$ is P then $\delta = -0.305\,\psi_\varepsilon - 0.306\,\dot{\psi} - 0.037$

If ψ_ε is P and $\dot{\psi}$ is N then $\delta = -0.590\,\psi_\varepsilon - 0.839\,\dot{\psi} - 0.117$

If ψ_ε is P and $\dot{\psi}$ is Z then $\delta = -0.481\,\psi_\varepsilon - 1.081\,\dot{\psi} - 0.061$

If ψ_ε is P and $\dot{\psi}$ is P then $\delta = -0.659\,\psi_\varepsilon - 1.311\,\dot{\psi} + 0.781$

whereas the rulebase of the chemotaxis tuned Sugeno style fuzzy autopilot was taken as:

If ψ_ε is N and $\dot{\psi}$ is N then $\delta = -0.129\,\psi_\varepsilon - 0.058\,\dot{\psi} + 0.143$

If ψ_ε is N and $\dot{\psi}$ is Z then $\delta = -0.101\,\psi_\varepsilon - 0.029\,\dot{\psi} + 0.148$

If ψ_ε is N and $\dot{\psi}$ is P then $\delta = -0.025\,\psi_\varepsilon - 0.061\,\dot{\psi} - 0.044$

If ψ_ε is Z and $\dot{\psi}$ is N then $\delta = -0.028\,\psi_\varepsilon - 0.109\,\dot{\psi} + 0.146$

If ψ_ε is Z and $\dot{\psi}$ is Z then $\delta = -0.202\,\psi_\varepsilon + 0.085\,\dot{\psi} - 0.005$

If ψ_ε is Z and $\dot{\psi}$ is P then $\delta = 0.149\,\psi_\varepsilon + 0.145\,\dot{\psi} - 0.067$

If ψ_ε is P and $\dot{\psi}$ is N then $\delta = -0.124\,\psi_\varepsilon + 0.067\,\dot{\psi} - 0.041$

If ψ_ε is P and $\dot{\psi}$ is Z then $\delta = -0.084\,\psi_\varepsilon + 0.170\,\dot{\psi} + 0.160$

If ψ_ε is P and $\dot{\psi}$ is P then $\delta = 0.009\,\psi_\varepsilon + 0.035\,\dot{\psi} + 0.059$

A qualitative assessment of the autopilot responses was provided by the AUV models responses to a series of random course changes as shown in Figure 3.

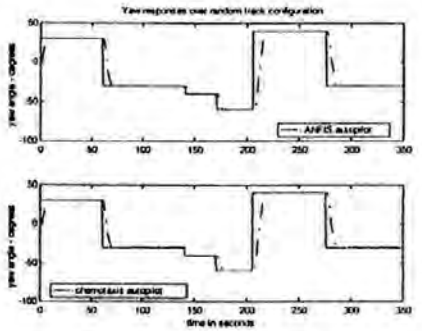


Figure 3. Autopilot responses to random track configuration

Figure 3 does not provide conclusive evidence of the ANFIS autopilot's superior performance over the chemotaxis autopilot, although it is apparent that the hybrid learning algorithm of the ANFIS technique leads to faster course-changing responses with less peak overshoot. Both autopilots perform well over the validation course and thus further performance indices were sought to ascertain the superior autopilot for AUV course-changing control.

With respect to quantitative performance measures for each autopilot, as a means of measuring the course-changing accuracy and rudder activity the integral square error (ITSE) for the yaw error (ψ_ε) and the canard demand (δ_ε) are employed:

$$\psi_\varepsilon = \int_0^t (\psi_d - \psi_a)^2\,dt \qquad (10)$$

$$\delta_\varepsilon = \int_0^t (\delta_d - \delta_a)^2\,dt \qquad (11)$$

where ψ_d and δ_d represent desired yaw angle and canard demand (and $\delta_d = 0$), and ψ_a and δ_a represent actual yaw angle and canard demand respectively. To assess the speed of response of the control system the rise time (T_R) was calculated for each fuzzy autopilot, and the peak overshoot ($M_p(t)$) was calculated to assess the oscillatory nature of each response. Here rise time is taken as the time to reach 99 per cent of a desired 40° course change, i.e. 39.6°, and the peak overshoot is measured as a relative percentage of this 40° course change demand.

Testing each autopilot over a wider AUV speed envelope provided some insight into the robustness of each autopilot to speed parameter variations. Results pertaining to three speeds, 5 7.5 and 10 knots, are displayed in Table 2, where 7.5 knots corresponds to the tuning speed.

AUV model	ANFIS autopilot				Chemotaxis autopilot			
	ψ_ε deg^2	δ_ε deg^2	T_R secs	$M_p(t)$ %	ψ_ε deg^2	δ_ε deg^2	T_R secs	$M_p(t)$ %
5 Knots	4394.3	2470.8	8.75	2.95	5300.1	1649.1	11.62	3.38
7.5 Knots	3050.1	1469.7	6.47	1.93	3599.6	1152.0	7.48	4.96
10 knots	2393.1	1092.9	4.98	1.78	2787.6	911.4	5.71	5.84

Table 2. Yaw responses over a course changing manoeuvre of 40 degrees

When operating at 7.5 knots the autopilot designed using the ANFIS technique was 15.27% more accurate than that of the chemotaxis tuned autopilot. Additionally, the rise time and peak overshoot figures for the ANFIS tuned autopilot were lower than those of the chemotaxis tuned autopilot suggesting a faster and less oscillatory course-changing response.

Intuitively one would expect more sluggish AUV responses at 5 knots due to the diminished hydrodynamic forces exerted upon the canard control surfaces. Indeed this was borne out in the 5 knot simulations. Course-changing accuracy was again more evident with the ANFIS autopilot at this speed although an increase in canard control effort was noted. The ANFIS autopilot proved 17.09% more accurate than the chemotaxis autopilot at 5 knots.

Finally, the increased effectiveness of the canard control surfaces at the higher operating speed of 10 knots lead to more responsive AUV course-changing manoeuvres, the ANFIS autopilot being 14.15% more accurate than the

chemotaxis tuned autopilot with respect to course-changing error.

The results pertaining to the rise time and peak percentage overshoots for the two autopilots suggest that the ANFIS autopilot produced faster, less oscillatory control of the AUV yaw dynamics over the tested AUV speed envelope.

CONCLUSIONS

The work described in this paper demonstrates that course-changing autopilots for AUVs may be designed using Jang's ANFIS approach. It is important to note that in this study, the design of the autopilot is the result of a fusion of neural and fuzzy techniques. However, a distinction exists in that the autopilot itself is entirely fuzzy and the network style implementation of the working controller is merely a convenience.

The use of a 300 epoch tuning period in this work is meant to provide a direct comparison between tuning algorithms. However, due to the random search nature of the chemotaxis algorithm, extended tuning periods may be necessary to ensure that the resulting set of autopilot parameters are the globally optimal set.

ACKNOWLEDGMENTS

The authors gratefully acknowledge the Sea Systems Sector, DERA, Winfrith, for supplying the AUV model and for many helpful discussions concerning its implementation.

REFERENCES

Jang, J.-S. R. 1993. ANFIS : adaptive network-based fuzzy inference system, IEEE Transactions on Systems, Man and Cybernetics, Vol. 23, no. 3: 665–685.

Koshland, D.E. 1980. Bacterial chemotaxis in relation to neurobiology, Annual Review of Neuroscience, Vol. 3: 45–75.

Smith, S.M., Rae, G.J.S, Anderson, D.T. and Shien, A.M. 1993. Fuzzy logic control of an autonomous underwater vehicle. In Proceedings of [illegible] on Intelligent Autonomous Vehicles (Southampton, UK), 318–323.

Yuh, J. 1990. A neural network controller for underwater robotic vehicles, IEEE Journal of Oceanic Engineering, Vol. 15, no.3: 161–166.

AUTHORS' BIOGRAPHIES

Mr Paul J Craven received a BSc in Mathematics with First Class Honours from the University of Plymouth in 1995. Since graduating, he has been employed in the Institute of Marine Studies of the University of Plymouth as a research student. His current research interest being the application of artificial intelligence approaches to the identification, modeling and control of unmanned underwater vehicles (UUVs). To date he has published 14 technical papers on various aspects of UUV control and identification.

Dr Robert Sutton holds the degree of BEng (Tech) in Engineering Production, and MEng and PhD in Control Engineering from the University of Wales. He served 16 years in the Royal Navy as a Commissioned Officer and left in 1991 to take up an appointment with the University of Plymouth in the Institute of Marine Studies (IMS). Currently, he is a Reader in Control Systems Engineering and is responsible for all Control Engineering activities in both the IMS and the School of Manufacturing, Materials and Mechanical Engineering. His research interests lie in the application of advanced Control Engineering methods and Artificial Intelligence techniques to industrial and marine dynamic systems.

Dr Yong-Ming Dai took a BEng (Hons) in Mechanical Engineering specializing in Internal Combustion Engines in China, in 1984. He spent his postgraduate studies at the University of Newcastle upon Tyne, and obtained a PhD in Marine Technology in 1992. He is presently a lecturer in the Institute of Marine Studies, University of Plymouth and, as module leader, responsible for a number of core modules mainly in the subject area of marine engineering systems and design. His main research interests are in marine diesel engines, mathematical modeling and computer simulation of marine dynamic systems and intelligent control systems.

NEUROFUZZY CONTROL OF A NON-LINEAR MULTIVARIABLE SYSTEM

P J Craven*, R Sutton* and M Kwiesielewicz#

* - University of Plymouth, United Kingdom
# - Technical University of Gdansk, Poland

INTRODUCTION

In order that the potentially disastrous effects of climatic change be accurately monitored, the ocean depths must first be explored. The corresponding requirement for improvements in data collection techniques, as described in [1], highlights the need for robust and reliable navigation, guidance and control (NGC) systems for AUVs which are becoming increasingly important in order to fully exploit the vastness of the oceans.

Traditional control methodologies often fail to compensate for the inherent coupling between AUV degrees of freedom and thus some interest in multivariable approaches to AUV autopilot design has grown in recent years. This work discusses the development of a new multivariable autopilot for simultaneous control of the sway and yaw modes of an AUV and highlights the requirement for the minimization of cross-coupling effects between these modes.

Multivariable control system design techniques are well established in the control theory literature (see [2]), and are usually employed when two or more degrees of freedom must be controlled simultaneously or when there is a need to compensate for the interaction between degrees of freedom in a plant. Various advanced multivariable control techniques have been applied directly to the problem of unmanned underwater vehicle autopilot design with varying amounts of success [3], [4], [5], [6], and [7].

The remainder of this paper begins in section 2 which provides an introduction to the sophisticated AUV model used within this study. Section 3 discusses the development of a novel architecture for the tuning of multivariable AUV autopilots to control multiple degrees of freedom simultaneously. The AUV model is used in section 4 to show the effectiveness of the developed multivariable autopilot over SISO AUV control strategies. Finally, concluding remarks are given in section 5.

MODELLING THE AUV DYNAMICS

Figure 1 shows the complete control authority of the AUV model. However, it should be noted that for this study the upper and lower canard rudders, situated at the bow of the AUV are used to control the yaw dynamics and sway control is achieved via the upper and lower stern rudders (both limited to +/- 25.2 degrees). Dimensionally, the model represents an underwater vehicle which is 7 m long, approximately 1 m in diameter and has a nominal displacement of 3600 kgs.

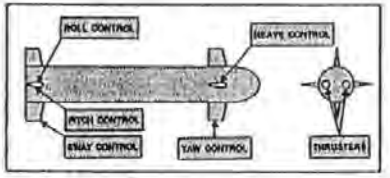


Figure 1. The complete control authority of the AUV

If required, the equations of motion describing the dynamic behaviour of the vehicle in the yaw and sway modes can be written in state-space form [8] :

$$E\dot{x} = Fx + Gu \qquad (1)$$

where :

$$E = \begin{bmatrix} (m - Y_{\dot{v}}) & -Y_{\dot{r}} & 0 & -(Y_{\dot{p}} + mZ_q) & 0 \\ -N_{\dot{v}} & (I_z - N_{\dot{r}}) & 0 & -N_{\dot{p}} & 0 \\ 0 & 0 & 1 & 0 & 0 \\ -(K_{\dot{v}} + mZ_q) & -K_{\dot{r}} & 0 & (I_x - K_{\dot{p}}) & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$F = \begin{bmatrix} Y_{uv}U & (Y_{ur} - m)U & 0 & Y_{up}U & 0 \\ N_{uv}U & N_{ur}U & 0 & N_{up}U & 0 \\ 0 & 1 & 0 & 0 & 0 \\ K_{uv}U & (K_{ur} + mZ_q)U & 0 & K_{up}U - mgBG & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$G = \begin{bmatrix} Y_{uu\delta_{sm}}U^2 & Y_{uu\delta_{bm}}U^2 & 0 & 0 & 1 & 1 \\ N_{uu\delta_{sm}}U^2 & N_{uu\delta_{bm}}U^2 & l_4 & -l_4 & l_v & -l_v \\ 0 & 0 & 0 & 0 & 0 & 0 \\ K_{uu\delta_{sm}}U^2 & K_{uu\delta_{bm}}U^2 & \gamma_4 & -\gamma_4 & \gamma_v & -\gamma_v \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$u = \begin{bmatrix} \delta_{sm-upper} & \delta_{sm-lower} & 0 & 0 & \delta_{bow-upper} & \delta_{bow-lower} \end{bmatrix}^T$$

and the state variables are V, R, ψ, P, ϕ. To implement the non-linear AUV equations of motion, use is made of an AUV MATLAB / Simulink simulation model supplied by the Defence Evaluation and Research Agency (DERA), Sea Systems Sector, Winfrith, the model having been validated against standard DERA non-linear hydrodynamic code using tank test data and an experimentally derived set of hydrodynamic coefficients from the Southampton Oceanography Centre's AUTOSUB vehicle.

MULTIVARIABLE AUTOPILOT DESIGN

By extending the ANFIS structure of Jang [9] to accommodate multiple outputs, a novel approach for the tuning of multivariable fuzzy autopilots using a neural network architecture and a hybrid learning rule is derived. If it is assumed that the fuzzy inference system (FIS) under consideration is of multivariable form then the fuzzy rule-based algorithm may be represented in the first order Sugeno form as shown below (for two functional outputs):

Rule 1 :If x is A_1 and y is B_1
then $f_{11} = p_{11}x + q_{11}y + r_{11}$ and $f_{12} = p_{12}x + q_{12}y + r_{12}$
: : : : : :
Rule n :If x is A_n and y is B_n
then $f_{n1} = p_{n1}x + q_{n1}y + r_{n1}$ and $f_{n2} = p_{n2}x + q_{n2}y + r_{n2}$

By encoding such a fuzzy rulebase as an adaptive network structure the proposed architecture for two Sugeno rules with two outputs per rule can be taken as that of Figure 2. Obviously, the actual architecture employed in this study is far more complex.

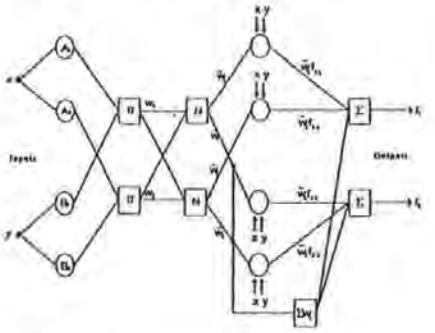


Figure 2. The multivariable autopilot structure.

In general, node functions in the same layer are of the same function family as described by the following:

Layer 1: Every *ith* node in this layer is an adaptive node with a node output defined by:

$$O_{1,i} = \mu_{A_i}(x) \qquad (2)$$

where x is the input to the general node and A_i is the fuzzy set associated with this nodes pertaining to input x_i. In other words, outputs of this layer are the membership values of the premise part. Here the membership functions for A_i can be any appropriate parameterized membership functions. Here $A_i, B_i, \ldots, P_i$ are characterized by the bell function of equation (3), where $\{a_i, b_i, c_i\}$ is the parameter set. Parameters in this layer are referred to as *premise parameters*.

$$\mu_{A_i}(x) = \frac{1}{1 + \left[\left(\frac{x - c_i}{a_i}\right)^2\right]^{b_i}} \qquad (3)$$

Layer 2: Every node in this layer is a fixed node labelled Π, which multiplies the incoming signals and outputs the product or T-norm operator result, e.g.

$$O_{2,i} = w_i = \mu_{A_i}(x_1) \times \mu_{B_i}(x_1) \times \ldots\ldots \times \mu_{S_i}(x_n), \quad i = 1,2,\ldots,n \qquad (4)$$

Each node output equals the *firing strength* of a rule.

AUVs destined for long duration mission scenarios for three main reasons. One is a non technical matter which needs to be resolved as soon as possible and revolves around the legal responsibilities and liabilities for AUVs deployed at sea. Without doubt, these are issues that need to be clarified, however, they are clearly outside the scope of this paper. The second reason relates to the limited endurance capacity of the power systems. Most AUVs depend on battery power which, inevitably, gives them limited range. To overcome this problem there needs to be a major breakthrough in battery technology or a shift to other power sources such as non air-breathing diesel plant. The third and final restricting factor is associated with the capabilities of the current generation of onboard navigation, guidance and control (NGC) systems.

In order for this type of vehicle to be truly autonomous, it is necessary for it to possess a reliable and robust NGC system. A key element of the NGC system is the control subsystem which is responsible for maintaining the vehicle on course. Several advanced control engineering concepts including H∞ [4], sliding mode [5], and predictive control [6] theories are being employed in the design of course-changing autopilots and have met with varying degrees of success.

AI approaches are now also being introduced into the design process. Autopilots formulated using fuzzy logic and artificial neural network (ANN) methods have been reported and shown to be endowed with commendable robustness properties. Encouraged by such results, this paper considers the development of a course-changing autopilot based on the innovative neuro-fuzzy methodology of Jang [7] known as the adaptive network-based fuzzy inference system (ANFIS) which was successfully employed to produce a control strategy for the classical inverted pendulum problem.

With the ANFIS approach implementation of the controller design differs in form from the more usual ANN in that it is not fully connected, and not all the weights or nodal parameters are modifiable. Essentially the fuzzy rule base is encoded in a parallel fashion so that all the rules are activated simultaneously so as to allow network training algorithms to be applied. As in Jang's original work, a back propagation algorithm is used to optimise the fuzzy sets of the antecedents in the ANFIS architecture and a least squares procedure is applied to the linear coefficients in the consequent terms.

For performance assessment purposes, the ANFIS design is compared to that of a simulated annealing (SA) tuned autopilot and a proportional-derivative controller. In the design of the SA tuned autopilot, an adaptive structure similar to the ANFIS architecture is employed. During its tuning process, however, the input data are only fed forward through the network in order to generate an error function. The SA algorithm is then applied to optimise the premise parameters.

The paper also considers the performance of a guidance subsystem based on a line of sight (LOS) algorithm.

Throughout this simulation study, the test bed platform for evaluating the control algorithms is a generic AUV dynamic model which is currently being employed by the Defence Evaluation and Research Agency (DERA), Sea Systems Sector, Winfrith, in a number of their integrated control systems design studies.

2. Modelling the autonomous underwater vehicle dynamics

Figure 1 shows the complete control authority of the AUV model. However, it should be noted that for this study the upper and lower canards are the only surfaces used to control its yaw dynamics. Dimensionally, the model represents an underwater vehicle which is 7 m long, 1 m in diameter and has a displacement of 3600 kg.

The equation of motion describing the dynamic behaviour of the vehicle in the lateral plane is as follows [8] :

$$E\dot{x} = Fx + Gu \qquad (1)$$

where :

$$E = \begin{bmatrix} (m - Y_{\dot{v}}) & -Y_{\dot{r}} & 0 & -(Y_{\dot{p}} + mZ_G) & 0 \\ -N_{\dot{v}} & (I_z - N_{\dot{r}}) & 0 & -N_{\dot{p}} & 0 \\ 0 & 0 & 1 & 0 & 0 \\ -(K_{\dot{v}} + mZ_G) & -K_{\dot{r}} & 0 & (I_x - K_{\dot{p}}) & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$F = \begin{bmatrix} Y_{uv}U & (Y_{ur} - m)U & 0 & Y_{up}U & 0 \\ N_{uv}U & N_{ur}U & 0 & N_{up}U & 0 \\ 0 & 1 & 0 & 0 & 0 \\ K_{uv}U & (K_{ur} + mZ_G)U & 0 & K_{up}U & -mgBG \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$G = \begin{bmatrix} Y_{uu\delta_{up}}U^2 & Y_{uu\delta_{lo}}U^2 & 0 & 0 & 1 & 1 \\ N_{uu\delta_{up}}U^2 & N_{uu\delta_{lo}}U^2 & \ell_\phi & -\ell_\phi & \ell_\psi & -\ell_\psi \\ 0 & 0 & 0 & 0 & 0 & 0 \\ K_{uu\delta_{up}}U^2 & K_{uu\delta_{lo}}U^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$u = \begin{bmatrix} \delta_{up} \\ \delta_{lo} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

and the state variables are V, R, ψ, P and ϕ. A nomenclature for the AUV parameters can be found in Appendix A. As mentioned earlier, to implement equation (1) use is made of an AUV MATLAB / Simulink simulation model supplied by DERA, Winfrith. The model having been validated against standard DERA non-linear hydrodynamic code using tank test data and an experimentally derived set of hydrodynamic coefficients from the Southampton Oceanography Centre's AUTOSUB vehicle. In addition, the MATLAB / Simulink model structure also takes into account the dynamic behaviour of the canard actuators by describing them as first order lags with appropriate limiters.

As can be seen in equation (1) the roll cross-coupling dynamics are included. However, control of the roll channel is not considered here.

3. Neuro-fuzzy autopilot design

As discussed above, the fuzzy controller design used in this study is based on the ANFIS. Functionally, there are almost no constraints on the membership functions of an adaptive network except piecewise differentiability. Structurally, the only limitation on network configuration is that it should be of the feed-forward type. Due to these minimal restrictions, the adaptive network's applications are immediate and immense in various areas.

If it is assumed that the fuzzy inference system under consideration has multiple inputs and one functional output (f) then the fuzzy rule-based algorithm may be represented in the first order Sugeno form as shown below [9] :

Rule 1 : If x is A_1 and y is B_1 then $f_1 = p_1 x + q_1 y + r_1$
Rule 2 : If x is A_2 and y is B_2 then $f_2 = p_2 x + q_2 y + r_2$
: : : : : : :
: : : : : : :
Rule n : If x is A_n and y is B_n then $f_n = p_n x + q_n y + r_n$

The corresponding ANFIS architecture being shown in Figure 2.

