Neural Comput & Applic (1998)7:131–140
© 1998 Springer-Verlag London Limited

Neural Computing & Applications

The ANFIS Approach Applied to AUV Autopilot Design

R. Sutton and P. J. Craven

Marine Technology Division, Institute of Marine Studies, University of Plymouth, Plymouth, UK

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), a chemotaxis tuning methodology and a fixed fuzzy rule-based approach. 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.

Keywords: Autonomous underwater vehicles; Autopilots; Neurofuzzy control

1. Introduction

The running costs of manned submersibles and support ship platforms for remotely operated vehicles are becoming increasingly high. As a consequence, to reduce financial overheads, considerable interest is now being shown in the development and construction of Autonomous Underwater Vehicles (AUVs) to undertake tasks such as ocean surveying for geological and biological reasons, pipeline inspection, explosive ordnance disposal and covert surveillance. For this type of vehicle to be truly autonomous, it is necessary for it to possess a reliable and robust onboard Navigation, Control and Guidance (NCG) system. A key element of the NCG system is the control subsystem, which is responsible for maintaining the vehicle on course. Several advanced control engineering concepts including $H\infty$ [1], sliding mode [2] and adaptive theories [3] are being employed in the design of autopilots, and have met with varying degrees of success.

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.

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 backpropagation algorithm is used to optimise 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 [9] and a fixed fuzzy rule-based autopilot.

2. Modelling the AUV Dynamics

Figure 1 shows the complete control authority of the AUV model. However, it should be noted that

Correspondence and offprint requests to: Dr R. Sutton, Marine Technology Division, Institute of Marine Studies, University of Plymouth, Drake Circus, Plymouth PL4 8AA, UK.

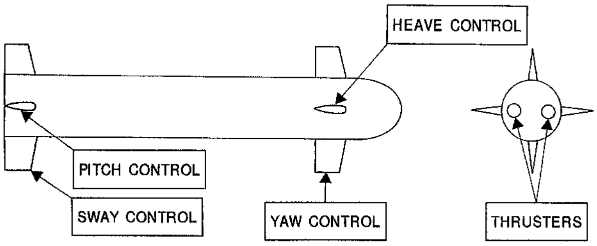

Fig. 1. Control authority of the AUV.

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

$$E = \begin{bmatrix} (m - Y_{\dot{v}}) & -Y_{\dot{R}} & 0 & -(Y_{\dot{P}} + mZ_G) & 0 \\ -N_{\dot{v}} & (I_z - N_{\dot{R}}) & 0 & -N_{\dot{P}} & 0 \\ 0 & 0 & 1 & 0 & 0 \\ -(K_{\dot{v}} + mZ_G) & -K_{\dot{R}} & 0 & (I_x - K_{\dot{P}}) & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$F = \begin{bmatrix} Y_{UV}U & (Y_{UR} - m)\,U & 0 & Y_{UP}U & 0 \\ N_{UV}U & N_{UR}U & 0 & N_{UP}U & 0 \\ 0 & 1 & 0 & 0 & 0 \\ K_{UV}U & (K_{UR} + mZ_G)U & 0 & K_{UP}U & -mgBG \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$G = \begin{bmatrix} Y_{UU\delta ru}U^2 & Y_{UU\delta rl}U^2 & 0 & 0 & 1 & 1 \\ N_{UU\delta ru}U^2 & N_{UU\delta r\ell}U^2 & \ell_\phi & -\ell_\phi & \ell_\psi & -\ell_\psi \\ 0 & 0 & 0 & 0 & 0 & 0 \\ K_{UU\delta ruU^2} & K_{UU\delta r\ell U^2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$u = \begin{bmatrix} \delta_{UP} \\ \delta_{LO} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

and the state variables are V, R, ψ, P, ϕ. For the interested reader, a nomenclature for the AUV parameters can be found at the end of the paper. To implement Eq. (1), use is made of an AUV MATLAB/Simulink simulation model supplied by the Defence Research Agency (DRA), Sea Systems Sector, Winfrith. The model, has 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 Eq. (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.