The node functions in the same layer are of the same function family as described by the following :

Layer 1 : Every *ith* node in this layer is an adaptive node with a node output defined by :

$$O_{1,i} = \mu_{A_i}(x) \qquad (2)$$

where x is the input to the general node and A_i is the fuzzy set associated with this node. In other words, outputs of this layer are the membership values of the premise part. Here the membership functions for A_i can be any appropriate parameterised membership functions. Here A_i is characterised by the generalised bell function :

$$\mu_{A_i}(x) = \frac{1}{1 + \left[\left(\frac{x - c_i}{a_i}\right)^2\right]^{b_i}} \qquad (3)$$

where $\{a_i, b_i, c_i\}$ is the parameter set. Parameters in this layer are referred to as *premise parameters.*

Layer 2 : Every node in this layer is a fixed node labelled Π, which multiplies the incoming signals and outputs the product or T-norm operator result, for example :

$$O_{2,i} = w_i = \mu_{A_i}(x) \times \mu_{B_i}(y), \quad i = 1, 2 \qquad (4)$$

Each node output represents the *firing strength* of a rule. (In fact, any other T- norm operators that perform the fuzzy AND operation can be used as the node function in this layer).

Layer 3 : Every node in this layer is a fixed node labelled N. The *ith* node calculates the ratio of the *ith* rules' firing strength to the sum of all rules' firing strengths :

$$O_{3,i} = \bar{w}_i = \frac{w_i}{w_1 + w_2}, \quad i = 1, 2 \qquad (5)$$

For convenience, outputs of this layer are called *normalised firing strengths.*

Layer 4 : Every *ith* node in this layer is an adaptive node with a node function :

$$O_{4,i} = \bar{w}_i f_i = \bar{w}_i (p_i x + q_i y + r_i) \qquad (6)$$

where $\bar{w}_i$ is the output of Layer 3 and $\{p_i, q_i, r_i\}$ is the parameter set. Parameters in this layer are referred to as *consequent parameters.*

Layer 5 : The single node in this layer is labelled $\sum$, which computes the overall output as the summation of incoming signals :

$$O_{5,i} = \text{overall output} = \sum_i \bar{w}_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \qquad (7)$$

Thus an adaptive network that has exactly the same function as a Sugeno fuzzy model may be constructed.

4. Simulated annealing tuned autopilot structure

The structure of the SA tuned autopilot is similar to that described in section 3 and depicted in Figure 3. However, there are dissimilarities. In this case, the nodes in Layer 4 are static and therefore are not modifiable. Also during the tuning process, input data are only fed forward through the network in order to generate an error function. The SA algorithm is then applied to optimise the premises.

5. The training algorithms

5.1 The Hybrid Learning Rules

Rewriting the premise membership function equation (3) as :

$$\mu_{ij}(x) = \frac{1}{1 + \left[\left(\frac{x - c_{ij}}{a_{ij}}\right)^2\right]^{b_{ij}}} \tag{8}$$

then equation (8) now represents the *jth* membership function on the *ith* input universe of discourse.

Therefore the learning rule for a general parameter may be described as follows :

$$\Delta\alpha_{ij} = -\eta \cdot \sum_{n=1}^{P} \frac{\partial E_n}{\partial O^i_{1n}} \cdot \frac{\partial O^i_{1n}}{\partial \alpha_{ij}}$$

$$= -\eta \cdot \sum_{n=1}^{P} \frac{\partial E_n}{\partial O^i_{1n}} \cdot \frac{\partial O^i_{2n}}{\partial O^i_{1n}} \cdot \frac{\partial O^i_{1n}}{\partial \alpha_{ij}} \cdot \tag{9}$$

Hence, as shown in Appendix B the learning rules for each individual parameter are :

$$\Delta a_{ij} = -\eta \cdot \sum_{n=1}^{P} \frac{\partial E_n}{\partial O^i_{2n}} \cdot \frac{\partial O^i_{2n}}{\partial O^i_{1n}} \cdot \left[\frac{2b_{ij}a_{ij}^{(2b_{ij}-1)}\left(x - c_{ij}\right)a_{ij}^{2b_{ij}}\left(x - c_{ij}\right)^{2b_{ij}}}{\left\{a_{ij}^{2b_{ij}}\left(x - c_{ij}\right)^{2b_{ij}} + \left(x - c_{ij}\right)^{2b_{ij}}a_{ij}^{2b_{ij}}\right\}^2}\right] \tag{10}$$

$$\Delta b_{ij} = -\eta \cdot \sum_{n=1}^{P} \frac{\partial E_n}{\partial O^i_{2n}} \cdot \frac{\partial O^i_{2n}}{\partial O^i_{1n}} \cdot \left[\frac{2a_{ij}^{2b_{ij}}\left(x - c_{ij}\right)^{2b_{ij}}a_{ij}^{2b_{ij}}\left(x - c_{ij}\right)^{2b_{ij}} \ln\left[\left|\frac{a_{ij}}{x - c_{ij}}\right|\right]}{\left\{a_{ij}^{2b_{ij}}\left(x - c_{ij}\right)^{2b_{ij}} + \left(x - c_{ij}\right)^{2b_{ij}}a_{ij}^{2b_{ij}}\right\}^2}\right] \tag{11}$$

$$\Delta c_{ij} = -\eta \cdot \sum_{n=1}^{P} \frac{\partial E_n}{\partial O^i_{2n}} \cdot \frac{\partial O^i_{2n}}{\partial O^i_{1n}} \cdot \left[\frac{-2b_{ij}a_{ij}^{2b_{ij}}\left(c_{ij} - x\right)^{(2b_{ij}-1)}a_{ij}^{2b_{ij}}\left(-\left(x - c_{ij}\right)\right)^{2b_{ij}}}{\left\{a_{ij}^{2b_{ij}}\left(-\left(x - c_{ij}\right)\right)^{2b_{ij}} + \left(c_{ij} - x\right)^{2b_{ij}}a_{ij}^{2b_{ij}}\right\}^2}\right] \tag{12}$$

The fuzzy consequent parameters being updated using a recursive least squares method and is also described in Appendix B.

5.2 Simulated annealing

The main problematic aspect of gradient descent based learning algorithms for optimisation problems, such as backpropagation, is the tendency for these methods to spend long periods of time in the neighbourhood of poor or sub-optimal local minima on the error hypersurface. A technique which can be employed to overcome this shortcoming is SA which was first introduced by Kirkpatrick, et al [10].

SA is a very efficient random search method for global minimisation [11], [12]. This method is based on an analogy between the global minimisation problem and that of determining the lowest energy state of a physical system.

Kirkpatrick, et al [10] adapted an algorithm taken from the statistical mechanics field for converging to one of many possible cooled or low energy states. Energies of this algoritlun are described by a Boltzman probability distribution such that the probability of any given energy E is an exponentially decreasing function of E. Thus, if a new matrix of parameters θ, which have been perturbed from an initially assumed solution by a randomly generated amount, lead to an improved performance of the system under consideration, then they are accepted and the process is repeated. However, if this new matrix leads to a worsened performance of the system it may be occasionally accepted with probability $P(\theta)$ such that :

$$P(\theta) = \exp\left[\frac{-E(\theta)}{kT}\right] \qquad (13)$$

where $E(\theta)$ is the energy associated with the state θ, k is the Boltzman's constant and T is a temperature parameter.

For a thermodynamic system, it can be demonstrated both by theoretical arguments and experimental verification that the most effective strategy for obtaining a global minimum energy state requires the temperature to be cooled slowly. Indeed, provided the cooling process is performed sufficiently slow, the system will by-pass locally stable states to reach one which is a global minimum. Thus, in analogous systems, the temperature T is allowed to decay during training according to the following equation :

$$T = \frac{T_o}{1 + an} \qquad (14)$$

where T_o is the initial temperature, a is a constant which governs the decay rate and n is the training epoch.

Hence, SA may be considered to consist of three distinct phases :

(i) a random search step;

(ii) a minimisation stage, and

(iii) a stopping rule.

The random search step is basically the iterative generation of random matrices in a domain $S(\theta_k)$, constituted by neighbouring matrices associated to the current matrix θ_k by :

$$\theta_k = \begin{bmatrix} \theta_k^{11} & \cdots & \theta_k^{1n} \\ \vdots & \cdots & \vdots \\ \theta_k^{m1} & \cdots & \theta_k^{mn} \end{bmatrix} \qquad (15)$$

$$\theta_k \in \Re^{n}$$

The minimisation stage consists of applying a local minimisation routine to some of the sampled matrices. Whilst, the stopping rule terminates the algorithm provided there is sufficient evidence that the global minimum has been detected within the limits of a specified accuracy or some explicit iteration number is reached.

In summary, the SA algorithm can be expressed as in Table I.

6. Proportional-derivative autopilot design

Whilst a number of advanced approaches are now being applied to the control of AUVs, there are still a number of craft employing autopilot designs based on variants of the classical proportional-integrate derivative (PID) controller.

Such a controller can be represented by a non-interacting structure in the time domain as :

$$u(t) = K\left[e(t) + \frac{1}{T_i}\int e(t)\,dt + T_d\,\frac{d\,e(t)}{dt}\right] \qquad (16)$$

or, alternatively, in the Laplace transform domain by :

$$\frac{u}{e}(s) = K\left[1 + \frac{1}{T_i s} + T_d\,s\right] \qquad (17)$$

For this study, it was expedient to use a PD controller as a benchmark and, therefore, results in equation (17) being reduced to :

$$\frac{u}{e}(s) = K\left[1 + T_d\,s\right] \qquad (18)$$

Using a Ziegler and Nichols method [13], the parameters K and T_d were obtained. From a practical viewpoint, it is customary to limit the bandwidth of the derivative action to approximately half a decade. By restricting the derivative bandwidth, the benefits of derivative action are maintained without too much amplification of high frequency noise. Thus, the PD autopilot design is taken as :

$$\frac{u}{e}(s) = \frac{0.007\left[1 + 0.643\,s\right]}{\left[1 + 0.117\,s\right]} \qquad (19)$$

7. Way point guidance by line of sight

LOS guidance algorithms are more usually associated with airborne missile systems. Nevertheless, here based on the work of Healey and Lienard [14], guidance of the AUV model is realised by a heading command to the steering mechanism of the vehicle to approach the LOS between its present position and the next way point. Ideally the design of the guidance and control systems should be fully integrated. Although this is not the case in this study, it is assumed the autopilot has a sufficiently large bandwidth to track the commands from the guidance subsystem.

As consideration is only being given to the vehicle operating in the lateral plane, then the LOS is defined as the horizontal angle given by :

$$\psi_d = \tan^{-1}\left[\frac{(y_k - y(t))}{(x_k - x(t))}\right] \qquad (20)$$

where $\{x_k, y_k\}$ are the way points stored in the mission planner of the AUV and $\{x(t), y(t)\}$ are the current co-ordinates as shown in Figure 4. It is pointed out in Reference [14] care must be exercised to ensure the proper quadrant is selected when programming the guidance law.

In order to inform the AUV that it has reached a given way point, a "circle of acceptance" which has a radius ρ_{CA} is defined :

$$\rho^2(t) = [x_k - x(t)]^2 + [y_k - y(t)]^2 < \rho_{CA}^2 \qquad (21)$$

For this study, the radius was taken as being three times the length of the AUV.

8. Results and discussion

Tuning algorithms based upon the ANFIS and SA techniques have been developed and applied to the task of tuning course-changing fuzzy autopilots for an AUV. This section discusses the performance of each autopilot in a qualitative and quantitative manner and makes comparisons to a traditional PD autopilot. For completeness, results are presented which detail autopilot robustness to parameter variations and sea current disturbances, and generalisation capability to course-changing demands which were not used as training data in the tuning process.

Tuning of the fuzzy autopilot parameters took place over a series of positive and negative course-changing demands of 40° at a surge velocity of 7.5 knots. Sufficient time intervals were given between consecutive course-changing demands to enable the AUV translational and rotational motions to stabilise, and thus ensure that each course-change was applied at similar initial conditions. This method was employed to effect symmetry within the final membership functions and rules of the neurally tuned fuzzy autopilots.

The ANFIS technique was applied to the task of tuning both the premise and consequent parameters of a nine rule fuzzy autopilot of Sugeno form, using the hybrid learning rule outlined in section 5.1.

The resulting linear fuzzy rules are of the form :

If ψ_e is N and $\dot{\psi}$ is N then δ = -1.46 ψ_e -0.89 $\dot{\psi}$ +0.66

If ψ_e is N and $\dot{\psi}$ is Z then δ = -0.49 ψ_e -0.88 $\dot{\psi}$ -0.05

If ψ_e is N and $\dot{\psi}$ is P then δ = -0.51 ψ_e -0.89 $\dot{\psi}$ -0.69

If ψ_e is Z and $\dot{\psi}$ is N then δ = -0.45 ψ_e -0.11 $\dot{\psi}$ +0.79

If ψ_e is Z and $\dot{\psi}$ is Z then δ = 0.00 ψ_e 0.00 $\dot{\psi}$ +0.00

If ψ_e is Z and $\dot{\psi}$ is P then δ = -0.45 ψ_e -0.11 $\dot{\psi}$ -0.79

If ψ_e is P and $\dot{\psi}$ is N then δ = -0.51 ψ_e -0.89 $\dot{\psi}$ +0.69

If ψ_e is P and $\dot{\psi}$ is Z then δ = -0.49 ψ_e -0.88 $\dot{\psi}$ +0.05

If ψ_e is P and $\dot{\psi}$ is P then δ = -1.46 ψ_e -0.89 $\dot{\psi}$ -0.66 (22)

In addition, the SA algorithm outlined in section 5.2 was applied to the task of tuning only the premise parameters of the same nine rule Sugeno fuzzy autopilot, whilst the consequent parameters remained fixed as equally spaced singletons upon the output universe of discourse.

The tuning regime resulted in the following fuzzy rule base :

If ψ_e is N and $\dot{\psi}$ is N then δ = +25.00

If ψ_e is N and $\dot{\psi}$ is Z then δ = +18.75

If ψ_e is N and $\dot{\psi}$ is P then δ = +12.50

If ψ_e is Z and $\dot{\psi}$ is N then δ = +6.25

If ψ_e is Z and $\dot{\psi}$ is Z then δ = 0.00

If ψ_e is Z and $\dot{\psi}$ is P then δ = -6.25

If ψ_e is P and $\dot{\psi}$ is N then δ = -12.50

If ψ_e is P and $\dot{\psi}$ is Z then δ = -18.75

If ψ_e is P and $\dot{\psi}$ is P then δ = -25.00 (23)

A qualitative assessment of autopilot responsiveness was provided by the AUV model's responses to a series of positive and negative course-changing demands of varying magnitude, as illustrated in Figure 5. Such a track configuration was deemed necessary to assess the ability of each autopilot to generalise to course-changes for which they had not been tuned. Figure 6 shows the corresponding canard demands to the track configuration of Figure 5.

The ANFIS tuned fuzzy autopilot displayed the most accurate response over this particular track configuration, with smaller rise times and no steady-state course error. Collectively these responses suggest that the hybrid learning rule employed by the ANFIS tuning regime was the most effective at producing a tuned autopilot with reduced off-course error with good generalisation qualities.

As a means of quantifying off-course error and course-changing control effort the following performance measures were adopted :

$$\psi_e = \int_{t_1}^{t_2} (\psi_d - \psi_a)^2 \, dt \qquad (24)$$

$$\psi_\delta = \int_{t_1}^{t_2} (\delta_d - \delta_a)^2 \, dt \qquad (25)$$

which represent the integral of square error (ISE), where ψ_d and δ_d represent desired yaw angle and canard demand respectively, and ψ_a and δ_a represent actual yaw angle and canard demand respectively. Additionally, to assess the response speed of the AUV model and the oscillatory nature of each AUV response to a particular autopilot, figures pertaining to the rise time (T_R) and the percentage peak overshoot (M_p (t)) were collected. Rise time is considered here as the time to reach 99% of the course-change demand and the percentage peak overshoot is calculated as a relative percentage of the course-change demand.

As parameter tuning took place at 7.5 knots, the robustness of each autopilot was assessed by simulating AUV responses to a course-change of 40° at surge velocities of 5, 7.5 and 10 knots. Thus Table 2 contains the results pertaining to these three surge velocities. Additionally, data are supplied for off-course error, canard effort, rise time and percentage peak overshoot.

When operating at 7.5 knots the autopilot tuned using the SA technique was 2.49% more accurate than the traditional PD autopilot. This illustrates that the SA tuned autopilot produces a reduced off-course error for the 40° course-changing demand, as shown in Figure 7. Moreover, the autopilot designed using the ANFIS technique is 32.35% more accurate than the PD autopilot and 30.62% more accurate than the SA tuned fuzzy autopilot.

At 5 knots the effectiveness of the canard control surfaces is significantly diminished due to the reduced hydrodynamic forces acting on them. Intuitively, one would anticipate more sluggish AUV response times as a consequence of this situation. Indeed this is borne out in the results of Table 2. The SA tuned autopilot again produced course-changing responses which were 5.89% more accurate than the PD autopilot. However, the ANFIS tuned fuzzy autopilot proved to be 29.14% and 33.32% more accurate than the SA and PD autopilots respectively.

Conversely, the increased effectiveness of the canard control surfaces at 10 knots lead to much sharper AUV responses. Figure 7 clearly illustrates the superior performance of the ANFIS tuned autopilot at 10 knots, with a reduction in off-course error of 32.89% and 34.73% over the SA and PD autopilots respectively.

Use was made of a LOS guidance algorithm (as detailed in Section 7) in order that each autopilot's effectiveness could be examined in the presence of sea current disturbances. Each autopilot was applied to the course-changing track configuration of Figure 8. To highlight the effect of sea current disturbances, results are presented in Figure 8 for no added disturbances and for a sea current disturbance of one metre per second (in the direction of the positive y-North axis) in Figure 9. Additionally, the canard demand responses of Figure 10 are given to provide some illustration of the control effort demands of each autopilot over the course-changing track.

The circles depicted in Figure 8 represent the target way points which are stored within the mission planner prior to mission embarkation. Therefore, the autopilot which traces the shortest distance between these way points (whilst visiting all way points) is considered the most effective at the course-changing task. At present there is a great deal of research interest concerning AUV guidance in the NGC community. Typical methods of AUV guidance include dead-reckoning position fixing using speed calculations based upon motor revolutions and torque data taken from the on-board data logger, and also intermittent surfacing of the vehicle to obtain GPS position fixes at pre-specified way points.

From Figure 8 it is clearly evident that the ANFIS tuned fuzzy autopilot is the most effective at following the specified set of target way points. The course-following error incurred by employing the ANFIS tuned autopilot with no current disturbance is considerably less than that of the SA and PD autopilots. Indeed, the inclusion of the current disturbance, as shown in Figure 9, serves to reinforce the superiority of the ANFIS tuned autopilot, even though the course-following capability of the PD autopilot can be seen to be improved by the addition of the disturbance.

From the canard demands (illustrated in Figure 10) for these LOS experiments it is clear that the ANFIS tuned fuzzy autopilot performs more effectively with and without the addition of sea current disturbances. A crisp canard angle is maintained throughout the mission when employing the ANFIS autopilot, as opposed to the demands of the SA tuned autopilot and the PD autopilot which appear highly oscillatory and show sustained periods of saturation.

Finally, comparison of the figures for canard demands in Table 2 highlight that the ANFIS tuned autopilot consistently greater amounts of canard effort for the course-changing task of 40°. However, it is clear from the LOS guidance simulations that the canard demands of the ANFIS tuned autopilot are significantly more effective than those of the SA and PD autopilots in achieving the desired course. This highlights that the control surface contour is more optimally shaped than for the SA and PD autopilots, and thus a more judicious amount of control effort is applied.

9. Conclusions

This paper has discussed the tuning of a fuzzy autopilot for course-changing control of an AUV using a neural network architecture and two neural algorithms. The resulting autopilots remain purely fuzzy as parameter tuning is conducted off-line. From the result presented it may be concluded that the ANFIS approach provides a viable autopilot design solution in the presence of environmental disturbances and changing vehicle surge velocities.

The LOS algorithm presented herein is used as a means of effecting AUV guidance. For the purposes of these simulations it is assumed that knowledge of the AUV global co-ordinates is constantly available underwater.

10. Acknowledgements

The authors wish to thank the DERA, Sea Systems Sector, Winfrith, for supplying the AUV model and their continuing support throughout this work.

11. References

1. D Goodrich, *Marine Foresight Report*, Department of Trade and Industry (May 1997).

2. C P Summerhayes, R Coles, B Wheeler, M Baker, D S Cronan, R Burt, G Griffiths, N Veck, D Anderson, H Young and M Murphy, 'Report of the Marine Technology Foresight Panel Working Group on Exploitation of Non-Living Marine Resources', *Underwater Technology*, Vol 22, No 3, pp 103-122 (1997).

3. J G Hawley, 'Diesel Engine Operation on Synthetic Atmospheres for Underwater Applications', PhD Thesis, University of Exeter, UK (February 1993).

4. C Silvestre and A Pascoal, 'Control of an AUV in the Vehicle and Horizontal Planes: System Design and Tests at Sea', *Proceedings of the Third International Symposium on Methods and Models in Automation and Robotics*, Vol 2, pp 463-468 (September 1996).

5. D R Yoerger and J J E Slotine, 'Robust Trajectory Control of Underwater Vehicles', *IEEE Journal of Oceanic Engineering*, Vol 10, pp 462-470 (October 1985).

6. M R Katebi and D Desanj, 'Integrated Predictive Control Design for Autonomous Underwater Vehicles', *Proceedings of the Thirteenth World Congress of IFAC*, Vol Q, pp 327-332, San Francisco, USA (June / July 1996).

7. J S R Jang, 'ANFIS: Adaptive Network-Based Fuzzy Inference System', *IEEE Transactions on Systems, Man and Cybernetics*, Vol 23, pp 665-685 (1993).

8. W B Marshfield, 'Submarine Data Set for use in Autopilot Research', *Technical Memorandum*, DRA/MAR TM (MTH) 92314, DRA Haslar (April 1992).

9. M Sugeno (ed), *Industrial Applications of Fuzzy Control*, North Holland, The Netherlands (1985).

10. S Kirkpatrick, C Gelatt and M Uecchi, 'Optimisation by Simulated Annealing', *Science*, Vol 220, pp 671-680 (1983).

11. E Aarts and J Korst, *Simulated Annealing and Boltzmann Machines*, John Wiley and Sons, New York (1989).

12. N E Collins, R W Eglese and B L Golden, 'Simulated Annealing: An Annotated Bibliography', *American Journal of Mathematical and Management Sciences*, Vol 8, pp 209-308 (1988).

13. K J Astrom, J Hagglund, C C Hang and W K Ho, 'Automatic Tuning and Adaptation for PID Controllers: A Survey', *Control Engineering Practice*, Vol 1, No 4, pp 699-714 (1993).

14. A J Healey and D Lienard, 'Multivariable Sliding Mode Control for Autonomous Diving and Steering of Unmanned Underwater Vehicles', *IEEE Journal of Oceanic Engineering*, Vol 18, pp 327-339 (July 1993).

APPENDIX A : Nomenclature of the AUV equation parameters

E, F, G	State equation matrices
m	Mass of AUV
P, R	Angular velocity components of rolling and yawing
U, V	Velocity components of surge and sway
ϕ, ψ	Angles of roll and heading
I	Moment of inertia
K, N	Moment components
Y	Force components
B	Centre of buoyancy
G	Centre of mass
K_{UP}	Dimensional hydrodynamic coefficients
$N_{UU\delta_{UP}}$ etc	Non-dimensional hydrodynamic coefficients
ℓ_ϕ	Roll moment arm length
ℓ_ψ	Yaw moment arm length
δ_{UP}, δ_{LO}	Upper and lower canard inputs

APPENDIX B : The Hybrid Learning Rule

The learning rule was based upon the hybrid learning rule of Jang [7]. The system is simulated using the dynamic model and data is collected across the trajectory. This training data is used to compare the system trajectory with the desired trajectory, and so form the error measure to be used for training of the adaptive network parameters. The error measure chosen was the integral square of heading error over time (ITSE) :

$$E = \sum\left[(\psi_d - \psi_a)^2\right] t \qquad \text{(B1)}$$

The parameters to be altered are the fuzzy parameters of both the premise and consequent layers. The hybrid learning rule employs the backpropagation method to update the fuzzy premise parameters and the recursive least squares method to update the fuzzy consequent parameters.

Writing the premise membership function as :

$$\mu_{ij}(x) = \frac{1}{1 + \left[\left(\frac{x - c_{ij}}{a_{ij}}\right)^2\right]^{b_{ij}}} \qquad \text{(B2)}$$

then equation (B2) now represents the jth membership function on the ith input universe of discourse, where a_{ij} governs the width of set, b_{ij} governs the flatness of the bell function and c_{ij} is the centre of the set on the ith input universe of discourse.

Therefore the learning rule for a general parameter may be described as follows :

$$\Delta\alpha_{ij} = -\eta \sum_{n=1}^{P} \frac{\partial E_n}{\partial O_{1n}} \cdot \frac{\partial O_{1n}}{\partial \alpha_{ij}}$$

$$= -\eta \sum_{n=1}^{P} \frac{\partial E_n}{\partial O_{2n}} \cdot \frac{\partial O_{2n}}{\partial O_{1n}} \cdot \frac{\partial O_{1n}}{\partial \alpha_{ij}} \qquad \text{(B3)}$$

where η is the learning rate, E_n is the error measure, P is the number of samples in the trajectory, and O_1 is the output of Layer 1. If the function $O_1 = f(\alpha_{ij})$ is differentiable then $\frac{\partial O_1}{\partial \alpha_{ij}}$ is a straightforward calculation. This was the motivation for choosing the set functions described by (B2). The main difficulty is in the calculation of $\frac{\partial E_n}{\partial O_{1n}}$. Considering the AUV model as the final layer in the adaptive network this calculation becomes simple for this layer :

$$\frac{\partial E_n}{\partial O_{1n}} = \frac{\partial}{O_{1n}} \left[\sum_{n=1}^{P} (T_n - O_{1n})^2 I + \rho (O_{4n})^2 \right] \qquad \text{(B4)}$$

$$= \sum_{n=1}^{P} -2(T_n - O_{1n}) I \qquad \text{(B5)}$$

There are no adaptable parameters in the ship model layer. The next layer, Layer 4, is the one that produces the defuzzified output. The computation of $\frac{\partial E_n}{\partial O_{4n}}$ uses a backpropagation of $\frac{\partial E_n}{\partial O_{3n}}$:

$$\frac{\partial E_n}{\partial O_{4n}} = \sum_{m=1}^{\#(5)} \frac{\partial E_n}{\partial O_{5n}^m} \cdot \frac{\partial O_{5n}^m}{\partial O_{4n}} \qquad \text{(B6)}$$

where $\#(5)$ is the number of nodes in Layer 5. Hence :

$$\frac{\partial E_n}{\partial O_{4n}} = \frac{\partial E_n}{\partial O_{3n}} \cdot \frac{\partial O_{3n}}{\partial O_{4n}} \qquad \text{(B7)}$$

as $\#(5)=1$. Now $\frac{\partial O_{3n}}{\partial O_{4n}}$ may be written as :

$$\frac{\partial O_{3n}}{\partial O_{4n}} = \frac{d\psi}{d\delta_n} \qquad \text{(B8)}$$

whereby the function relating ψ to δ_n is non-linear and the derivative (or Jacobian) is approximated by :

$$\frac{\partial O_{3n}}{\partial O_{4n}} = \frac{O_3(n) - O_3(n-1)}{O_4(n) - O_4(n-1)} \qquad \text{(B9)}$$

The only layer to be adapted using the backpropagation method is the first layer. Hence continuing the above process for each layer we arrive at the following learning rules for each individual parameter within Layer 1 :

$$\Delta a_{ij} = -\eta \sum_{n=1}^{P} \frac{\partial E_n}{\partial O_{2n}} \cdot \frac{\partial O_{2n}}{\partial O_{1n}} \cdot \left[\frac{2b_{ij} a_{ij}^{(2b_{ij}-1)} (x - c_{ij}) a_{ij}^{2b_{ij}} (x - c_{ij})^{2b_{ij}}}{\left\{ a_{ij}^{2b_{ij}} (x - c_{ij})^{2b_{ij}} + (x - c_{ij})^{2b_{ij}} a_{ij}^{2b_{ij}} \right\}^2} \right] \qquad \text{(B10)}$$

$$\Delta b_{ij} = -\eta \sum_{n=1}^{P} \frac{\partial E_n}{\partial O_{2n}} \cdot \frac{\partial O_{2n}}{\partial O_{1n}} \cdot \left[\frac{2a_{ij}^{2b_{ij}} (x - c_{ij})^{2b_{ij}} a_{ij}^{2b_{ij}} (x - c_{ij})^{2b_{ij}} \ln\left[\left| \frac{a_{ij}}{x - c_{ij}} \right| \right]}{\left\{ a_{ij}^{2b_{ij}} (x - c_{ij})^{2b_{ij}} + (x - c_{ij})^{2b_{ij}} a_{ij}^{2b_{ij}} \right\}^2} \right] \qquad \text{(B11)}$$

$$\Delta c_{ij} = -\eta \sum_{n=1}^{P} \frac{\partial E_n}{\partial O_{2n}} \cdot \frac{\partial O_{2n}}{\partial O_{1n}} \cdot \left[\frac{-2b_{ij} a_{ij}^{2b_{ij}} (c_{ij} - x)^{(2b_{ij}-1)} a_{ij}^{2b_{ij}} \left(-(x - c_{ij}) \right)^{2b_{ij}}}{\left\{ a_{ij}^{2b_{ij}} \left(-(x - c_{ij}) \right)^{2b_{ij}} + (c_{ij} - x)^{2b_{ij}} a_{ij}^{2b_{ij}} \right\}^2} \right] \qquad \text{(B12)}$$

It is given that if an adaptive networks output is linear in some of the networks parameters, then these linear parameters can be identified by the well documented least-squares method. Considering the case of one network output :

$$\text{output} = F\left(\bar{I}, S\right) \tag{B13}$$

where $\bar{I}$ is the vector of input variables and S is the set of parameters. If there exists a function H such that the composite function $H \bullet F$ is linear in some of the elements of S then these elements can be identified by the least-squares method. More formally, if the parameter set S can be decomposed into two sets

$$S = S_1 \oplus S_2 \tag{B14}$$

(where $\oplus$ represents direct sum) such that $H \bullet F$ is linear in the elements of S_2 then upon applying H to (B13) yields

$$H(\text{output}) = H \bullet F\left(\bar{I}, S\right) \tag{B15}$$

which is linear in the elements of S_2. Now given values of elements of S_1, P training data can be collected for input into (B15) which yields the matrix equation :

$$A\vartheta = B \tag{B16}$$

where ϑ is an unknown vector whose elements are parameters in S_2. This equation represents the standard linear least-squares problem and the best solution for ϑ, which minimises $\|A\vartheta - B\|^2$, is the least-squares estimator (LSE) ϑ^* :

$$\vartheta^* = \left(A^T A\right)^{-1} A^T B \tag{B17}$$

where A^T is the transpose of A and $\left(A^T A\right)^{-1} A^T$ is the pseudo-inverse of A if $A^T A$ is non-singular. The recursive LSE formula can be employed by letting the ith row vector of matrix A defined in (B16) be a_i^T and the ith element of B be b_i^T ; then ϑ can be calculated iteratively as follows :

$$\vartheta_{i+1} = \vartheta_i + S_{i+1} a_{i+1}\left(b_{i+1}^T - a_{i+1}^T \vartheta_i\right) \tag{B18}$$

$$S_{i+1} = S_i - \frac{S_i a_{i+1} a_{i+1}^T S_i}{1 + a_{i+1}^T S_i a_{i+1}} \tag{B19}$$

$$i = 0, 1, \ldots, P-1.$$

where the least-squares estimator ϑ^* is equal to ϑ_P . The initial conditions needed to bootstrap (B18) and (B19) are $\vartheta_0 = 0$ and $S_0 = \gamma I$ where γ is a positive large number and I is identity matrix of dimension $M \times M$.

Consequently the gradient descent method and the least-squares method have been combined to update the parameters in an adaptive network. Each epoch consists of a forward pass in which inputs are presented and the matrices A and B are calculated and the consequent parameters are updated via the recursive least-squares method. Additionally each epoch consists of a backward pass in which the derivative of the error measure with respect to each nodes output is propagated from the output to the input of the network architecture. At the end of the backward pass the parameters of the premise layer are updated by the gradient descent method.

TABLE 1 : Simulated Annealing Algorithm

1. Generate set of initial parameters and simulate system.
2. Make random changes to the parameters and re-simulate the system.
3. If performance improved then retain changes and re-apply.
4. If performance degraded then compute probability of accepting poorer parameters according to equations (13) and (14).
5. Generate random number in the range 0 - 1 and compare with probability computed at 4. If random number less then accept poorer parameters; otherwise reject.
6. Re-simulate and return to 3 until convergence.

TABLE 2 : Yaw responses over a course-changing manoeuvre of 40 degrees

AUV Model	Proportional + Derivative autopilot				Simulated Annealing autopilot				ANFIS autopilot			
	ψ_ε (°)^2	δ_ε (°)^2	T_{RS}	M_P (t)%	ψ_ε (°)^2	δ_ε (°)^2	T_{RS}	M_P (t)%	ψ_ε (°)^2	δ_ε (°)^2	T_{RS}	M_P (t)%
5 knots	124.74	17.28	20.77	1.14	117.39	18.71	19.98	0.61	83.18	33.86	9.76	2.86
7.5 knots	87.64	11.38	15.16	0.73	85.46	11.33	15.29	0.01	59.29	20.98	7.79	1.90
10 knots	70.51	8.66	14.90	0.42	68.58	7.79	14.30	0.01	46.02	13.90	7.51	1.32

LIST OF FIGURES

Fig 1

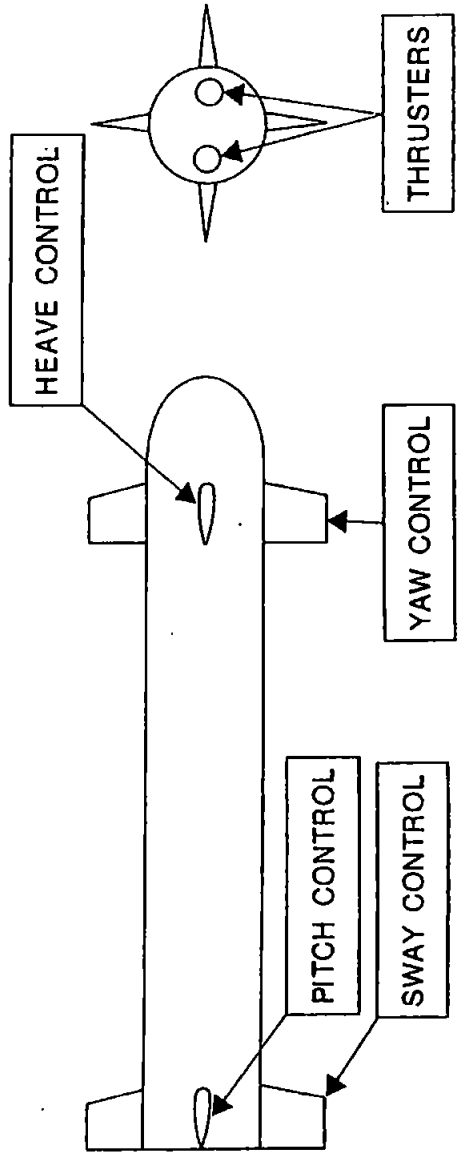


Fig 2

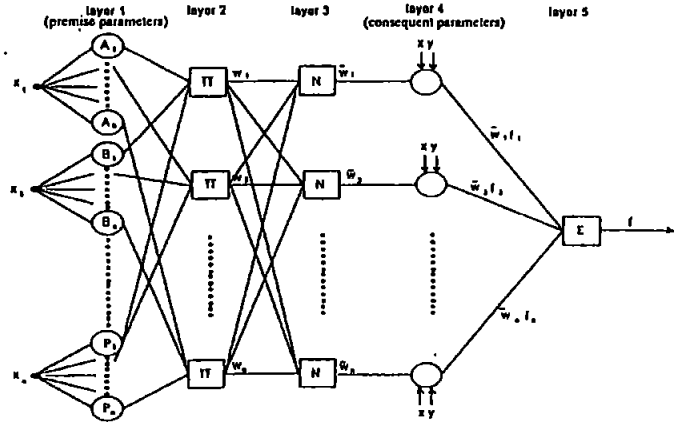

layer 1
(premise parameters)
layer 2
layer 3
layer 4
(consequent parameters)
layer 5
x y
TT
N
Σ

Fig 3

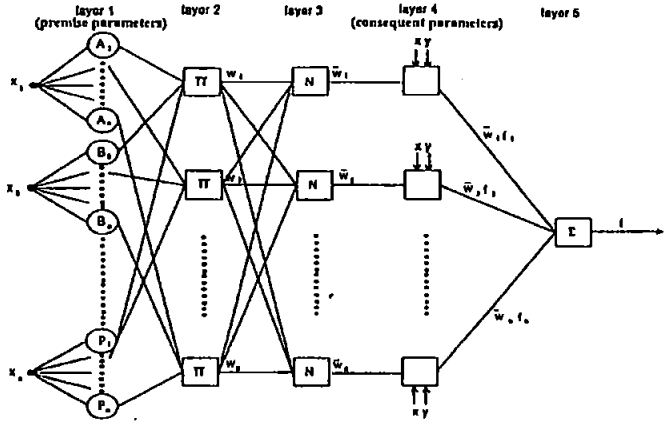

layer 1
(premise parameters)
layer 2
layer 3
layer 4
(consequent parameters)
layer 5
x y
TT
N
Σ

Fig4

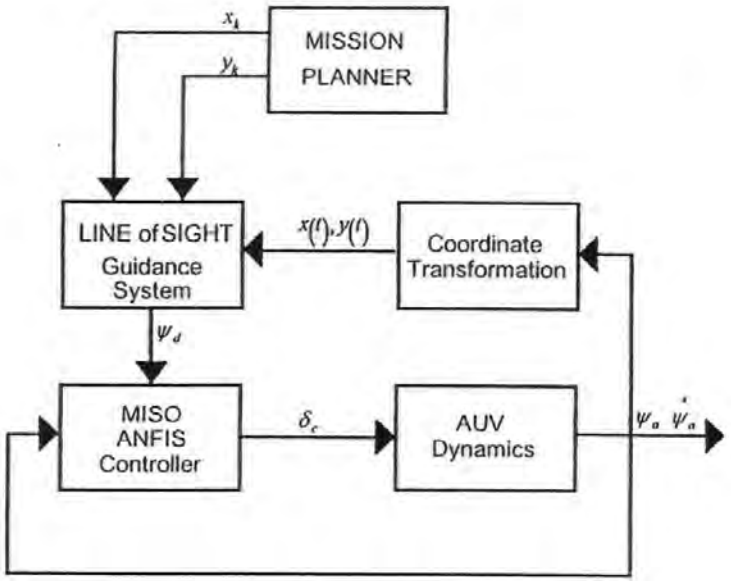

MISSION
PLANNER
LINE of SIGHT
Guidance
System
Coordinate
Transformation
MISO
ANFIS
Controller
AUV
Dynamics

Fig5

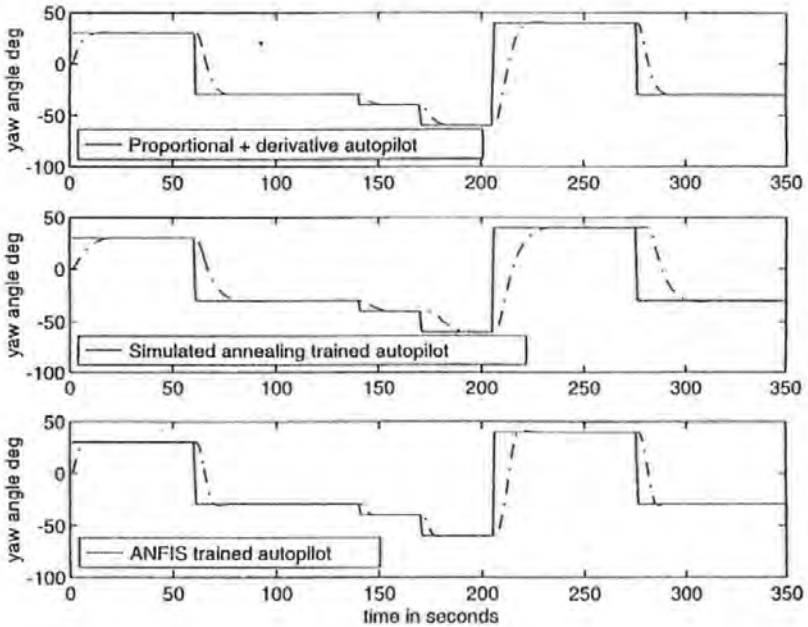

yaw angle deg
Proportional + derivative autopilot
Simulated annealing trained autopilot
ANFIS trained autopilot
time in seconds

Fig 6

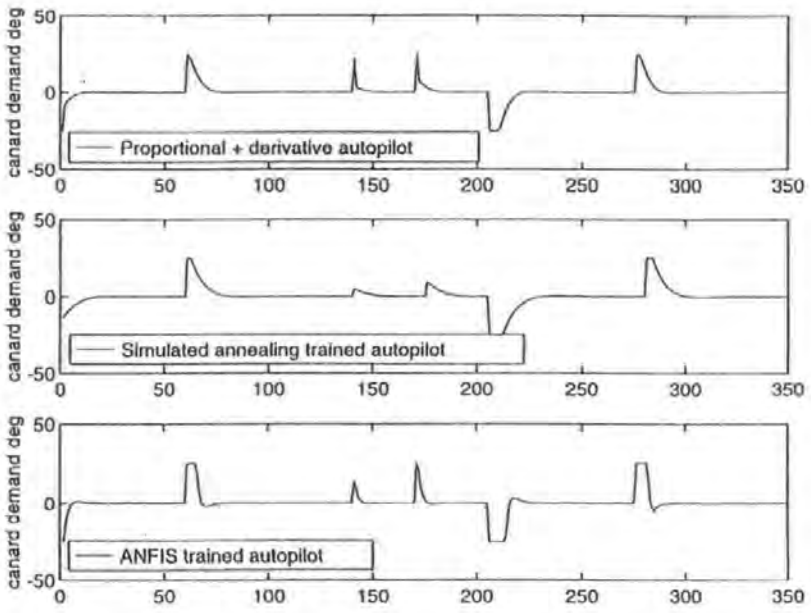

canard demand deg
50
0
-50
Proportional + derivative autopilot
0
50
100
150
200
250
300
350
canard demand deg
Simulated annealing trained autopilot
canard demand deg
ANFIS trained autopilot

Fig 7

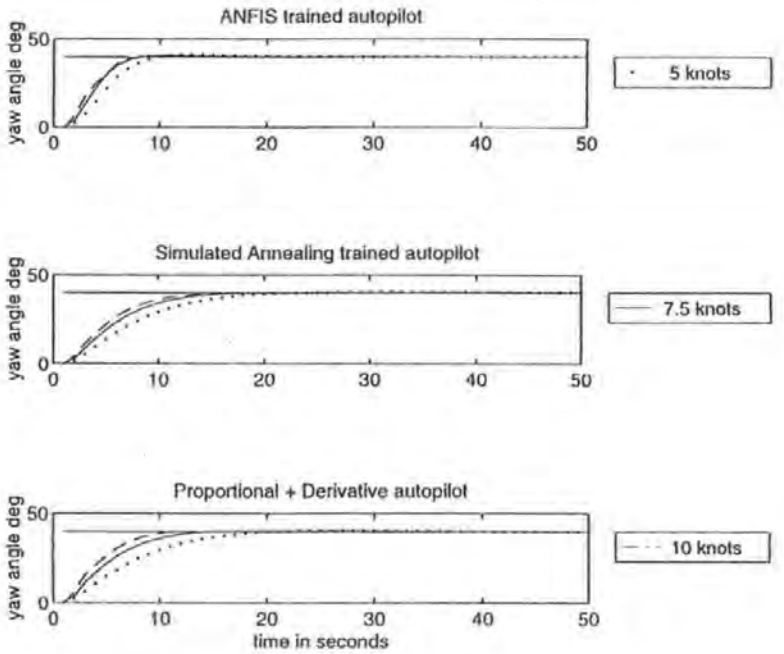

ANFIS trained autopilot
yaw angle deg
50
0
0
10
20
30
40
50
5 knots
Simulated Annealing trained autopilot
7.5 knots
Proportional + Derivative autopilot
10 knots
time in seconds