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 is shown in Fig. 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

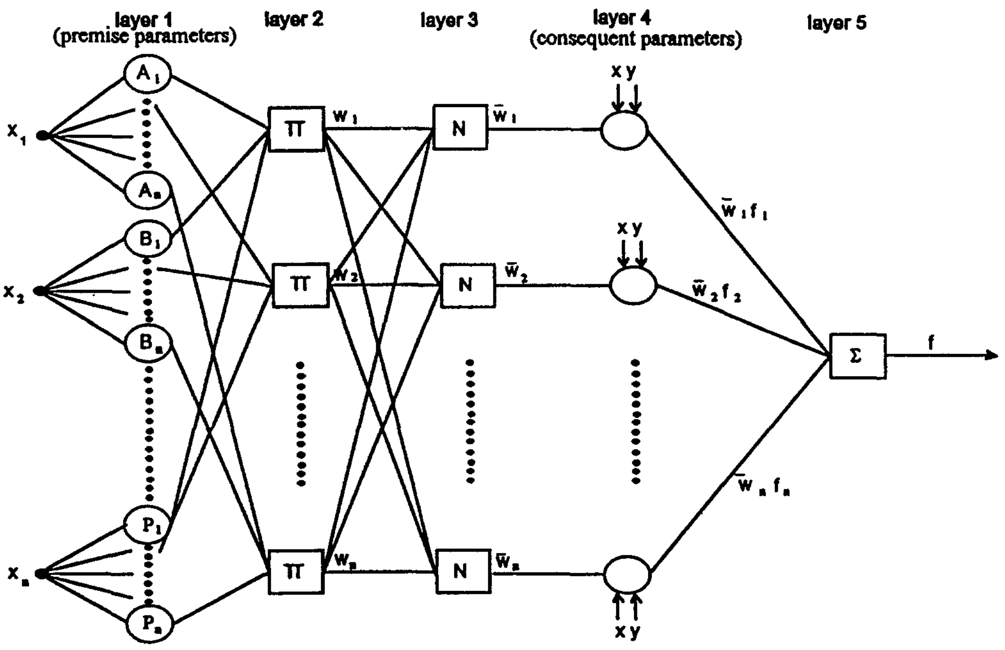

Fig. 2. The ANFIS architecture.

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, e.g.

$$O_{2,i} = w_i = \mu_{A_i}(x) \times \mu_{B_i}(y), \quad i = 1, 2 \qquad (4)$$

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 *i*th node calculates the ratio of the *i*th rules' firing strength to the sum of all rules' firing strengths:

$$O_{3,i} = \bar{w}_1 = \frac{w_i}{w_1 + w_2}, \quad i = 1, 2 \qquad (5)$$

For convenience, outputs of this layer are called *normalised firing strengths.*

Layer 4. Every *i*th 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 Σ, 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. Chemotaxis Tuned Autopilot Structure

The structure of the chemotaxis tuned autopilot is similar to that described in Section 3 and depicted in Fig. 2 for the ANFIS architecture. 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 chemotaxis algorithm is then applied to optimise the premises.

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:

$$E = \Sigma\ [\psi_d - \psi_a)^2 + \rho(\delta)^2]t \tag{8}$$

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 Eq. (3) as

$$\mu_{ij}(x) = \frac{1}{1 + \left[\left(\frac{x - c_{ij}}{a_{ij}}\right)^2\right]b_{ij}} \tag{9}$$

then Eq. (9) now represents the *j*th membership function on the *i*th 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 *i*th input universe of discourse. Therefore, the learning rule for a general parameter may be described as follows:

$$\begin{aligned}\Delta\alpha_{ij} &= -\ \eta \cdot \sum_{n=1}^{P} \frac{\partial E_n}{\partial O_{1n}} \cdot \frac{\partial O_{1n}}{\partial \alpha_{ij}} \\ &= -\eta \cdot \sum_{n=1}^{P} \frac{\partial E_n}{\partial O_{2n}} \cdot \frac{\partial O_{2n}}{\partial O_{1n}} \cdot \frac{\partial O_{1n}}{\partial \alpha_{ij}}\end{aligned} \tag{10}$$

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 $\frac{\partial O_1}{\partial \alpha_{ij}}$ is a straightforward calculation. This was the motivation for choosing the set functions described by Eq. (9).