Fig 8

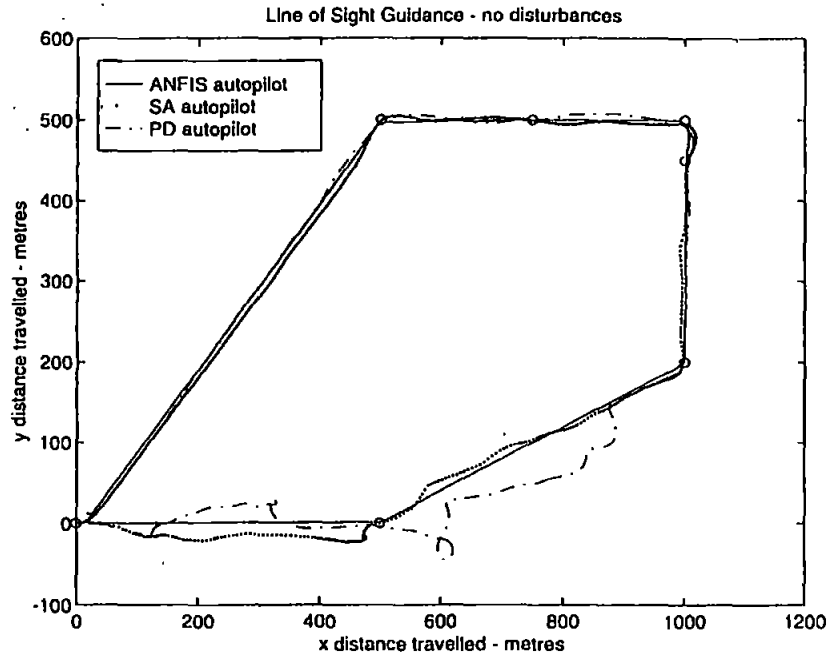
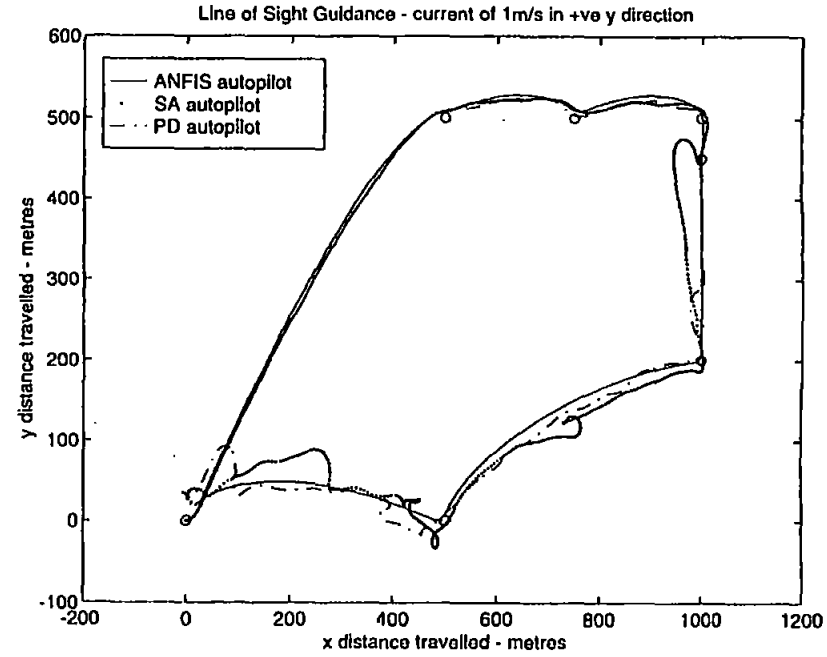
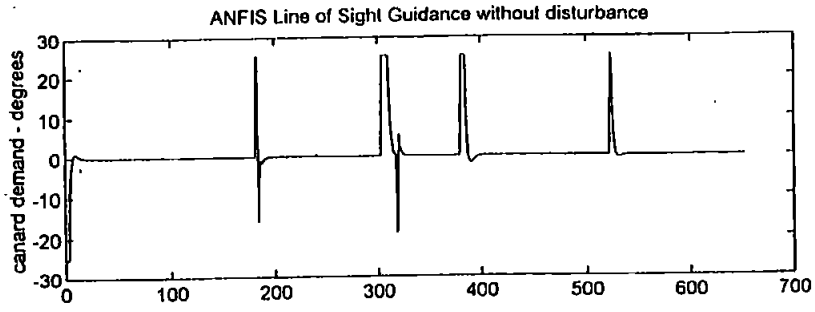
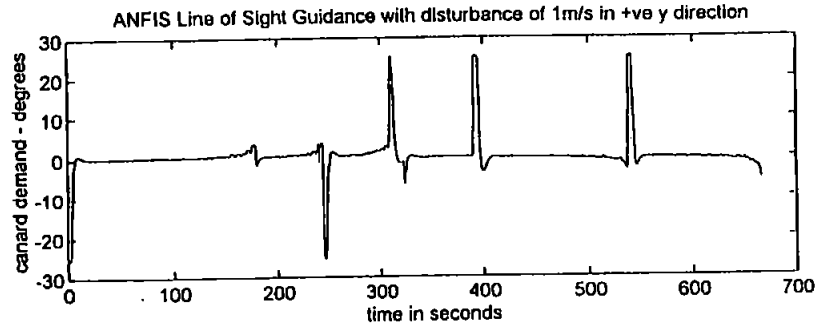
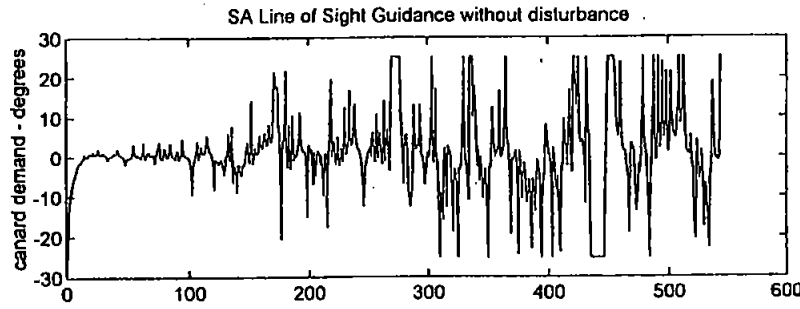

Line of Sight Guidance - no disturbances
ANFIS autopilot
SA autopilot
PD autopilot
y distance travelled - metres
x distance travelled - metres
600
500
400
300
200
100
0
-100
0
200
400
600
800
1000
1200

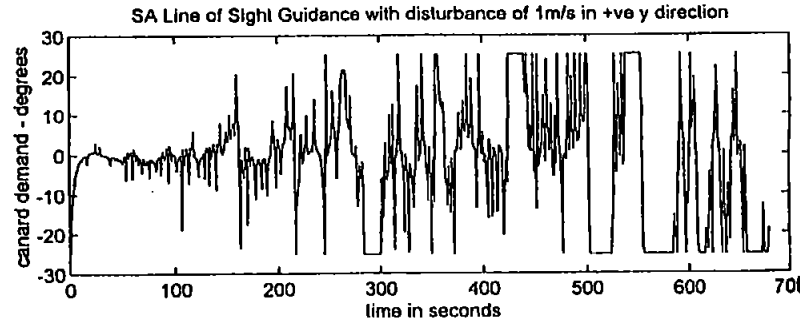
Fig 9

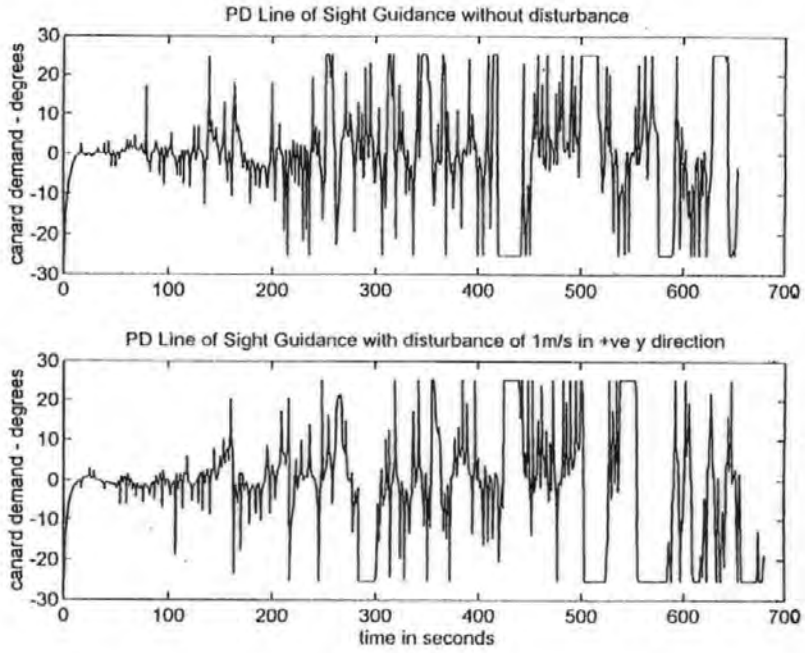

Line of Sight Guidance - current of 1m/s in +ve y direction
ANFIS autopilot
SA autopilot
PD autopilot
y distance travelled - metres
x distance travelled - metres
600
500
400
300
200
100
0
-100
-200
0
200
400
600
800
1000
1200


ANFIS Line of Sight Guidance without disturbance
canard demand - degrees
30
20
10
0
-10
-20
-30
0
100
200
300
400
500
600
700


ANFIS Line of Sight Guidance with disturbance of 1m/s in +ve y direction
canard demand - degrees
30
20
10
0
-10
-20
-30
0
100
200
300
400
500
600
700
time in seconds


SA Line of Sight Guidance without disturbance
canard demand - degrees
30
20
10
0
-10
-20
-30
0
100
200
300
400
500
600


SA Line of Sight Guidance with disturbance of 1m/s in +ve y direction
canard demand - degrees
30
20
10
0
-10
-20
-30
0
100
200
300
400
500
600
700
time in seconds


PD Line of Sight Guidance without disturbance
canard demand - degrees
30
20
10
0
-10
-20
-30
0
100
200
300
400
500
600
700
PD Line of Sight Guidance with disturbance of 1m/s in +ve y direction
canard demand - degrees
30
20
10
0
-10
-20
-30
0
100
200
300
400
500
600
700
time in seconds

Appendix B: Classical and Modern Control Strategies for Unmanned Underwater Vehicles

This work details classical and modern control strategies for unmanned underwater vehicles and is intended as a supplement to chapter 2.

B 1 Introduction

In the forthcoming text several references will be made to the "robustness qualities" of certain controllers. Therefore it is felt an explanation of this phrase is in order. During the design stage of a control algorithm some form of mathematical model which describes the dynamics of the vehicle to be controlled will be employed. When in actual operation and over a period of time, the characteristics of the plant will change from those for which the control system was originally designed. Also within a class of vehicles, differences in their performances will exist. Hence, a control system is said to have good robustness qualities provided it can cope with plant uncertainties whilst possessing noise and disturbance rejection properties.

B 2 Classical Techniques

B 2.2.1 Proportional-Integral-Derivative Control

The first autopilots to be called proportional (P) controllers were employed in the period 1930 to 1950 and used a heading error signal ($e(t)$) which was then used to adjust the steering mechanism of a ship. These controllers had no method of reducing overshoot and thus caused transient oscillations in the actual heading of the ship. The 1950s saw the introduction of the derivative (D) term, which improved the stability of the controlled vessel. Around the same period the integral (I) term was also implemented to produce counter thrust to external disturbances. The control action u_c produced by the complete PID controller is thus given by:

$$u_c = K_p\left[e(t)+\frac{1}{T_i}\int e(t)+T_d\frac{de(t)}{dt}\right] \tag{B1}$$

where K_p is the proportional gain of the controller, T_i is the integral time constant, T_d is the derivative time constant and $e(t)$ is the error signal.

The transfer function of the controller, in the Laplace domain, then becomes:

$$\frac{U_c(s)}{E_s(s)} = K_p\left[1+\frac{1}{T_i s}+T_d s\right] \tag{B2}$$

When controlling underwater vehicles the ideal requirement is to have high proportional gain in order to ensure rapid response to error and effective removal of steady state errors, however, these requirements tend to reduce the stability of the system and so a high derivative gain is also demanded. This allows these requirements to be met. The PID control strategy depends upon the availability of an accurate linear representation of the relevant non-linear vehicle dynamics. Such models take the form of transfer functions in the Laplace or z-domains (Eqn.(B3)), representing the input/output relationship to be controlled.

$$G(z) = \frac{\sum_{i=0}^{n} a_i z^{-i}}{\sum_{i=0}^{n} b_i z^{-i}} \tag{B3}$$

In Figure B1 a non-linear relationship is represented. At specific positions on this relationship, known as operating points (ops), the dominant non-linear dynamics can be linearized to give a transfer function (TF) at that point. Provided the operating points are chosen judiciously, it is possible to approximate the non-linear function as a series of linear transfer functions across the whole operating range.

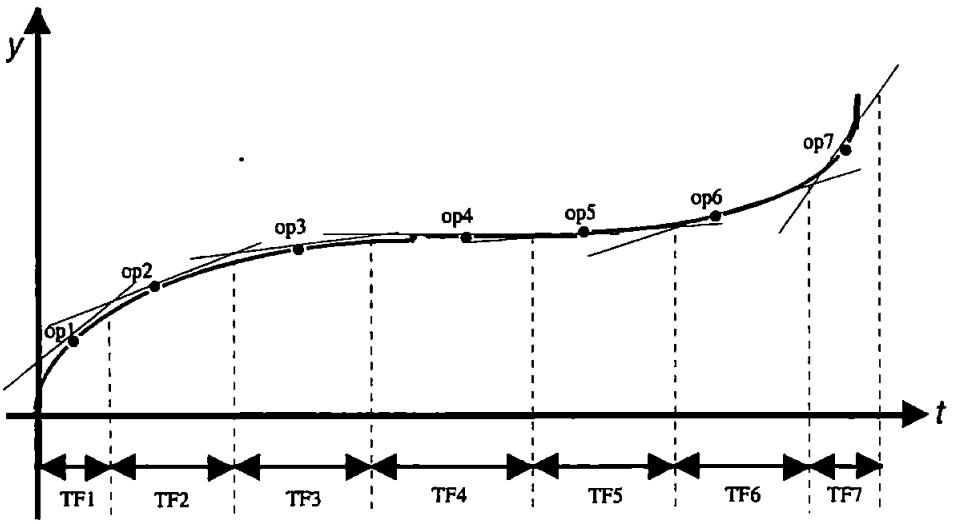


Figure B1: Linearization of a non-linear function.

For each of these operating regions where the linear relationship generally holds true, a PID controller, a PI controller or a PD controller can be designed to meet the required specifications. Methods to perform this are well-established [Shao (1988), [Daley and Gill (1986), Mandic et *al.* (1985)]. However, the further the system diverges from the operating point, the more likely it is that the non-linearities of the system will dominate and the controller will become ineffective.

Additionally, controllers based on the PD format are not able to prevent the high frequency rudder movements that often arise. Motora (1953) applied a low-pass filter to the output signal to prevent rudder oscillation, but this it was suggested would represent a loss in stability. Taylor (1995) reports that PID controllers have historically been employed in ship course keeping modes due to the manual complexity of the aforementioned tuning process when on full-scale sea trials. Da Chuna et *al.* (1995) employ this technique as a PI controller where the derivative term is neglected. This is

in fact a popular approach and has been utilized in many marine control applications, for example by Hsu et *al.* (1994).

UUV dynamics are such that it is usually very difficult to clearly define the dominant characteristics in order to derive a transfer function which is representative of the system under consideration.

B 2.2.2 Gain-Scheduling Control

Gain scheduling is often employed as a means of adjusting the gain terms of a linear controller to suit the current operating range. Depending on the current operating conditions, the gain terms of the controller will be adjusted to provide the best controller performance. To aid transition between operating regions, an interpolating function is often used. The structure of a typical gain-scheduled controller is shown in Figure B2 where a scheduling algorithm monitors the operating condition of the system and selects the controller parameters ($C_1 \ldots C_n$) best able to produce satisfactory

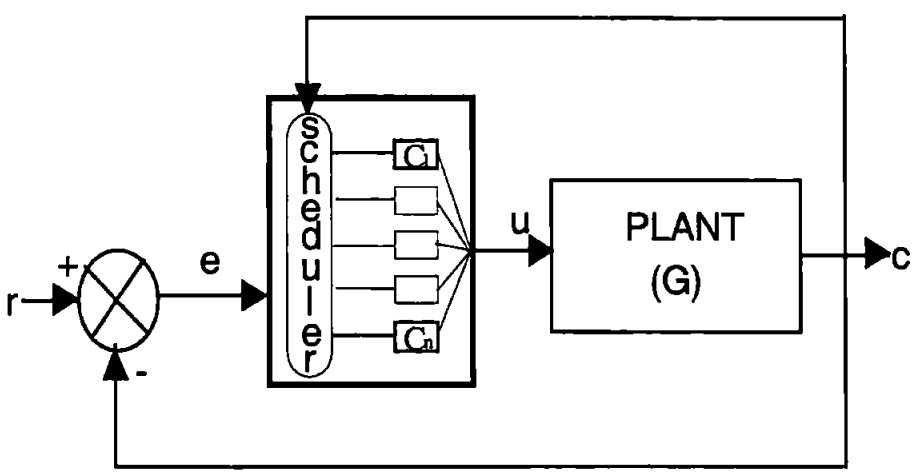


Figure B2: A gain-scheduled controller.

performance. If designed correctly the algorithm will also ensure a 'bumpless' transfer between the operating points.

The *MARIUS* (marine utility vehicle system) is an example of a successful implementation of a gain-scheduled controller in an UUV [Egeskov et *al.* (1994)]. This work documents four fundamental steps to the implementation of such a control strategy:

(i) Linearize the plant about a finite number of points,

(ii) Design linear controllers around each operating point,

(iii) Interpolate the parameters of the linear controllers of step (ii) to achieve adequate performance of the linearized closed-loop systems at all points where the plant is expected to operate. The interpolation is performed according to an external scheduling vector and the resulting family of linear controllers is referred to as a gain-scheduling controller,

(iv) Implement the gain-scheduled controller on the original non-linear plant.

The linear controllers outlined in step (ii) often take the classical PID form and are time consuming to develop. This technique requires thorough research into suitable controllers for each operating point and is not well documented among UUV control techniques, although a velocity algorithm for the subsequent implementation of a non-linear gain-scheduling controller has been reported [Kaminer et *al.* (1993)].

B 2.3 Modern Control Approaches

B 2.3.1 Adaptive Control

The past twenty-five years has seen a considerable amount of research in the application of control techniques to the problem of course keeping and manoeuvring of marine vehicles. Particular interest has been shown in the field of autopilots that adapt to dynamic and environmental changes, and consequently update the controller's

parameters to cope with these disturbances. The popularity of adaptive techniques concerns the poorly known hydrodynamic coefficients of the vehicle as well as the inherent non-linearities usually involved. Indeed, the majority of successful UUV control studies can be seen to include some form of adaptive control strategy. A comprehensive outline of adaptive control techniques and a brief review of the historical developments are given in Harris and Billings (1981). Additionally, Astrom and Wittenmark (1989) report the implementation of an adaptive autopilot for the ship course keeping task, based on a PID algorithm, the *Steermaster 2000.*

B 2.3.1.1 Model Reference Adaptive Control

The model reference adaptive control (MRAC) technique is one of the main approaches to adaptive control. A reference model gives the desired performance of the system. A feedback loop allows an error measure to be computed between the output of the system and the reference model. Thus based upon the error measure the parameters of the controller are adjusted to reduce this error measure. MRAC techniques were first designed to control the servo problem in deterministic continuous-time systems. Figure B3 shows a typical MRAC scheme.

The key problem with this approach is in the determination of the adjustment mechanism in order that a stable system is achieved. The MIT rule (Eqn.(B4)) was developed for the original adjustment of parameters in the MRAC system.

$$\frac{d\theta}{dt} = -\gamma e \frac{de}{d\theta} \tag{B4}$$

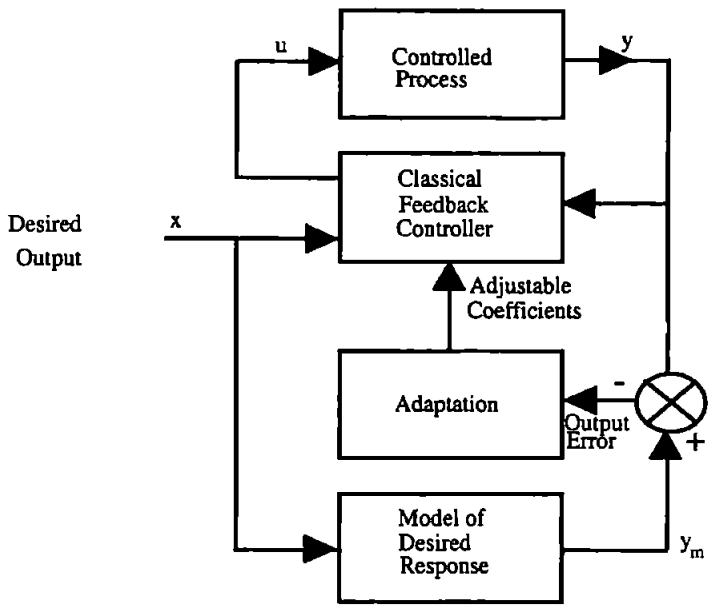


Figure B3: The model reference adaptive control scheme.

Van Amerongen (1982) applied the MRAC approach to the problem of ship course keeping and manoeuvring. This process employed a cost function (Eqn.(B5)) which was chosen to enable the derivation of optimal controller gains, via the use of a reference model (for comparison purposes).

$$J = \int_0^T \left(\varepsilon^2 + \lambda_1 \dot{\psi} + \lambda_2 \delta^2 \right) dt \qquad \text{(B5)}$$

This method was based on the assumptions that the modelled process was linear and external disturbances could be disregarded. Thus obvious criticisms of this approach are the linearity assumption concerning the vehicles dynamics, and the technique is only really viable when external conditions can be considered unimportant, which is almost never for a marine application. In the surveyed literature this technique is scarcely mentioned with respect to UUV control strategies. Shimmin and Lucas (1993)

document the technique and note the instability of such systems when the adaptation method relates controller parameter adjustments to system errors.

Da Cunha et *al.* (1995) designed an adaptive position controller for an ROV based on a recently developed output feedback variable structure control algorithm and was given the acronym VS-MRAC. The performance of the technique is evaluated by initially using simulation models and then full-scale sea trials utilizing an actual ROV. Results show that this technique consistently outperforms the conventional PI control technique, by providing more accurate position tracking and disturbance rejection.

B 2.3.1.2 Indirect Adaptive Control

Farrell and Clauberg (1993) report the implementation of an indirect adaptive control system on-board the AUV *Sea Squirt*. This system is designed in two layers, a standard adaptive layer and a 'supervisory logic' layer (to control the behaviour of the adaptive layer), as shown in Figure B4. Although this technique provides convergence of the modelling parameters towards their optimum tracking performance values, it takes no account of the varying mission to mission modularity and dynamics that such vehicles often encounter, thus suggesting the incorporation of some form of learning control strategy to model/estimate these variations. It is felt that this addition may also provide a means by which the vehicle can compensate for variations in hydrodynamics, effected by velocity variations.

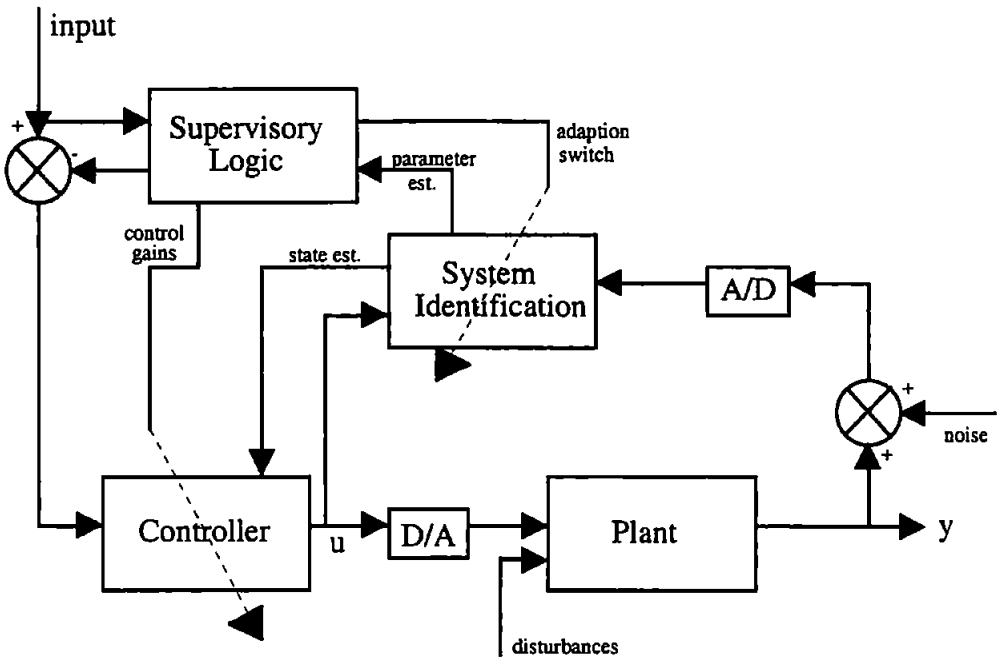


Figure B4: The indirect adaptive control scheme.

B 2.4 Self-Tuning Control

Self-tuning controllers can be used for processes with time-varying dynamics, by using a derived model of the process and environment to adjust the coefficients of the controller in order to satisfy a desired closed-loop system performance. A key publication by Astrom and Wittenmark (1989) states that self-tuning can be applied to a controller if, initially, control of the system is achieved by means of a conventional state or output feedback. Subsequently by adjusting the coefficients of the controller by means of a second, slower recursive loop better control can be achieved. Typically, algorithms have been implemented for adaptive ship steering and the dynamic positioning of drill ships [Boulton (1985)].

Triantafyllou and Grosenbaugh (1991) applied a multi input-multi output (MIMO) self-tuning controller to the difficult problem of automatic guidance of an AUV by manipulating thruster outputs to produce the desired translational and yaw velocities.

The self-tuning controller was used due to the lack of an accurate mathematical model of the AUV's open loop dynamics.

B 2.4.1 Explicit Self-Tuning Control

An explicit self-tuning control scheme is shown in Figure B5. In this control law all unknown dynamics are characterized by a time constant and steady-state gain between each velocity output and thruster input. The open-loop system can therefore be approximated by a square m by m matrix with a first order lag if, and only if

$$G_A^{-1}(z) = (z-1)B_0 + B_1, \tag{B6}$$

for some $m \times m$ matrices B_0 and B_1 where $|B_0| \neq 0$. In the case of $m = 1$, this definition reduces to a scalar first order lag.

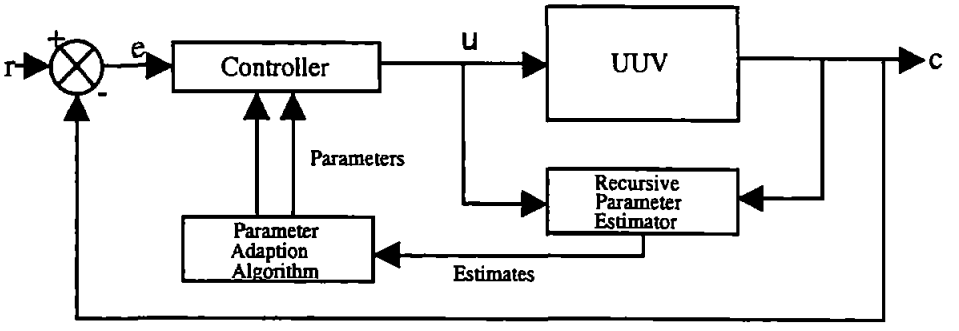


Figure B5: Explicit self-tuning control system.

It was established by Owens and Chotai (1983), using a PI controller for this type of system, that for minimum phase systems with fast enough sampling rates, the closed-loop system becomes decoupled into m separate single input-single output (SISO)

loops. It can also be shown that reference signals will be tracked with zero steady-state errors.

The self-tuning mechanism works by using the thruster input/velocity output data to update the model at each sampling instant. The model is of the form:

$$y^P(t+1) = Ay(t) + Bu(t) \tag{B7}$$

where $y^P(t+1)$ is the vector of predicted velocity outputs. The matrices A and B of Eqn.(B7) are used along with the closed-loop poles to calculate the PI control law, which takes the form:

$$B_0 = B^{-1} \tag{B8}$$

$$B_1 = B^{-1}(I - A) \tag{B9}$$

Finally the thruster inputs are determined and applied and the entire cycle is repeated. The α_i factors in the cost function (Eqn.(B10)) can be used to discount measurements that are known to be spurious or to give less weight to values of the model output when the algorithm is 'tuning-in' after a large change in the real system.

The versatility of this control scheme is demonstrated in Katebi and Byrne (1988), where it was employed to provide adaptation capability to a ship autopilot in adverse weather conditions and was customised to produce minimum variance to low-frequency heading variations and resistance to steering, again using the cost function of Eqn.(B10):

$$J = \frac{1}{n}\sum_{i=1}^{n} \alpha_i \left\| y(i) - \overset{p}{y}(i) \right\|^2 \tag{B10}$$

i.e. the weighted sum of the squared errors between the real system and model responses.

Yuh et *al.* (1990) used an adaptive control algorithm to control the pitch of an underwater robotic vehicle. It was concluded that because the initial controller parameters were derived from rough estimates of the system model, the adaptive controller showed large errors over the first few time steps, yet afterwards performed well. However, it was also found that the stability of the system could not be guaranteed for unmodelled system dynamics.

Goheen and Jeffreys (1989) implemented explicit 'one-shot self-tuning' in the *Seapup* and *PAP104* underwater vehicles, whereby first-order lags for the sway, yaw and surge velocities are used together with a PI controller. The resulting controller displays the expected robustness of PI control but does not adequately account for the inherent non-linearity of the UUVs, due to the underlying linearity of such self-tuning controllers. The one-shot controller is not an adaptive strategy either and as such cannot account for the time-varying dynamics of the UUV.

B 2.4.2 Implicit Self-Tuning Control

An implicit self-tuning controller identifies the parameters of the system directly and then uses this data in the control law to update the regulator. Consequently this method requires *a priori* knowledge of the system characteristics. Figure B6 shows the schematic diagram of an implicit self-tuning control system. The implicit self-tuning controller algorithms of Clarke and Gawthorp (1975) incorporated the controller output into a cost function for variance minimization of performance objectives. This approach was also employed by Lim and Forsythe (1983) for the design of an autopilot applied to ship control. The expression for the cost function output in order to minimize the expected variance is Eqn.(B11):

$$J = E\left[\left\|y(t+1) - r(t)\right\|^2 + \lambda u(t)^2\right] \tag{B11}$$

where: y is the actual output

r is the desired output

u is the control input

$E\{*\}$ is the form of the variance to be minimized, and

λ is a constant factor (a compromise between the control action and reference tracking against adaptation speed and is determined empirically).

Therefore, the generalized output Θ:

$$\Theta(t+1) = \sum_{i=0}^{ny} Fy(t-i) + \sum_{j=0}^{nr} H_j r(t-j) + \sum_{k=0}^{nu} G_k u(t-k) \tag{B12}$$

is identified and the control law follows directly, hence the implicit nature is that of:

$$u(t) = G_0^{-1}\left[\sum_{i=0}^{ny} Fy(t-i) + \sum_{j=0}^{nr} H_j r(t-j) + \sum_{k=1}^{nu} G_k (t-k)\right] \tag{B13}$$

as described in the paper by Goheen and Jeffreys (1990) who showed that second order models ($ny = nr = nu = 2$) produced the best closed-loop response with the *Seapup* simulation data, the reason being that the second order pitch and roll dynamics couple in other modes, unless the centre of rotation is chosen as the origin. This situation very rarely occurs in an UUV due to loading and weight distribution along the vehicle axes. This application somewhat omitted the finer points of the controller implementation.

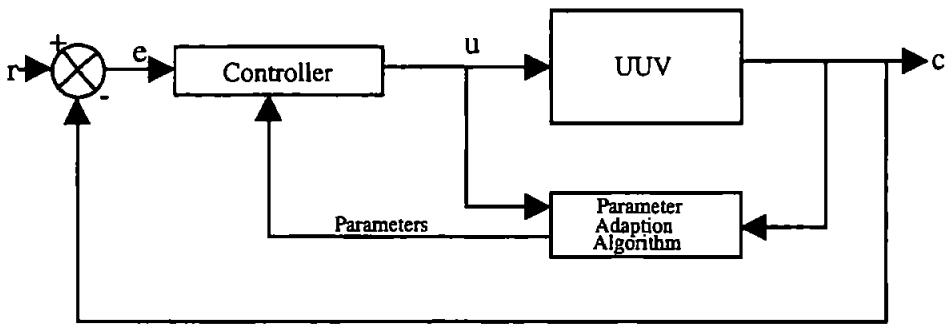


Figure B6: Implicit self-tuning control system.

Explicit self-tuning has the advantage of being computationally easier to implement as concerns the process algorithms than the implicit method, however the implicit method does not require the same degree of control knowledge by the operator to determine the correct closed-loop pole positions to meet the performance criteria.

B 2.5 Sliding Mode Control

A non-linear strategy that has been extensively applied to the UUV control problem is that of the sliding mode controller. A switching control law transforms the state trajectory of the plant onto a user chosen sliding surface in the state space, thus providing a technique that is robust to parametric uncertainty. Figure B7 shows a schematic of a sliding mode controller.

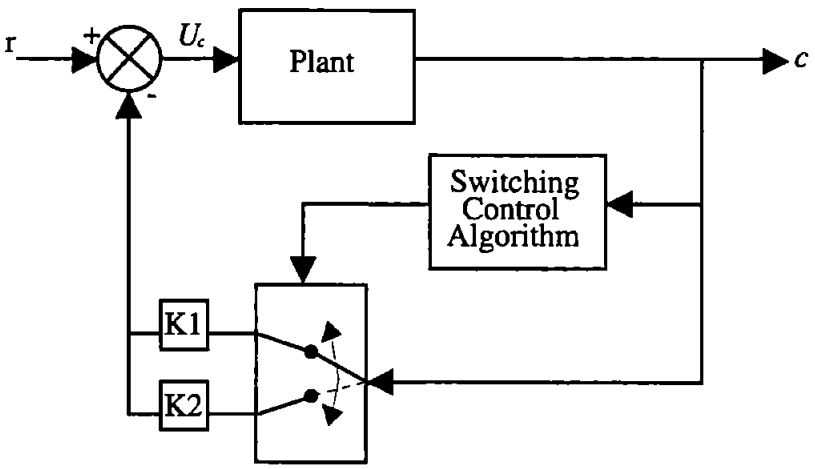


Figure B7: A schematic of a sliding mode controller.

The reader is referred to Cristi et *al.* (1990), who states that any UUV description based on a set of differential equations can only be approximate in its nature and therefore there are uncertainties in the model. This calls for a robust input u, of the form:

$$u = u + \overline{u} \tag{B14}$$

where u is determined on the basis of the nominal model and $\overline{u}$ compensates for deviations from ideal performance due to uncertainties.

Sliding control theory has been developed to apply to a large class of non-linear systems [Slotine (1983), Slotine(1985)]. The only restriction on the choice of sliding surface is that it has to be associated with stable dynamics, i.e. the following applies:

$$s(x(t)) = 0, \text{ for all } t > t \Rightarrow \lim_{t \to 0} x(t) = 0 \tag{B15}$$

for any initial conditions $x(t_0)$. The choice of a linear sliding surface being:

$$s(x) = s^T x \tag{B16}$$

for some vectors, $s^T x$ allows the use of pole placement techniques in the design of the non-linear controller. Using the defined Lyapunov function [Cristi et *al.* (1990)], the sliding surface $s(x) = 0$ is reached in a finite time by the condition:

$$\sigma \dot{\sigma} = -\eta_0^2(x)|\sigma(x)| \tag{B17}$$

$$\dot{\sigma} = -\eta_0^2(x) sign(\sigma). \tag{B18}$$

The dynamic matrix of the model and Eqn.(B16) are combined to obtain:

$$s^T(Ax + bu + f) = -h_0^2(x) sign(s) \tag{B19}$$

By knowing a bound h on the non-linearity for all conditions of x, the state described in Eqn.(B16) is satisfied by choosing the control input:

$$u = -(s^T b)^{-1} s^T Ax - \eta^2 (s^T b)^{-1} sign(s) \tag{B20}$$

$$u = \hat{u} + \overline{u}$$

As mentioned previously, due to the uncertainties in modelling an UUV, it is important to recognize that the feedback law u is composed of two parts. The first:

$$\hat{u} = -(s^T b)^{-1} s^T Ax \tag{B21}$$

is a linear feedback law based on the nominal model, whereas the second,

$$\bar{u} = -h^2\left(s^T b\right)^{-1} sign(s) \tag{B22}$$

is a non-linear feedback law, with its sign alternating between plus and minus according to which side of the sliding plane the system is currently located. Since u has to change its sign as the system crosses $\sigma(x) = 0$, the sliding surface has to be a hyperplane, i.e. the sliding surface dimension has to be one less than the state space. u is also largely responsible for driving the system onto, and keeping it on, the sliding plane $\sigma(x) = 0$ (where u = 0 as well). Provided that the gain has been chosen sufficiently large, u can provide the robustness required to handle random disturbances and unmodelled dynamics without compromise. This is achieved by designing the linear feedback law to ensure that the system has the desired dynamics on the sliding plane.

Yoerger and Slotine (1985) develop and apply a sliding mode controller to an ROV and document their simulation results on the experimental autonomous vehicle (*EAVE*). Although the authors reported successful implementation and control of *EAVE* they neglected the effects of pitch in their simulations, even though the heave and pitch channels are known to have quite influential cross-coupling effects.

Also Fossen (1991) and Fossen and Satagun (1991) report the implementation of multivariable sliding mode controllers to the positioning of an ROV. Simulation results demonstrate the controllers ability to achieve robustness to parameter uncertainty. Healey and Lienard (1993) have used this approach to control the speed, yaw and dive channels of an AUV individually. This work was then extended to develop a combined channel autopilot for the AUV. Results show robust performance for each of the individually controlled channels at low speed, and robust control in the combined autopilot for acceleration up to the chosen operational speed.

Trebi-Ollennu et *al.* (1995) provide a review of four robust multivariable control designs, including input-output linearization control with sliding mode depth control for an ROV. This technique is reported to result in a very robust controller but requires raw estimation of the bounds on the parametric uncertainties, which in itself is a non-trivial task.

B 2.6 H-Infinity (H_∞) Robust Control

The UUV operating environment is varied; the speed of such a vehicle may vary, payloads may be increased or decreased and the underlying mathematical models are inherently uncertain. Classical and optimal types of controllers are designed around a specific set of environmental conditions; the performance consequently degrades as these factors vary.

Robust control addresses these problems. It guarantees, given actuator limitations, a minimum level of performance and stability for a specified operation envelope, not only in terms of disturbances which impinge on the system but also for those due to uncertainties produced by the inadequacies of the mathematical representations of the UUV system. This method is embodied by the μ-synthesis procedure, essentially, an iterative process for the design of H_∞ controllers such that the closed loop adheres to the specified performance and stability criteria.

Given that G is a matrix representation of the plant and K is the matrix describing the controller, let the matrices T (the complementary sensitivity), S (sensitivity) and C (control sensitivity), be defined as in [Sharif et *al.* (1996)]:

$$T = GK(I + GK)^{-1} \tag{B23}$$

$$S = (I + GK)^{-1} \tag{B24}$$

$$C = K(I + GK)^{-1} \tag{B25}$$

Figure B8 depicts the schematic control scheme, where u is the control signal, and v represents disturbance and noise inputs, y physical quantities, e error signals, and x and z the uncertainty inputs/outputs. P is the nominal plant and Δ the block-diagonal representations of uncertainty, environmental and mathematical.

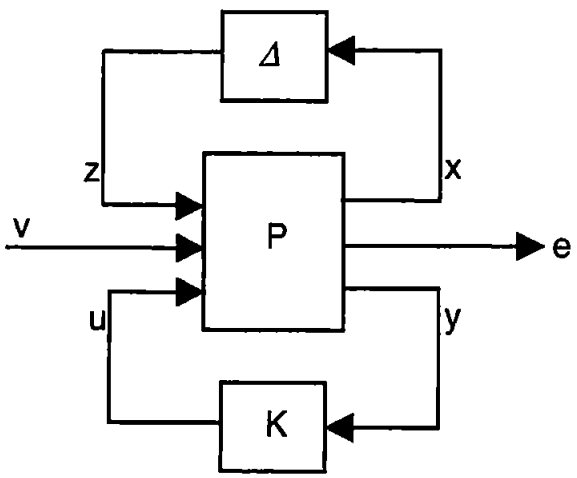


Figure B8: H-infinity robust control scheme.

If P is partitioned as shown in Eqn.(B26)

$$P = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} \tag{B26}$$

then let M denote the closed-loop function mapping v to e; this is known as the lower fractional transformation (Eqn.(B27)):

$$M = P_{11} + P_{12}K(I - P_{22}K)^{-1}P_{21} = F(P, K) \tag{B27}$$

The H_∞ optimization problem is then to minimize Eqn.(B27) over all stabilizing and realizable controllers, the constraints being defined, dependent upon engineering constraints, by weighting functions [Maciejowski (1989)]. Provided that the following H_∞ norm inequalities are satisfied, then robust stability and performance are assured. Here γ is a search variable and the weightings are described below:

$$\left\| \gamma W_p S \right\|_\infty < 1 \tag{B28}$$

$$\left\| \gamma W_\Delta T \right\|_\infty < 1 \tag{B29}$$

$$\left\| \gamma W_c C \right\|_\infty < 1 \tag{B30}$$

where W_p is a weight matrix reflecting the frequency locations where the desired disturbance attenuation is to occur, W_Δ encapsulates the uncertainty contained within the mathematical models and, W_c depicts the restrictions on regions of operation of the servomechanisms.

Using the structured singular value, μ approach [Doyle (1982)] a less conservative measure of robustness may be calculated. If the controller, K, is absorbed into the plant, P, and provided that Δ has a block-diagonal structure and is normalized, then partitioning

$$\begin{pmatrix} e \\ x \end{pmatrix} = Q \begin{pmatrix} v \\ z \end{pmatrix} = \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{bmatrix} \begin{pmatrix} v \\ z \end{pmatrix} \tag{B31}$$

Then for robustness, the μ is defined as Eqn.(B32), and must remain less than unity, i.e.,

$$\mu\left(Q_{11}(j\omega)\right) = \frac{1}{\min\left(\bar{\sigma}\left(\Delta(j\omega)\right), \det\left(I - Q_{11}(j\omega)\Delta(j\omega)\right) = 0\right)} \tag{B32}$$

White et *al.* (1994) successfully applied the loop shaping H_∞ technique to the ROV depth control problem. Unlike other H_∞ control methods this technique concerns shaping the open loop dynamics of the ROV as opposed to the closed loop transfer function. Loop shaping was performed via two weighting functions, W_1 and W_2 which modified the open loop systems inputs and outputs respectively, thus achieving the desired loop shape for the system, namely; high gain at low frequencies, low gain at high frequencies, and some bandwidth and crossover frequency which yields desired gain and phase margins. Subsequent loop shaping designs emphasize the effects of different choices of W_1 and W_2 on the required closed loop design specifications. The resulting ROV controller's performance was assessed to variations in forward speed of ± 50 percent (5 to 15-knots) and perturbations in the pitch and heave coefficients of ± 20 percent.

The main difficulty with this application lies with finding a combination of weighting matrices that yields a controller which demanded inputs within the saturation limits of hydroplane and thruster actuators.

B 2.7 Concluding Remarks

It has been shown that when attempting to design a controller for an UUV, the control engineer is faced with several difficult problems. Thus, whilst many of the present generation of control systems installed within UUVs perform satisfactorily within given specifications, their overall effectiveness is limited.

Traditionally control system designs have been variants of the analogue PID controller. The main shortcoming of such designs has been the requirement for manual adjustment of the controller's parameters to compensate for changes in the craft's environment, but

these settings are rarely optimal for the UUV. The adjustments necessary for current and payload variations are time consuming. Consequently, there has been a growth of interest in autopilots that can automatically adapt themselves.

Adaptive control techniques for autopilot designs have, via the use of suitable cost functions in the optimization algorithms, enabled more efficient use of vehicle actuators in the event of environmental changes and varying vehicle dynamics. Due to the use of a cost function, usually minimizing the rudder activity and heading error when performing yaw changing manoeuvres, adaptive strategies do not always reduce the fuel consumption of the vehicle in all sea states. This inherent inability to measure the sea states effect upon the vehicle suggests that more advanced techniques must be developed for UUV autopilots. H_∞ approaches have enabled the design of optimal controllers in the presence of significant uncertainties within the UUV model without the need for on-line identification of the vehicles dynamics, and these controllers have been shown to be robust in operation.

Chapter 2 provides the reader with an overview of artificially intelligent approaches to UUV control system design. In summary, fuzzy logic control systems are inherently robust to non-linear time varying plant but remain reliant upon a rule base. Indeed, the self-organizing fuzzy logic controller develops its own rule base but requires some initial performance criterion. Such approaches have proved to be very successful at controlling UUVs. The fusion of fuzzy logic and neural network control methodologies offers a means by which the inherently robust and non-linear nature of the fuzzy controller can be combined with the powerful learning abilities of the neural network. Although there are examples of such fusions as applied to ship autopilot designs, little attention has been given to the design of UUV autopilots using these techniques. Consequently, the use of neuro-fuzzy approaches to control the dynamic behaviour of UUVs could offer significant technological advances in the field of UUV autopilot design and thus provide an excellent research area.

Whilst Chapter 3 discusses the implementation of the dynamic UUV model used throughout the thesis, Chapter 4 details the use of a neural autopilot structure for the tuning of fuzzy autopilots for course-changing control of an AUV.

References

Astrom, K.J. and Wittenmark, B. (1989). Adaptive control. Addison-Wesley Publishing Company. Boston.

Boulton, D.B. (1985). The development of multi-variable control algorithms for a dynamically positioned ship. Proceedings of the IEE Control 85, Cambridge, U.K., pp151-156.

Clarke, D.W. and Gawthorp, P. J. (1975). Self-tuning controller. Proceedings of the IEE. Part D, Vol. 122, No. 9. pp929-934.

Cristi, R., Papoulias, F.A. and Healey, A.J. (1990). Adaptive sliding mode control of autonomous underwater vehicles in the dive plane. IEEE Journal of Oceanic Engineering. Vol. 15, No. 3.

Da Cunha, J.P.V.S., Costa, R.R. and Hsu, L. (1995). Design of a high performance variable structure position control of ROVs. *ISOPE 91, Proceedings of the First Offshore and Polar Engineering Conference*, Edinrurgh, U.K.

Daley, S. and Gill, K.F. (1986). A design study of a self-organising fuzzy logic controller. Proc. of the Institution of Mechanical Engineers. Part C, Vol. 200. pp59-69.

Doyle, J.C. (1982). Performance and robustness analysis for structured uncertainty. Proceedings IEEE Decision and Control.

Egeskov, P., Bjerrum, A., Pascoal, A., Silvestre, C., Aage, C. and Wagner Smitt, L. (1994). Design, construction and hydrodynamic testing of the AUV MARIUS", Proceedings of the 1994 AUV Conference. Cambridge, Massachusetts.

Farrell, J. and Clauberg, B. (1993). Issues in the implementation of an indirect adaptive control system. IEEE Journal of Oceanic Engineering. Vol. 18, No. 3.