The main difficulty is in the calculation of $\partial E_n/\partial O_{2n}$. Considering the AUV model as the final layer in the adaptive network, this calculation becomes simple for this layer:

$$\frac{\partial E_n}{\partial O_{5n}} = \frac{\partial}{O_{5n}}\left[\sum_{n=1}^{P} (T_n - O_{5n})^2\, t + \rho(O_{4n})^2\right] \tag{11}$$

$$= \sum_{n=1}^{P} -2(T_n - O_{5n})t \tag{12}$$

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 $\partial E_n/\partial O_{5n}$:

$$\frac{\partial E_n}{\partial O_{4n}} = \sum_{m=1}^{\#(5)} \frac{\partial E_n}{\partial O^m_{5n}} \cdot \frac{\partial O^m_{5n}}{\partial O_{4n}} \tag{13}$$

where #(5) is the number of nodes in Layer 5. Hence,

$$\frac{\partial E_n}{\partial O_{4n}} = \frac{\partial E_n}{\partial O_{5n}} \cdot \frac{\partial O_{5n}}{\partial O_{4n}} \tag{14}$$

as #(5) = 1. Now $\partial O_{5n}/\partial O_{4n}$ may be written as

$$\frac{\partial O_{5n}}{\partial O_{4n}} = \frac{d\psi}{d\delta_a} \tag{15}$$

whereby the function relating ψ to δ_a is non-linear and the derivative (or Jacobian) is approximated by

$$\frac{\partial O_{5n}}{\partial O_{4n}} = \frac{O_5(n) - O_5(n-1)}{O_4(n) - O_4(n-1)} \tag{16}$$

The only layer to be adapted using the backpropagation 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:

$$\begin{aligned}\Delta a_{ij} = -\ \eta \cdot \sum_{n=1}^{P} \frac{\partial E_n}{\partial O_{2n}} \cdot \frac{\partial O_{2n}}{\partial O_{1n}} \\ \cdot \left[\frac{2b_{ij}a_{ij}^{(2b_{ij}-1)}\ (x-c_{ij})a_{ij}^{2b_{ij}}\ (x-c_{ij})^{2b_{ij}}}{\{a_{ij}^{2b_{ij}}\ (x-c_{ij})^{2b_{ij}} + (x-c_{ij})^{2b_{ij}}a_{ij}^{2b_{ij}}\}^2}\right]\end{aligned} \tag{17}$$

$$\begin{aligned}\Delta \mathrm{b}_{ij} = -\ \eta \cdot \sum_{n=1}^{P} \frac{\partial \mathrm{E}_n}{\partial \mathrm{O}_{2n}} \cdot \frac{\partial \mathrm{O}_{2n}}{\partial \mathrm{O}_{1n}} \\ \cdot \left[\frac{2\mathrm{a}_{ij}^{2b_{ij}}\ (\mathrm{x}-\mathrm{c}_{ij})^{2b_{ij}}\mathrm{a}_{ij}^{2b_{ij}}\ (\mathrm{x}-\mathrm{c}_{ij})^{2b_{ij}}\ \ln\left[\left|\frac{\mathrm{a}_{ij}}{\mathrm{x}-\mathrm{c}_{ij}}\right|\right]}{\{\mathrm{a}_{ij}^{2b_{ij}}\ (\mathrm{x}-\mathrm{c}_{ij})^{2b_{ij}} + (\mathrm{x}\ -\mathrm{c}_{ij})^{2b_{ij}}\mathrm{a}_{ij}^{2b_{ij}}\}^2}\right]\end{aligned} \tag{18}$$

$$\Delta \mathrm{c}_{ij} = -\ \eta \cdot \sum_{n=1}^{P} \frac{\partial \mathrm{E}_n}{\partial \mathrm{O}_{2n}} \cdot \frac{\partial \mathrm{O}_{2n}}{\partial \mathrm{O}_{1n}}$$

$$\left[\frac{-2\mathrm{b}_{ij}\mathrm{a}_{ij}^{2b_{ij}}\,(\mathrm{c}_{ij}-\mathrm{x})^{(2b_{ij}-1)}\mathrm{a}_{ij}^{2b_{ij}}\,(-(\mathrm{x}-\mathrm{c}_{ij}))^{2b_{ij}}}{\{\mathrm{a}_{ij}^{2b_{ij}}\,(-(\mathrm{x}-\mathrm{c}_{ij}))^{2b_{ij}}+(\mathrm{c}_{ij}-\mathrm{x})^{2b_{ij}}\mathrm{a}_{ij}^{2b_{ij}}\}^2}\right] \tag{19}$$