Fossen, T.I. (1991). Non-linear modelling and control of underwater vehicles. Dr. Ing. Thesis, Norwegian Institute of Technology, Trondheim.

Fossen, T.I. and Satungen, S. (1991). Adaptive control of non-linear systems: a case study of underwater robotic systems. Journal of Robotic Systems. Vol. 8. pp 393-412.

Goheen, K.R. and Jeffreys, E.R. (1989). Hardware and software considerations for a self-tuning controller. Proceedings of the DOI. Brighton, U.K.

Goheen, K.R. and Jeffreys, E.R. (1990). Multivariable self-tuning autopilots for autonomous and remotely operated vehicles. IEEE Journal of Oceanic Engineering. Vol. 15, No. 3.

Golten, J. and Verwer, A. (1991). Control system design and simulation. McGraw Hill.

Harris, C.J. and Billings, S.A. (1981). Self-tuning and adaptive control. IEE Engineering Series 15. Peter Peregrinus Press Ltd. U.K. and New York.

Healey, A.J. and Lienard, D. (1993). Multivariable sliding mode control for diving and steering of unmanned underwater vehicles. IEEE Journal of Oceanic Engineering. Vol. 18, No. 3.

Hsu, L., Costa R.R., Lizarralde, F.C., Da Cunha, J.P.V.S., Scieszko, J.L., Romanov, A.V., Junior, D.W. and Sant'anna, A.C. (1994). Underwater vehicle dynamic positioning based on a passive arm measurement system. Proc. of 2nd International Advanced Robotics Programme workshop: Mobile Robots for Subsea Environments, Monterey, California.

Kaminar, I., Pascoal, A.M., Khargonekar, P.P., and Thomson, C. (1993). A velocity algorithm for the implementation of gain-scheduled nonlinear controllers. Proceedings of the Second European Control Conference. pp787-792.

Katebi, M.R. and Byrne, J.C. (1988). LQG adaptive ship autopilot. Transactions of the Institute of Measurement and Control. Vol. 10, No. 4. pp187-191.

Kuo, B. (1987). Automatic control systems. Prentice-Hall International Editions. Seventh Edition.

Macejowski, J.M. (1989). Multivariable feedback design. Addison-Wesley.

Mandic, N.J., Scharf, E.M. and Mamdani, E.H. (1985). Practical application of a heuristic fuzzy rule-based controller to the dynamic control of a robot arm. IEE Proceedings. Part D, Vol. 132, No. 4. pp190-203.

Motora, S. (1953). On the automatic steering and yawing of ships in rough seas. Journal Society of Naval Architects, Japan. Volume 94.

Ogata, K. (1990). Modern control engineering. Second Edition. Prentice-Hall International Limited.

Owens, D.H. and Chotai, A. (1983). Robust controller design for linear dynamic systems using approximate models. Proceedings of the IEE. Part D, Vol. 130, No. 2.

Shao, S. (1988). Fuzzy self-organising controller and its application for dynamic processes. Journal of Fuzzy Sets and Systems. No. 26, pp151-164.

Sharif, M.T., Roberts, G.N. and Sutton, R. (1996). Final experimental results of full scale fin/rudder roll stabilisation sea trials. Control Engineering Practice. Vol. 4, No. 3. pp377-384.

Shimmin, D.W. and Lucas, J. (1993). Control strategies for underwater vehicle autopilots. IEE Colloquium on Control and Guidance of Underwater Vehicles. Digest No:1993/232.

Slotine, J.J.E. (1983). Tracking control of non-linear systems using sliding surfaces. Ph.D Thesis. Department of Aeronautics and Astronautics, MIT Cambridge, MA.

Slotine, J.J.E. (1985). The robust control of robot manipulators. International Journal of Robotics Research. Vol. 4, No. 2. pp49-64.

Taylor, S.D.H. (1995). The design and tuning of fuzzy autopilots using artificial neural networks. MPhil Thesis, University of Plymouth/RNEC.

Trebi-Ollennu, A., King, J. and White, B.A. (1995). A study of robust multivariable control designs for remotely operated vehicle depth control systems. IEE Colloquium on Control and Guidance of ROVs. Digest No. 124.

Triantafyllou, M.S. and Grosenbaugh, M.A. (1991). Robust control for underwater vehicle systems with time delays. IEEE Journal of Oceanic Engineering. Vol. 16, No. 1. pp146-151.

Van Amerongen, J. (1982). Adaptive steering of ships: a model reference approach to improved manoeuvring and economical course-keeping. Ph.D Thesis. Delft University of Technology, Holland.

White, B.A., Stacey, B.A., Patel, Y. and Bingham, C. (1994). Robust control design of an ROV. Technical Report, Contract No. DRA/PD/102/94. The Royal Military College of Science, Cranfield University.

Yoerger, D.R. and Slotine, J.J. (1985). Robust trajectory control of underwater vehicles. IEEE Journal of Oceanic Engineering. Vol. 10, No. 4.

Yuh, J., Fox, J.S. and Lakshmi, R. (1990). Control and optimal sensing in underwater robotic vehicles (URVs). Proc. IEEE Oceans 90. Washington D.C., USA. pp88-93.

Appendix C: The Premise Tuned Autopilot Information

The contents of the autopilots produced via the premise tuning regimes are reproduced herein for completeness:

1. Name	9Tsnorm	bp_prem	chem_prem	sa_prem
2. Type	sugeno	sugeno	sugeno	sugeno
3. No. Inputs/Outputs	[2 1]	[2 1]	[2 1]	[2 1]
4. No. Input MFs	[3 3]	[3 3]	[3 3]	[3 3]
5. No. Output MFs	9	9	9	9
6. No. Rules	9	9	9	9
7. And Method	prod	prod	prod	prod
8. Or Method	probor	probor	probor	probor
9. Implication Method	min	min	min	min
10. Aggregation Method	max	max	max	max
11. DefuzzMethod	wtaver	wtaver	wtaver	wtaver
12. Input Labels	err	err	err	err
13.	der	der	der	der
14. Output Labels	rud	rud	rud	rud
15. Input Range	[-1 1]	[-1 1]	[-1 1]	[-1 1]
16.	[-1 1]	[-1 1]	[-1 1]	[-1 1]
17. Output Range	[-1 1]	[-1 1]	[-1 1]	[-1 1]
18. Input Membership	neg	neg	neg	neg
19. Function Labels	zero	zero	zero	zero
20.	pos	pos	pos	pos
21.	neg	neg	neg	neg
22.	zero	zero	zero	zero
23.	pos	pos	pos	pos
24. Output Membership	posbig	posbig	posbig	posbig
25. Function Labels	posmed	posmed	posmed	posmed
26.	possmall	possmall	possmall	possmall
27.	zeropos	zeropos	zeropos	zeropos
28.	zero	zero	zero	zero
29.	zeroneg	zeroneg	zeroneg	zeroneg
30.	negsmall	negsmall	negsmall	negsmall
31.	negmed	negmed	negmed	negmed
32.	negbig	negbig	negbig	negbig
33. Input Membership	gbellmf	gbellmf	gbellmf	gbellmf
34. Function Types	gbellmf	gbellmf	gbellmf	gbellmf
35.	gbellmf	gbellmf	gbellmf	gbellmf
36.	gbellmf	gbellmf	gbellmf	gbellmf
37.	gbellmf	gbellmf	gbellmf	gbellmf
38.	gbellmf	gbellmf	gbellmf	gbellmf

39. Output Membership	linear	linear	linear	linear
40. Function Types	linear	linear	linear	linear
41.	linear	linear	linear	linear
42.	linear	linear	linear	linear
43.	linear	linear	linear	linear
44.	linear	linear	linear	linear
45.	linear	linear	linear	linear
46.	linear	linear	linear	linear
47.	linear	linear	linear	linear

Input Membership Function Parameters

48. [0.500 2.500 -1.000]	[0.560 2.618 -1.046]	[0.500 2.495 -1.000]	[0.502 2.406 -0.995]
49. [0.500 2.500 0.000]	[0.464 2.623 0.034]	[0.499 2.511 -0.001]	[0.502 2.666 0.002]
50. [0.500 2.500 1.000]	[0.510 2.272 1.045]	[0.499 2.490 0.999]	[0.498 2.372 0.999]
51. [0.500 2.500 -1.000]	[0.353 2.326 -0.748]	[0.501 2.501 -0.999]	[0.499 2.496 -1.003]
52. [0.500 2.500 0.000]	[0.649 2.683 -0.001]	[0.499 2.497 0.000]	[0.504 2.505 0.008]
53. [0.500 2.500 1.000]	[0.309 2.408 0.808]	[0.499 2.499 1.000]	[0.494 2.504 1.002]

Output Membership Function Parameters

54. [0 0 1]	[0 0 1]	[0 0 1]	[0 0 1]
55. [0 0 0.75]	[0 0 0.75]	[0 0 0.75]	[0 0 0.75]
56. [0 0 0.5]	[0 0 0.5]	[0 0 0.5]	[0 0 0.5]
57. [0 0 0.25]	[0 0 0.25]	[0 0 0.25]	[0 0 0.25]
58. [0 0 0]	[0 0 0]	[0 0 0]	[0 0 0]
59. [0 0 -0.25]	[0 0 -0.25]	[0 0 -0.25]	[0 0 -0.25]
60. [0 0 -0.5]	[0 0 -0.5]	[0 0 -0.5]	[0 0 -0.5]
61. [0 0 -0.75]	[0 0 -0.75]	[0 0 -0.75]	[0 0 -0.75]
62. [0 0 -1]	[0 0 -1]	[0 0 -1]	[0 0 -1]

63. Rule List	[1 1 1 1 1]	[1 1 1 1 1]	[1 1 1 1 1]	[1 1 1 1 1]
64.	[1 2 2 1 1]	[1 2 2 1 1]	[1 2 2 1 1]	[1 2 2 1 1]
65.	[1 3 3 1 1]	[1 3 3 1 1]	[1 3 3 1 1]	[1 3 3 1 1]
66.	[2 1 4 1 1]	[2 1 4 1 1]	[2 1 4 1 1]	[2 1 4 1 1]
67.	[2 2 5 1 1]	[2 2 5 1 1]	[2 2 5 1 1]	[2 2 5 1 1]
68.	[2 3 6 1 1]	[2 3 6 1 1]	[2 3 6 1 1]	[2 3 6 1 1]
69.	[3 1 7 1 1]	[3 1 7 1 1]	[3 1 7 1 1]	[3 1 7 1 1]
70.	[3 2 8 1 1]	[3 2 8 1 1]	[3 2 8 1 1]	[3 2 8 1 1]
71.	[3 3 9 1 1]	[3 3 9 1 1]	[3 3 9 1 1]	[3 3 9 1 1]

Appendix D: The Premise and Consequent Tuned Autopilot Information

The details of the autopilots produced via the full parameter tuning regimes are reproduced herein for completeness:

1. Name	9Tsnorm	hyb_all	chem_all	sa_all
2. Type	sugeno	sugeno	sugeno	sugeno
3. No. Inputs/Outputs	[2 1]	[2 1]	[2 1]	[2 1]
4. No. Input MFs	[3 3]	[3 3]	[3 3]	[3 3]
5. No. Output MFs	9	9	9	9
6. No. Rules	9	9	9	9
7. And Method	prod	prod	prod	prod
8. Or Method	probor	probor	probor	probor
9. Implication Method	min	min	min	min
10. Aggregation Method	max	max	max	max
11. DefuzzMethod	wtaver	wtaver	wtaver	wtaver
12. Input Labels	err	err	err	err
13.	der	der	der	der
14. Output Labels	rud	rud	rud	rud
15. Input Range	[-1 1]	[-1 1]	[-1 1]	[-1 1]
16.	[-1 1]	[-1 1]	[-1 1]	[-1 1]
17. Output Range	[-1 1]	[-1 1]	[-1 1]	[-1 1]
18. Input Membership	neg	neg	neg	neg
19. Function Labels	zero	zero	zero	zero
20.	pos	pos	pos	pos
21.	neg	neg	neg	neg
22.	zero	zero	zero	zero
23.	pos	pos	pos	pos
24. Output Membership	posbig	posbig	posbig	posbig
25. Function Labels	posmed	posmed	posmed	posmed
26.	possmall	possmall	possmall	possmall
27.	zeropos	zeropos	zeropos	zeropos
28.	zero	zero	zero	zero
29.	zeroneg	zeroneg	zeroneg	zeroneg
30.	negsmall	negsmall	negsmall	negsmall
31.	negmed	negmed	negmed	negmed
32.	negbig	negbig	negbig	negbig
33. Input Membership	gbellmf	gbellmf	gbellmf	gbellmf
34. Function Types	gbellmf	gbellmf	gbellmf	gbellmf
35.	gbellmf	gbellmf	gbellmf	gbellmf
36.	gbellmf	gbellmf	gbellmf	gbellmf
37.	gbellmf	gbellmf	gbellmf	gbellmf
38.	gbellmf	gbellmf	gbellmf	gbellmf

39. Output Membership	linear	linear	linear	linear
40. Function Types	linear	linear	linear	linear
41.	linear	linear	linear	linear
42.	linear	linear	linear	linear
43.	linear	linear	linear	linear
44.	linear	linear	linear	linear
45.	linear	linear	linear	linear
46.	linear	linear	linear	linear
47.	linear	linear	linear	linear

Input Membership Function Parameters

48.	[0.500 2.500 -1.000]	[0.918 2.959 -1.082]	[0.504 3.553 -1.080]	[0.412 6.371 -1.211]
49.	[0.500 2.500 0.000]	[0.450 2.791 0.003]	[0.561 1.756 -0.007]	[0.683 2.763 -0.158]
50.	[0.500 2.500 1.000]	[0.059 1.877 1.013]	[0.464 2.868 1.074]	[0.424 1.709 1.168]
51.	[0.500 2.500 -1.000]	[0.579 2.710 -1.233]	[0.493 2.506 -1.002]	[0.489 2.591 -1.061]
52.	[0.500 2.500 0.000]	[0.505 2.685 -0.081]	[0.519 2.406 0.012]	[0.485 2.237 0.004]
53.	[0.500 2.500 1.000]	[0.611 2.243 1.161]	[0.459 2.534 0.981]	[0.401 2.690 0.992]

Output Membership Function Parameters

54.	[0 0 1]	[-0.486 -0.879 -0.029]	[0.129 -0.058 0.143]	[0.122 -0.215 0.268]
55.	[0 0 0.75]	[-0.489 -0.902 0.001]	[-0.101 -0.029 0.148]	[-0.042 -0.011 0.314]
56.	[0 0 0.5]	[-0.415 -0.816 0.003]	[-0.025 -0.061 -0.044]	[0.404 -0.326 0.102]
57.	[0 0 0.25]	[-0.299 -0.703 -0.123]	[-0.028 -0.109 0.146]	[0.115 -0.215 0.351]
58.	[0 0 0]	[-0.149 -0.891 0.004]	[-0.202 0.085 -0.005]	[-0.472 0.492 -0.004]
59.	[0 0 -0.25]	[-0.305 -0.306 -0.037]	[0.149 0.145 -0.067]	[0.434 0.291 -0.254]
60.	[0 0 -0.5]	[-0.590 -0.839 -0.117]	[-0.124 0.067 -0.004]	[-0.262 0.281 -0.190]
61.	[0 0 -0.75]	[-0.481 -1.081 -0.061]	[-0.084 0.170 0.160]	[-0.053 0.110 0.100]
62.	[0 0 -1]	[-0.659 -1.311 0.781]	[0.009 0.035 0.059]	[-0.083 0.163 0.059]

63. Rule List	[1 1 1 1 1]	[1 1 1 1 1]	[1 1 1 1 1]	[1 1 1 1 1]
64.	[1 2 2 1 1]	[1 2 2 1 1]	[1 2 2 1 1]	[1 2 2 1 1]
65.	[1 3 3 1 1]	[1 3 3 1 1]	[1 3 3 1 1]	[1 3 3 1 1]
66.	[2 1 4 1 1]	[2 1 4 1 1]	[2 1 4 1 1]	[2 1 4 1 1]
67.	[2 2 5 1 1]	[2 2 5 1 1]	[2 2 5 1 1]	[2 2 5 1 1]
68.	[2 3 6 1 1]	[2 3 6 1 1]	[2 3 6 1 1]	[2 3 6 1 1]
69.	[3 1 7 1 1]	[3 1 7 1 1]	[3 1 7 1 1]	[3 1 7 1 1]
70.	[3 2 8 1 1]	[3 2 8 1 1]	[3 2 8 1 1]	[3 2 8 1 1]
71.	[3 3 9 1 1]	[3 3 9 1 1]	[3 3 9 1 1]	[3 3 9 1 1]

Appendix E: Non-Interacting Control System Design

This work herein describes the design method used to produce the decoupling elements of Eqn.(5.7) and Eqn.(5.8).

Figure E1 illustrates the block diagram representation of the non-interacting design approach whereby a 2 input - 2 output system is shown. Each controller is specifically designed in order that the interactions within the process are exactly cancelled. More specifically, the system is assumed to be at an equilibrium operating point such that small changes in the input variables will lead to small changes in the outputs (Doebelin (1985)):

$$\Delta c_1 \approx \left.\frac{\partial c_1}{\partial m_1}\right|_{m_2 \ const} \Delta m_1 + \left.\frac{\partial c_1}{\partial m_2}\right|_{m_1 \ const} \Delta m_2 \tag{E1}$$

$$\Delta c_2 \approx \left.\frac{\partial c_2}{\partial m_1}\right|_{m_2 \ const} \Delta m_1 + \left.\frac{\partial c_2}{\partial m_2}\right|_{m_1 \ const} \Delta m_2 \tag{E2}$$

Rewriting these equations in more general form yields:

$$\Delta c_1(s) = G_{11}(s)\Delta M_1(s) + G_{12}(s)\Delta M_2(s) \tag{E3}$$

$$\Delta c_2(s) = G_{21}(s)\Delta M_1(s) + G_{22}(s)\Delta M_2(s) \tag{E4}$$

assuming that $D_{11} = D_{22} = 1$ for simplicity. In order that the loops be successfully decoupled, the signal passing from M_1^* to C_2 via the path $D_{21}G_{22}$ should cancel that going by the path $D_{11}G_{21}$ such that

$$M_1^* D_{21} G_{22} = - M_1^* G_{21} \tag{E5}$$

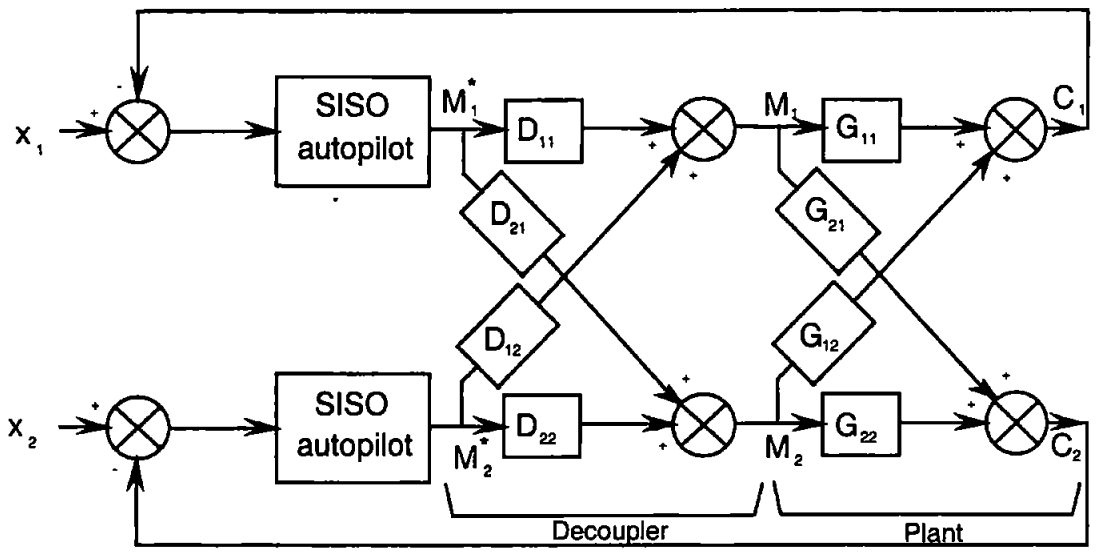


Figure E1: Non-Interacting Control System Design.

and the signal passing from M_2^* to C_1 via the path $D_{12}G_{11}$ should cancel that going by the path $D_{22}G_{12}$:

$$M_2^* D_{12}G_{11} = - M_2^* G_{12} \tag{E6}$$

Hence the decoupling elements for this system are defined as:

$$D_{21} = -\frac{G_{21}}{G_{22}} \text{ and } D_{12} = -\frac{G_{12}}{G_{11}} \tag{E7}$$

where M_1^* affects only C_1 and M_2^* affects only C_2. However the transfer functions are not simply G_{11} and G_{22} but are replaced by

$$M_1^* G_{11} + M_1^* D_{21}.G_{12} = C_1 \tag{E8}$$

and

$$M_2^* G_{22} + M_2^* D_{12} G_{21} = C_2 \tag{E9}$$

thus

$$\frac{C_1}{M_1^*} = G_{11} - \frac{G_{21}G_{12}}{G_{22}} \tag{E10}$$

$$\frac{C_2}{M_2^*} = G_{22} - \frac{G_{21}G_{12}}{G_{11}} \tag{E11}$$

Thus controllers can now be designed for each loop of the plant independently, using traditional SISO techniques. This approach can naturally be extended to accommodate a general class of multivariable systems. In real systems there is inevitably some mismatch between the mathematical model of the interaction terms and the actual interaction terms themselves. However, vast improvements can still be obtained by approaching a multivariable problem in this way.

References

Doebelin, E.O. (1985). Control System Principles and Design. John Wiley and Sons Inc., New York, USA.

Appendix F: The Hybrid Rule Tuned Roll Autopilot Information – 9 Rules

The details of the hybrid tuned MISO roll regulating autopilot are reproduced herein for completeness:

1. Name	roll_hy
2. Type	sugeno
3. Inputs/Outputs	[2 1]
4. NumInputMFs	[3 3]
5. NumOutputMFs	9
6. NumRules	9
7. AndMethod	prod
8. OrMethod	probor
9. ImpMethod	min
10. AggMethod	max
11. DefuzzMethod	wtaver
12. InLabels	rerr
13.	rder
14. OutLabels	rrud
15. InRange	[-1 1]
16.	[-1 1]
17. OutRange	[-1 1]
18. InMFLabels	neg
19.	zero
20.	pos
21.	neg
22.	zero
23.	pos
24. OutMFLabels	posbig
25.	posmed
26.	possmall
27.	zeropos
28.	zero
29.	zeroneg
30.	negsmall
31.	negmed
32.	negbig
33. InMFTypes	gbellmf
34.	gbellmf
35.	gbellmf
36.	gbellmf
37.	gbellmf
38.	gbellmf

39.	OutMFTypes	linear
40.		linear
41.		linear
42.		linear
43.		linear
44.		linear
45.		linear
46.		linear
47.		linear
48.	InMFParams	[0.9184 2.959 -1.082]
49.		[0.4503 2.791 0.003105]
50.		[0.3566 1.877 0.7513]
51.		[0.5796 2.71 -1.233]
52.		[0.5048 2.685 -0.08115]
53.		[0.6114 2.243 1.161]
54.	OutMFParams	[-0.4861 -0.8793 -0.0289]
55.		[-0.4887 -0.9022 0.0014]
56.		[-0.4864 -0.8960 0.0030]
57.		[-0.2992 0.7034 -0.1231]
58.		[-0.4876 -0.8912 0.0044]
59.		[-0.3045 -0.3063 -0.0370]
60.		[-0.5901 -0.8388 -0.1174]
61.		[-0.4805 -1.0810 -0.0611]
62.		[-0.6590 -1.311 0.7808]
63.	RuleList	[1 1 1 1 1]
64.		[1 2 2 1 1]
65.		[1 3 3 1 1]
66.		[2 1 4 1 1]
67.		[2 2 5 1 1]
68.		[2 3 6 1 1]
69.		[3 1 7 1 1]
70.		[3 2 8 1 1]
71.		[3 3 9 1 1]

Appendix G: The CANFIS Autopilot Information - 16 Rules

This work details the full matrix of parameters pertaining to the 16 fuzzy rule CANFIS hybrid rule tuned multivariable autopilot for yaw and roll control.

1. Name	CANFIS16
2. Type	sugeno
3. Inputs/Outputs	[4 2]
4. NumInputMFs	[2 2 2 2]
5. NumOutputMFs	[16 16]
6. NumRules	32
7. AndMethod	prod
8. OrMethod	probor
9. ImpMethod	min
10. AggMethod	max
11. DefuzzMethod	wtaver
12. InLabels	yerr
13.	yder
14.	serr
15.	sder
16. OutLabels	canards
17.	sternrd
18. InRange	[-1 1]
19.	[-1 1]
20.	[-1 1]
21.	[-1 1]
22. OutRange	[-1 1]
23.	[-1 1]
24. InMFLabels	neg
25.	pos
26.	neg
27.	pos
28.	neg
29.	pos
30.	neg
31.	pos
32. OutMFLabels	mf1
33.	mf2
34.	mf3
35.	mf4
36.	mf5
37.	mf6

38.		mf7
39.		mf8
40.		mf9
41.		mf10
42.		mf11
43.		mf12
44.		mf13
45.		mf14
46.		mf15
47.		mf16
48.		mf1
49.		mf2
50.		mf3
51.		mf4
52.		mf5
53.		mf6
54.		mf7
55.		mf8
56.		mf9
57.		mf10
58.		mf11
59.		mf12
60.		mf13
61.		mf14
62.		mf15
63.		mf16
64.	InMFTypes	gbellmf
65.		gbellmf
66.		gbellmf
67.		gbellmf
68.		gbellmf
69.		gbellmf
70.		gbellmf
71.		gbellmf
72.	OutMFTypes	linear
73.		linear
74.		linear
75.		linear
76.		linear
77.		linear
78.		linear
79.		linear
80.		linear
81.		linear
82.		linear

83.	linear
84.	linear
85.	linear
86.	linear
87.	linear
88.	linear
89.	linear
90.	linear
91.	linear
92.	linear
93.	linear
94.	linear
95.	linear
96.	linear
97.	linear
98.	linear
99.	linear
100.	linear
101.	linear
102.	linear
103.	linear
104. InMFParams	[0.9437 2.572 -0.944 0]
105.	[1.042 2.428 0.9818 0]
106.	[0.9715 2.531 -1.061 0]
107.	[0.9747 2.494 1.018 0]
108.	[0.8696 2.541 -0.954 0]
109.	[1.02 2.564 1.005 0]
110.	[0.9203 2.537 -0.925 0]
111.	[0.8584 2.509 1.083 0]
112. OutMFParams	[-0.0938 0.01186 0.09168 0.01037 -0.9447]
113.	[-0.09849 -0.03734 -0.07325 0.006042 0.02454]
114.	[-0.1259 0.07393 -0.0766 0.004301 -0.04256]
115.	[0.09554 0.02822 -0.01118 0.03402 0.0007768]
116.	[-0.01518 -0.02504 -0.02283 -0.01215 0.02896]
117.	[-0.01228 -0.0181 -0.07017 0.01761 -0.1178]
118.	[-0.06512 -0.0007333 0.0549 -0.1503 0.09814]
119.	[-0.07991 -0.06053 0.06759 -0.04874 -0.005731]
120.	[-0.2677 0.09207 -0.04177 0.003466 -0.03107]
121.	[-0.05288 -0.08954 0.1008 -0.07809 -0.1019]
122.	[-0.08666 -0.07202 -0.06384 -0.01019 -0.1286]
123.	[0.04118 0.02067 -0.03468 -0.1345 0.03141]
124.	[0.01379 -0.08343 -0.01257 -0.06115 -0.05321]
125.	[-0.08494 -0.06348 0.06887 -0.04667 0.04643]
126.	[0.07067 -0.07485 -0.005343 0.03873 -0.03032]
127.	[0.07396 0.02911 0.07811 0.02216 1.075]

128. [0.02972 0.06331 -0.02828 0.03148 -1.121]
129. [-0.03309 0.06209 0.04053 0.02942 -0.01404]
130. [0.03773 -0.048 -0.04138 -0.07171 -0.08232]
131. [-0.00851 0.05143 0.1392 0.0769 0.06729]
132. [-0.0887 -0.0311 0.08146 -0.007866 0.005566]
133. [-0.1456 -0.009546 0.0301 0.05896 0.09615]
134. [0.01561 -0.02747 0.05218 0.01402 0.01732]
135. [-0.026 -0.02265 0.01581 0.04728 0.0689]
136. [0.008604 -0.00645 -0.1481 0.1034 0.02088]
137. [-0.111 0.03482 0.02266 -0.03373 -0.0156]
138. [-0.04401 0.02489 0.03059 -0.008537 0.03702]
139. [0.07623 0.02189 -0.1335 0.01453 0.07764]
140. [-0.04596 -0.03258 -0.09966 -0.07056 0.06046]
141. [-0.1276 -0.06471 -0.00153 0.0388 -0.02721]
142. [0.04345 0.06957 0.07364 0.07236 -0.07393]
143. [-0.05549 -0.04549 0.02437 -0.06023 0.9446]
144. RuleList [1 1 1 1 16 0 1 1]
145. [1 1 1 2 15 0 1 1]
146. [1 1 2 1 14 0 1 1]
147. [1 1 2 2 13 0 1 1]
148. [1 2 1 1 12 0 1 1]
149. [1 2 1 2 11 0 1 1]
150. [1 2 2 1 10 0 1 1]
151. [1 2 2 2 9 0 1 1]
152. [2 1 1 1 8 0 1 1]
153. [2 1 1 2 7 0 1 1]
154. [2 1 2 1 6 0 1 1]
155. [2 1 2 2 5 0 1 1]
156. [2 2 1 1 4 0 1 1]
157. [2 2 1 2 3 0 1 1]
158. [2 2 2 1 2 0 1 1]
159. [2 2 2 2 1 0 1 1]
160. [1 1 1 1 0 16 1 1]
161. [1 2 1 1 0 15 1 1]
162. [2 1 1 1 0 14 1 1]
163. [2 2 1 1 0 13 1 1]
164. [1 1 1 2 0 12 1 1]
165. [1 2 1 2 0 11 1 1]
166. [2 1 1 2 0 10 1 1]
167. [2 2 1 2 0 9 1 1]
168. [1 1 2 1 0 8 1 1]
169. [1 2 2 1 0 7 1 1]
170. [2 1 2 1 0 6 1 1]
171. [2 2 2 1 0 5 1 1]
172. [1 1 2 2 0 4 1 1]

173. [1 2 2 2 0 3 1 1]
174. [2 1 2 2 0 2 1 1]
175. [2 2 2 2 0 1 1 1]

Appendix H: The On-Line Tuned Gaussian Autopilot Information – 16 Rules

Details of the on-line tuned autopilot produced via the hybrid rule tuning regime (with a forgetting factor of 0.975 and a step size of 5%) are reproduced herein for completeness:

1.	Name	OL_975_5
2.	Type	Sugeno
3.	Inputs/Outputs	[4 2]
4.	NumInputMFs	[2 2 2 2]
5.	NumOutputMFs	[16 16]
6.	NumRules	32
7.	AndMethod	prod
8.	OrMethod	probor
9.	ImpMethod	min
10.	AggMethod	max
11.	DefuzzMethod	wtaver
12.	InLabels	yerr
13.		yder
14.		rerr
15.		rder
16.	OutLabels	can
17.		ste
18.	InRange	[-1 1]
19.		[-1 1]
20.		[-1 1]
21.		[-1 1]
22.	OutRange	[-1 1]
23.		[-1 1]
24.	InMFLabels	neg
25.		pos
26.		neg
27.		pos
28.		neg
29.		pos
30.		neg
31.		pos
32.	OutMFLabels	mf1
33.		mf2
34.		mf3
35.		mf4
36.		mf5
37.		mf6

38. mf7
39. mf8
40. mf9
41. mf10
42. mf11
43. mf12
44. mf13
45. mf14
46. mf15
47. mf16
48. mf1
49. mf2
50. mf3
51. mf4
52. mf5
53. mf6
54. mf7
55. mf8
56. mf9
57. mf10
58. mf11
59. mf12
60. mf13
61. mf14
62. mf15
63. mf16
64. InMFTypes gbellmf
65. gbellmf
66. gbellmf
67. gbellmf
68. gbellmf
69. gbellmf
70. gbellmf
71. gbellmf
72. OutMFTypes linear
73. linear
74. linear
75. linear
76. linear
77. linear
78. linear
79. linear
80. linear
81. linear
82. linear

83.	linear
84.	linear
85.	linear
86.	linear
87.	linear
88.	linear
89.	linear
90.	linear
91.	linear
92.	linear
93.	linear
94.	linear
95.	linear
96.	linear
97.	linear
98.	linear
99.	linear
100.	linear
101.	linear
102.	linear
103.	linear
104 InMFParams	[0.2654 3.127 -1.0 0]
105.	[3.121 2.609 0.5289 0]
106.	[1.396 2.501 -0.381 0]
107.	[1.755 2.398 0.444 0]
108.	[1.498 2.663 -0.548 0]
109.	[1.987 2.462 0.3001 0]
110.	[1.450 2.529 -0.3801 0]
111.	[1.400 2.551 0.4111 0]
112. OutMFParams	[0.877 -0.05152 -0.1083 -0.01407 0.02972]
113.	[0.1268 -0.009987 -0.01959 -0.001706 0.004428]
114.	[0.07797 -0.006236 -0.01219 -0.001044 0.002715]
115.	[0.01438 -0.00134 -0.002538 -0.0001498 0.0005347]
116.	[0.2909 -0.0003214 -0.009953 -0.006734 0.009106]
117.	[0.01692 -0.0001089 -0.0007172 -0.0003743 0.0005443]
118.	[0.009086 -0.0000475 -0.000369 -0.0002045 0.000288]
119.	[0.0008702 -0.0000718 -0.0001374 -2.8e-6 0.00004443]
120.	[0.5015 -0.1883 -1.561 0.3624 0.05529]
121.	[0.2602 -0.3148 -1.767 0.2863 -0.1314]
122.	[-0.2411 -0.1457 -1.541 0.4015 0.04173]
123.	[1.917 -0.6347 -2.345 0.4627 0.1039]
124.	[-0.6036 0.08536 0.1311 -0.04445 -0.59]
125.	[-0.264 -0.07701 -0.125 -0.01428 -0.571]
126.	[-0.9672 0.07787 0.1239 -0.0334 -0.6004]
127.	[0.03124 -0.1411 -0.2406 -0.002756 -0.543]

128. [0.877 -0.05152 -0.1083 -0.01407 0.02972]
129. [0.2909 -0.0003214 -0.009953 -0.006734 0.009106]
130. [0.5015 -0.1883 -1.561 0.3624 0.05529]
131. [-0.6036 0.08536 0.1311 -0.04445 -0.59]
132. [0.1268 -0.009987 -0.01959 -0.001706 0.004428]
133. [0.01692 -0.0001089 -0.0007172 -0.0003743 0.0005443]
134. [0.2602 -0.3148 -1.767 0.2863 -0.1314]
135. [-0.264 -0.07701 -0.125 -0.01428 -0.571]
136. [0.07797 -0.006236 -0.01219 -0.001044 0.002715]
137. [0.009086 -0.00004748 -0.0003686 -0.000205 0.000288]
138. [-0.2411 -0.1457 -1.541 0.4015 0.04173]
139. [-0.9672 0.07787 0.1239 -0.0334 -0.6004]
140. [0.01438 -0.00134 -0.002538 -0.0001498 0.0005347]
141. [0.0008702 -0.0000718 -0.0001374 -2.8e-6 0.00004443]
142. [1.917 -0.6347 -2.345 0.4627 0.1039]
143. [0.03124 -0.1411 -0.2406 -0.002756 -0.543]
144. RuleList [1 1 1 1 16 0 1 1]
145. [1 1 1 2 15 0 1 1]
146. [1 1 2 1 14 0 1 1]
147. [1 1 2 2 13 0 1 1]
148. [1 2 1 1 12 0 1 1]
149. [1 2 1 2 11 0 1 1]
150. [1 2 2 1 10 0 1 1]
151. [1 2 2 2 9 0 1 1]
152. [2 1 1 1 8 0 1 1]
153. [2 1 1 2 7 0 1 1]
154. [2 1 2 1 6 0 1 1]
155. [2 1 2 2 5 0 1 1]
156. [2 2 1 1 4 0 1 1]
157. [2 2 1 2 3 0 1 1]
158. [2 2 2 1 2 0 1 1]
159. [2 2 2 2 1 0 1 1]
160. [1 1 1 1 0 16 1 1]
161. [1 2 1 1 0 15 1 1]
162. [2 1 1 1 0 14 1 1]
163. [2 2 1 1 0 13 1 1]
164. [1 1 1 2 0 12 1 1]
165. [1 2 1 2 0 11 1 1]
166. [2 1 1 2 0 10 1 1]
167. [2 2 1 2 0 9 1 1]
168. [1 1 2 1 0 8 1 1]
169. [1 2 2 1 0 7 1 1]
170. [2 1 2 1 0 6 1 1]
171. [2 2 2 1 0 5 1 1]
172. [1 1 2 2 0 4 1 1]

173.	[1 2 2 2 0 3 1 1]
174.	[2 1 2 2 0 2 1 1]
175.	[2 2 2 2 0 1 1 1]

Appendix I: The On-Line Tuned Gaussian Autopilot Information – 16 Rules

Details of the on-line tuned autopilot produced via the hybrid rule tuning regime (with a forgetting factor of 0.99 and a step size of 5%) are reproduced herein for completeness:

1. Name	OL_99_5
2. Type	Sugeno
3. Inputs/Outputs	[4 2]
4. NumInputMFs	[2 2 2 2]
5. NumOutputMFs	[16 16]
6. NumRules	32
7. AndMethod	min
8. OrMethod	max
9. ImpMethod	min
10. AggMethod	max
11. DefuzzMethod	centroid
12. InLabels	yerr
13.	yder
14.	rerr
15.	rder
16. OutLabels	can
17.	ste
18. InRange	[-1 1]
19.	[-1 1]
20.	[-1 1]
21.	[-1 1]
22. OutRange	[-1 1]
23.	[-1 1]
24. InMFLabels	neg
25.	pos
26.	neg
27.	pos
28.	neg
29.	pos
30.	neg
31.	pos
32. OutMFLabels	mf1
33.	mf2
34.	mf3
35.	mf4
36.	mf5
37.	mf6
38.	mf7

39.	mf8
40.	mf9
41.	mf10
42.	mf11
43.	mf12
44.	mf13
45.	mf14
46.	mf15
47.	mf16
48.	mf1
49.	mf2
50.	mf3
51.	mf4
52.	mf5
53.	mf6
54.	mf7
55.	mf8
56.	mf9
57.	mf10
58.	mf11
59.	mf12
60.	mf13
61.	mf14
62.	mf15
63.	mf16
64. InMFTypes	gbellmf
65.	gbellmf
66.	gbellmf
67.	gbellmf
68.	gbellmf
69.	gbellmf
70.	gbellmf
71.	gbellmf
72. OutMFTypes	linear
73.	linear
74.	linear
75.	linear
76.	linear
77.	linear
78.	linear
79.	linear
80.	linear
81.	linear
82.	linear
83.	linear

84.	linear
85.	linear
86.	linear
87.	linear
88.	linear
89.	linear
90.	linear
91.	linear
92.	linear
93.	linear
94.	linear
95.	linear
96.	linear
97.	linear
98.	linear
99.	linear
100.	linear
101.	linear
102.	linear
103.	linear
104 InMFParams	[0.2291 3.375 -1.002 0]
105.	[3.036 2.672 0.6747 0]
106.	[1.376 2.536 -0.3799 0]
107.	[1.743 2.411 0.4235 0]
108.	[1.573 2.766 -0.5118 0]
109.	[1.877 2.357 0.2987 0]
110.	[1.489 2.546 -0.3811 0]
111.	[1.399 2.551 0.4129 0]
112. OutMFParams	[4.584e-6 2.716e-7 -1.414e-6 -3.896e-7 -0.00002737]
113.	[4.578e-6 2.713e-7 -1.413e-6 -3.891e-7 -0.00002734]
114.	[4.626e-6 2.741e-7 -1.428e-6 -3.932e-7 -0.00002763]
115.	[4.621e-6 2.738e-7 -1.426e-6 -3.927e-7 -0.00002759]
116.	[4.586e-6 2.717e-7 -1.415e-6 -3.897e-7 -0.00002738]
117.	[4.58e-6 2.714e-7 -1.413e-6 -3.892e-7 -0.00002735]
118.	[4.628e-6 2.743e-7 -1.428e-6 -3.934e-7 -0.00002764]
119.	[4.623e-6 2.739e-7 -1.427e-6 -3.929e-7 -0.0000276]
120.	[0.03682 0.002182 -0.01136 -0.003129 -0.2199]
121.	[0.03678 0.002179 -0.01135 -0.003126 -0.2196]
122.	[0.03717 0.002202 -0.01147 -0.003159 -0.2219]
123.	[0.03712 0.0022 -0.01146 -0.003155 -0.2217]
124.	[0.03684 0.002183 -0.01137 -0.003131 -0.22]
125.	[0.03679 0.00218 -0.01135 -0.003127 -0.2197]
126.	[0.03718 0.002203 -0.01147 -0.00316 -0.222]
127.	[0.03714 0.002201 -0.01146 -0.003156 -0.2218]
128.	[4.584e-6 2.716e-7 -1.414e-6 -3.896e-7 -0.00002737]

129. [4.586e-6 2.717e-7 -1.415e-6 -3.897e-7 -0.00002738]
130. [0.03682 0.002182 -0.01136 -0.003129 -0.2199]
131. [0.03684 0.002183 -0.01137 -0.003131 -0.22]
132. [4.578e-6 2.713e-7 -1.413e-6 -3.891e-7 -0.00002734]
133. [4.58e-6 2.714e-7 -1.413e-6 -3.892e-7 -0.00002735]
134. [0.03678 0.002179 -0.01135 -0.003126 -0.2196]
135. [0.03679 0.00218 -0.01135 -0.003127 -0.2197]
136. [4.626e-6 2.741e-7 -1.428e-6 -3.932e-7 -0.00002763]
137. [4.628e-6 2.743e-7 -1.428e-6 -3.934e-7 -0.00002764]
138. [0.03717 0.002202 -0.01147 -0.003159 -0.2219]
139. [0.03718 0.002203 -0.01147 -0.00316 -0.222]
140. [4.621e-6 2.738e-7 -1.426e-6 -3.927e-7 -0.00002759]
141. [4.623e-6 2.739e-7 -1.427e-6 -3.929e-7 -0.0000276]
142. [0.03712 0.0022 -0.01146 -0.003155 -0.2217]
143. [0.03714 0.002201 -0.01146 -0.003156 -0.2218]
144. RuleList [1 1 1 1 16 0 1 1]
145. [1 1 1 2 15 0 1 1]
146. [1 1 2 1 14 0 1 1]
147. [1 1 2 2 13 0 1 1]
148. [1 2 1 1 12 0 1 1]
149. [1 2 1 2 11 0 1 1]
150. [1 2 2 1 10 0 1 1]
151. [1 2 2 2 9 0 1 1]
152. [2 1 1 1 8 0 1 1]
153. [2 1 1 2 7 0 1 1]
154. [2 1 2 1 6 0 1 1]
155. [2 1 2 2 5 0 1 1]
156. [2 2 1 1 4 0 1 1]
157. [2 2 1 2 3 0 1 1]
158. [2 2 2 1 2 0 1 1]
159. [2 2 2 2 1 0 1 1]
160. [1 1 1 1 0 16 1 1]
161. [1 2 1 1 0 15 1 1]
162. [2 1 1 1 0 14 1 1]
163. [2 2 1 1 0 13 1 1]
164. [1 1 1 2 0 12 1 1]
165. [1 2 1 2 0 11 1 1]
166. [2 1 1 2 0 10 1 1]
167. [2 2 1 2 0 9 1 1]
168. [1 1 2 1 0 8 1 1]
169. [1 2 2 1 0 7 1 1]
170. [2 1 2 1 0 6 1 1]
171. [2 2 2 1 0 5 1 1]
172. [1 1 2 2 0 4 1 1]
173. [1 2 2 2 0 3 1 1]

174. [2 1 2 2 0 2 1 1]
175. [2 2 2 2 0 1 1 1]

Appendix J: The Hybrid Rule Tuned Gaussian Inference Autopilot Information – 9 Rules

The details of the Gaussian MISO autopilot produced via the hybrid rule parameter tuning regime are reproduced herein for completeness:

1. Name	G_SISO
2. Type	Gaussian
3. No. Inputs/Outputs	[2 1]
4. No. Input MFs	[3 3]
5. No. Output MFs	9
6. No. Rules	9
7. And Method	min
8. Or Method	max
9. Implication Method	min
10. Aggregation Method	max
11. DefuzzMethod	centroid
12. Input Labels	err
13.	der
14. Output Labels	can
15. Input Range	[-1 1]
16.	[-1 1]
17. Output Range	[-1 1]
18. Input Membership	neg
19. Function Labels	zero
20.	pos
21.	neg
22.	zero
23.	pos
24. Output Membership	posbig
25. Function Labels	posmed
26.	possmall
27.	zeropos
28.	zero
29.	zeroneg
30.	negsmall
31.	negmed
32.	negbig
33. Input Membership	gbellmf
34. Function Types	gbellmf
35.	gbellmf
36.	gbellmf
37.	gbellmf
38.	gbellmf

39. Output Membership	c2_gauss				
40. Function Types	c2_gauss				
41.	c2_gauss				
42.	c2_gauss				
43.	c2_gauss				
44.	c2_gauss				
45.	c2_gauss				
46.	c2_gauss				
47.	c2_gauss				
Input Membership Function Parameters					
48.	[0.485 2.113 -1.020]				
49.	[0.503 1.399 0.041]				
50.	[0.476 2.790 0.953]				
51.	[0.477 2.774 -1.076]				
52.	[0.557 2.388 -0.047]				
53.	[0.527 2.289 0.984]				
Output Membership Function Parameters					
54.	[0.4 1 1 1.499]				
55.	[0.4 0.75 0.75 0.368]				
56.	[0.4 0.5 0.5 2.091]				
57.	[0.4 0.25 0.25 0.390]				
58.	[0.4 0 0 0.897]				
59.	[0.4 -0.25 -0.25 1.350]				
60.	[0.4 -0.5 -0.5 1.012]				
61.	[0.4 -0.75 -0.75 0.576]				
62.	[0.4 -1 -1 -0.601]				
63. Rule List	[1 1 1 1 1]	[1 1 1 1 1]	[1 1 1 1 1]	[1 1 1 1 1]	
64.	[1 2 2 1 1]	[1 2 2 1 1]	[1 2 2 1 1]	[1 2 2 1 1]	
65.	[1 3 3 1 1]	[1 3 3 1 1]	[1 3 3 1 1]	[1 3 3 1 1]	
66.	[2 1 4 1 1]	[2 1 4 1 1]	[2 1 4 1 1]	[2 1 4 1 1]	
67.	[2 2 5 1 1]	[2 2 5 1 1]	[2 2 5 1 1]	[2 2 5 1 1]	
68.	[2 3 6 1 1]	[2 3 6 1 1]	[2 3 6 1 1]	[2 3 6 1 1]	
69.	[3 1 7 1 1]	[3 1 7 1 1]	[3 1 7 1 1]	[3 1 7 1 1]	
70.	[3 2 8 1 1]	[3 2 8 1 1]	[3 2 8 1 1]	[3 2 8 1 1]	
71.	[3 3 9 1 1]	[3 3 9 1 1]	[3 3 9 1 1]	[3 3 9 1 1]	

Appendix K: Modelling a Non-Linear Function: Gaussian Inference 'v' ANFIS

K1 Recovering information using ANFIS

Within the MATLAB Fuzzy Logic Toolbox exists an M-File which demonstrates adaptive non-linear noise cancellation using the adaptive network-based fuzzy inference system (ANFIS) command. This demonstration is summarized here to provide a comparative performance assessment between ANFIS modelling and the proposed Gaussian inference of Chapter 7.

Essentially, an information signal x (Eqn. (K1)) is sampled at 100Hz over 6 seconds. The information signal cannot be measured (recovered) without knowing the form of an interference signal β, which is generated from another random noise source γ via an unknown non-linear process. The interference signal γ that appears in the measured signal is assumed to be a function of an unknown non-linear equation which is actually given by Eqn. (K2).

$$x = \sin\left(\frac{40}{t+0.01}\right) \tag{K1}$$

$$\gamma(k) = \frac{4\sin(\beta(k))\beta(k-1)}{1+\beta(k-1)^2} \tag{K2}$$

This function is illustrated in Figure K1.

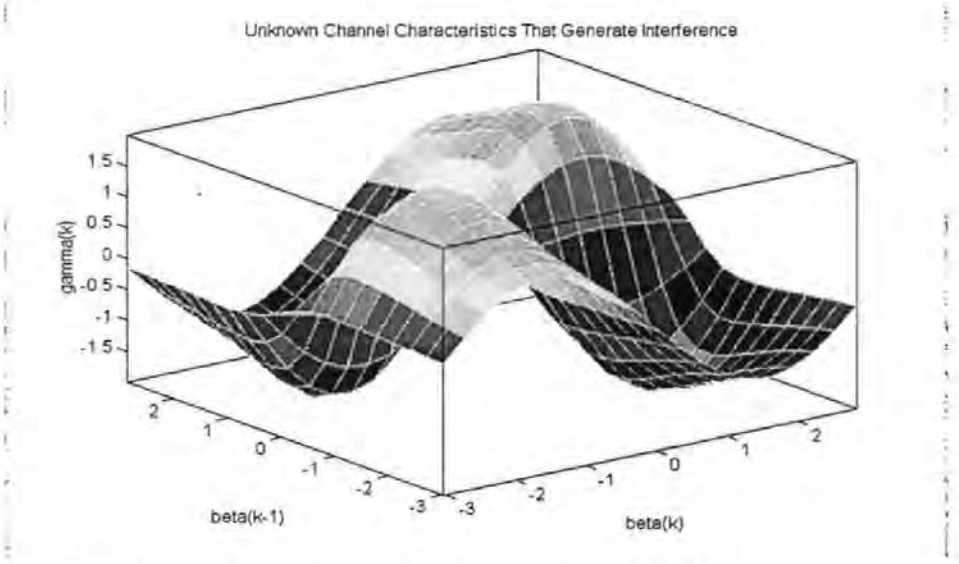


Figure K1: The non-linear channel characteristics (γ) defining the interference.

The noise source β and interference γ are shown together in Figure K2. Note that γ is related to β via the highly non-linear process of Figure K1; it is virtually impossible to ascertain if these two signals are correlated from the diagram.

The measured signal m is the sum of the original information signal x and the interference γ. However, the function γ is unknown. Consequently, it is required by ANFIS to recover the original information signal x given the interference. To train the ANFIS model the measured signal m is employed as a noisy version of the interference characteristics γ during training. It is assumed that the order of the non-linear channel is known to be 2; a 2-input ANFIS model is used for training. Two membership functions are subsequently assigned to each input, yielding 4 fuzzy rules. The step size is set equal to 0.2.

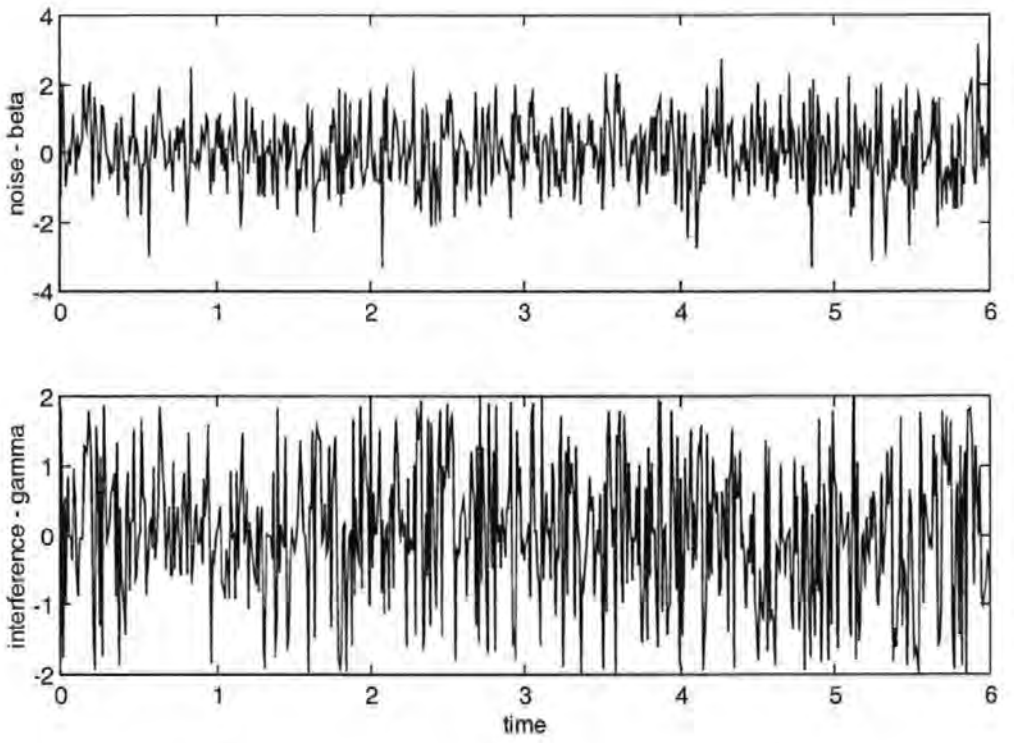


Figure K2: The noise source (β) and the interference (γ).

The estimated information signal x is equal to the difference between the measured signal m and the ANFIS output.

The original information signal x and the estimated x by ANFIS are plotted in Figure K3. With minimal training (10 epochs of the hybrid learning rule), the ANFIS model has produced an accurate representation of the under-lying interference signal γ, and has thus recovered x.

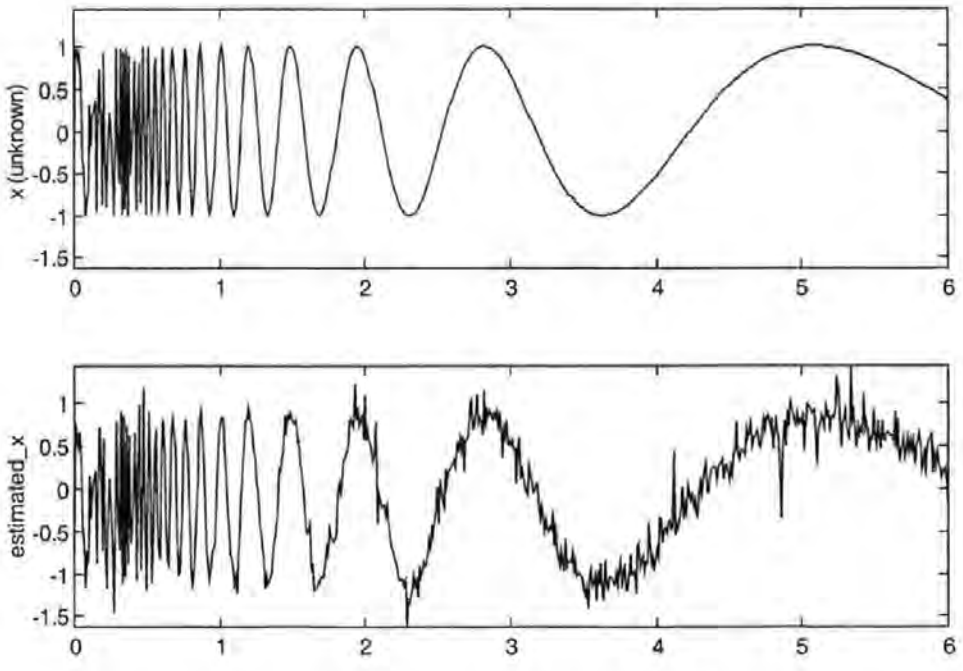


Figure K3: The original information signal x and the output of the ANFIS model (estimated x).

K2 Recovering information using Gaussian Inference

By applying the Gaussian inference of Chapter 7 to the same problem a comparison can be made between the two approaches. This comparison is arguably quite subjective, but will serve to illustrate the modelling capabilities of the proposed scheme.

Identical information (x), interference (β) and noise source (γ) signals were employed throughout this experiment to provide a direct comparison. Figure K4 illustrates the results when employing the Gaussian inference technique.

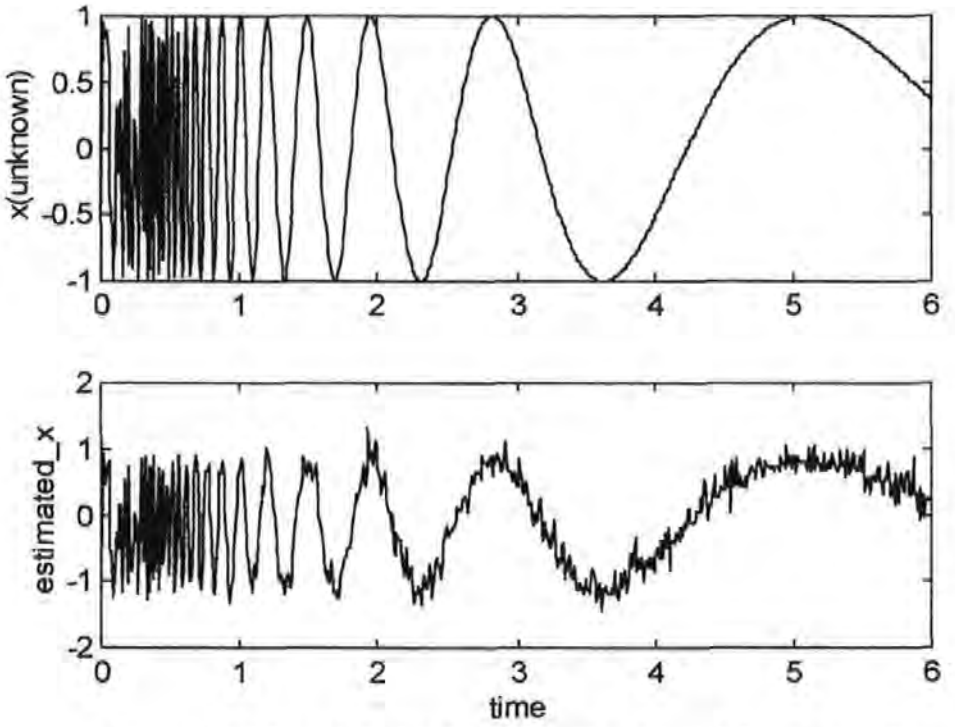


Figure K4: The original information signal x and the output of the Gaussian Inference model (estimated x).

Whilst the scale for the estimated information signal is [-2, 2] for the Gaussian Inference model compared to [-1, 1] for the ANFIS model, the difference between the information signal and the estimated information signals highlights the greater accuracy of the Gaussian Inference method. This is depicted in Figure K5. Table K1 documents summary statistics for each of the fuzzy models and further highlights the superior accuracy of this model when considering the difference between the required signal and the recovered signal.

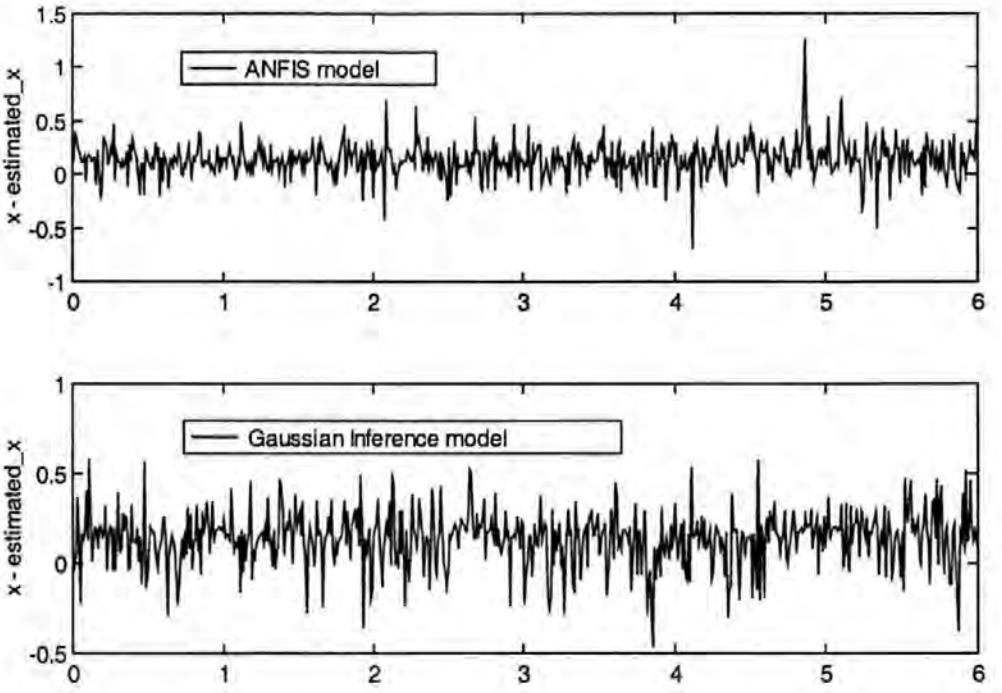


Figure K5: The information signal minus the estimated information signal – ANFIS and Gaussian inference fuzzy models.

Model	Max	Min	Variance	Standard Deviation
ANFIS	1.2680	-0.7021	0.0255	0.1597
Gaussian inference	0.5790	-0.4705	0.0237	0.1539

Table K1: Summary statistics for each modelling technique.

Appendix L: The Extended Hybrid Rule Tuned Gaussian Inference Autopilot Information – 16 Rules

The details of the Gaussian MIMO autopilot produced via the extended hybrid rule parameter tuning regime are reproduced herein for completeness:

1. Name	G_MIMOex
2. Type	Gaussian
3. Inputs/Outputs	[4 2]
4. NumInputMFs	[2 2 2 2]
5. NumOutputMFs	[16 16]
6. NumRules	32
7. AndMethod	min
8. OrMethod	max
9. ImpMethod	min
10. AggMethod	max
11. DefuzzMethod	centroid
12. InLabels	yerr
13.	yder
14.	rerr
15.	rder
16. OutLabels	can
17.	stern_hyd
18. InRange	[-1 1]
19.	[-1 1]
20.	[-1 1]
21.	[-1 1]
22. OutRange	[-1 1]
23.	[-1 1]
24. InMFLabels	neg
25.	pos
26.	neg
27.	pos
28.	neg
29.	pos
30.	neg
31.	pos
32. OutMFLabels	mf1
33.	mf2
34.	mf3
35.	mf4
36.	mf5
37.	mf6

38.		mf7
39.		mf8
40.		mf9
41.		mf10
42.		mf11
43.		mf12
44.		mf13
45.		mf14
46.		mf15
47.		mf16
48.		mf1
49.		mf2
50.		mf3
51.		mf4
52.		mf5
53.		mf6
54.		mf7
55.		mf8
56.		mf9
57.		mf10
58.		mf11
59.		mf12
60.		mf13
61.		mf14
62.		mf15
63.		mf16
64.	InMFTypes	gbellmf
65.		gbellmf
66.		gbellmf
67.		gbellmf
68.		gbellmf
69.		gbellmf
70.		gbellmf
71.		gbellmf
72.	OutMFTypes	gauss4mf
73.		gauss4mf
74.		gauss4mf
75.		gauss4mf
76.		gauss4mf
77.		gauss4mf
78.		gauss4mf
79.		gauss4mf
80.		gauss4mf
81.		gauss4mf
82.		gauss4mf

```
83.                     gauss4mf
84.                     gauss4mf
85.                     gauss4mf
86.                     gauss4mf
87.                     gauss4mf
88.                     gauss4mf
89.                     gauss4mf
90.                     gauss4mf
91.                     gauss4mf
92.                     gauss4mf
93.                     gauss4mf
94.                     gauss4mf
95.                     gauss4mf
96.                     gauss4mf
97.                     gauss4mf
98.                     gauss4mf
99.                     gauss4mf
100.                    gauss4mf
101.                    gauss4mf
102.                    gauss4mf
103.                    gauss4mf
104. InMFParams         [29.25 3.034 -31.62 0]
105.                    [31.74 4.686 30.71 0]
106.                    [3.044 2.714 -3.042 0]
107.                    [2.929 2.359 3.14 0]
108.                    [9.603 3.353 -11.07 0]
109.                    [10.33 2.896 10.7 0]
110.                    [1.049 2.452 -0.9862 0]
111.                    [1.04 2.432 0.9996 0]
112. OutMFParams        [0.3764 -0.5904 -0.7906 -1.407 -2.106 1.004]
113.                    [0.3965 0.4207 -0.4184 -0.5837 1.472 0.9937]
114.                    [0.3887 -0.3199 -0.4467 1.636 -0.2943 0.8256]
115.                    [0.3749 -2.039 -0.803 0.6605 0.4274 1.103]
116.                    [0.4049 -1.362 0.0219 -0.7394 -0.5888 1.011]
117.                    [0.3912 -0.3693 -1.352 -0.2986 0.08724 1.095]
118.                    [0.3961 0.0875 1.026 0.174 -0.7874 1.181]
119.                    [0.3876 0.4565 -0.7375 0.0341 0.1629 1.008]
120.                    [0.4081 0.1448 -0.407 -2.267 -1.807 1.141]
121.                    [0.3911 0.3836 -0.9694 -1.176 0.8542 1.247]
122.                    [0.4049 1.326 -0.3293 0.3779 -1.233 1.382]
123.                    [0.3937 0.5677 -0.9117 0.001 0.5197 0.8325]
124.                    [0.391 0.868 -1.356 0.563 -1.301 1.156]
125.                    [0.3865 0.596 0.4802 -1.17 0.5248 1.033]
126.                    [0.4109 -0.5992 -0.1073 -0.361 0.171 0.9392]
127.                    [0.3806 0.628 0.9703 1.305 1.386 1.353]
```

128. [0.4065 -1.917 0.2177 0.1113 -2.331 0.972]
129. [0.414 -0.0595 -0.4885 -0.3987 0.7263 0.999]
130. [0.4084 -0.9997 -0.6363 -0.633 -0.1174 1.225]
131. [0.4073 -0.1403 -0.7054 1.733 0.32 1.041]
132. [0.393 -0.1757 -0.3781 0.0961 -0.3417 1.163]
133. [0.3793 -0.175 -1.662 -1.321 0.4903 0.7439]
134. [0.4057 -0.6313 0.1111 0.3947 0.1652 1.163]
135. [0.3815 0.3245 0.0814 0.4327 1.617 1.006]
136. [0.4015 1.101 -1.06 -1.882 -0.0844 1.093]
137. [0.3955 0.3912 -0.4049 -1.292 -1.24 1.301]
138. [0.4074 -0.6408 -1.741 1.246 0.05701 0.6443]
139. [0.3984 -0.3339 -0.5742 -0.3913 1.126 1.141]
140. [0.3859 0.7413 0.9353 0.1486 -0.85 0.892]
141. [0.3945 0.3893 0.7527 -1.046 1.221 1.141]
142. [0.3936 0.00297 0.6548 0.0354 -0.4872 1.016]
143. [0.3898 2.058 -0.1135 0.4532 0.9786 1.088]
144. RuleList [1 1 1 1 16 0 1 1]
145. [1 1 1 2 15 0 1 1]
146. [1 1 2 1 14 0 1 1]
147. [1 1 2 2 13 0 1 1]
148. [1 2 1 1 12 0 1 1]
149. [1 2 1 2 11 0 1 1]
150. [1 2 2 1 10 0 1 1]
151. [1 2 2 2 9 0 1 1]
152. [2 1 1 1 8 0 1 1]
153. [2 1 1 2 7 0 1 1]
154. [2 1 2 1 6 0 1 1]
155. [2 1 2 2 5 0 1 1]
156. [2 2 1 1 4 0 1 1]
157. [2 2 1 2 3 0 1 1]
158. [2 2 2 1 2 0 1 1]
159. [2 2 2 2 1 0 1 1]
160. [1 1 1 1 0 16 1 1]
161. [1 2 1 1 0 15 1 1]
162. [2 1 1 1 0 14 1 1]
163. [2 2 1 1 0 13 1 1]
164. [1 1 1 2 0 12 1 1]
165. [1 2 1 2 0 11 1 1]
166. [2 1 1 2 0 10 1 1]
167. [2 2 1 2 0 9 1 1]
168. [1 1 2 1 0 8 1 1]
169. [1 2 2 1 0 7 1 1]
170. [2 1 2 1 0 6 1 1]
171. [2 2 2 1 0 5 1 1]
172. [1 1 2 2 0 4 1 1]

173.	[1 2 2 2 0 3 1 1]
174.	[2 1 2 2 0 2 1 1]
175.	[2 2 2 2 0 1 1 1]