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

$$output = F(\bar{I}, S) \tag{20}$$

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 $H \cdot F$ 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 Eq. (20) yields

$$H(output) = H \cdot F(\bar{I}, S) \tag{22}$$

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 Eq. (22), which yields the matrix equation

$$A\theta = B \tag{23}$$

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 $\|A\theta - B\|^2$, is the Least-Squares Estimator (LSE) θ^*

$$\theta^* = (A^TA)^{-1}\,A^TB \tag{24}$$

where A^T is the transpose of A and $(A^TA)^{-1}\,A^T$ is the pseudo-inverse of A if A^TA is non-singular. The recursive LSE formula can be employed by letting the ith row vector of matrix A defined in Eq. (23) be a_i^T and the ith element of B be b_i^T; then θ can be calculated iteratively as follows:

$$\theta_{i+1} = \theta_i + S_{i+1}a_{i+1}\,(b_{i+1}^T - a_{i+1}^T\theta_i), \tag{25}$$

$$S_{i+1} = S_i - \frac{S_i a_{i+1} a_{i+1}^T S_i}{1 + a_{i+1}^T S_i a_{i+1}}, \tag{26}$$

$$i = 0, 1, \ldots, P-1$$

where the least-squares estimator θ^* is equal to θ_p. The initial conditions needed to bootstrap Eqs (25) and (26) are $\theta_0 = 0$ 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, 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. 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 [9]. 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 characterised by an increase/decrease in a suitable cost function for the optimisation, and by converting this better/worse situation into a reinforcement signal, according to

$$r(t) = 1, \quad \text{better}$$

$$r(t) = 0, \quad \text{worse} \tag{27}$$

the chemotaxis search algorithm may be classed as a reinforcement learning technique. The algorithm is summarised in Table 1.

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.

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 AUV autopilot and its internal structure the rules may take the form:

> If yaw error (ψ_ϵ) is negative and yaw rate ($\dot{\psi}$) is positive, then canard demand is $f(\psi_\epsilon, \dot{\psi})$.

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 yaw error (ψ_ϵ) 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

$$E_i = (\psi_\epsilon,\ \mu_{Ei},\ (\psi_\epsilon)) \subset E$$

$$CE_i = (\psi_{\epsilon,\ \mu_{CEi}}\ (\dot{\psi})) \subset CE$$

Table 1. The 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.

and ψ_ϵ and $\dot{\psi}$ are elements of the appropriate universes of discourse with membership functions of μ_{Ei} (ψ_ϵ) and μ_{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 $Z_N = f(E_N, CE_N)$

To elicit the canard demand output (δ_a) then

$$w_i = E_i(\psi_\epsilon) \wedge CE_i(\dot{\psi}) \tag{28}$$

$$\delta_a = \frac{\sum_i w_i Z_i}{\sum_i w_i} \tag{29}$$

Results and Discussion

In the previous sections, the development of three nine-rule Sugeno type fuzzy autopilots has been discussed. First, the hybrid learning algorithm of Jang was applied to the task of tuning the premise and consequent parameters of a fuzzy autopilot based on a revised cost function (Eq. (8)), which was employed to account for control effort reduction. 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. Finally, a fixed rule base fuzzy autopilot was described. This design was included as a means of comparing and assessing the performance of the chosen neural network tuning methods.

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_ϵ is N and $\dot{\psi}$ is N then $\delta = -1.4619\,\psi_\epsilon - 0.8922\dot{\psi} + 0.6559$

if ψ_ϵ is N and $\dot{\psi}$ is Z then $\delta = -0.4916\psi_\epsilon - 0.8833\,\dot{\psi} - 0.0502$

if ψ_ϵ is N and $\dot{\psi}$ is P then $\delta = -0.5074\,\psi_\epsilon - 0.8987\dot{\psi} - 0.6972$

if ψ_ϵ is Z and $\dot{\psi}$ is N then $\delta = -0.4542\,\psi_\epsilon - 0.1090\dot{\psi} + 0.7879$

if ψ_ϵ is Z and $\dot{\psi}$ is Z then $\delta = 0.0000\psi_\epsilon + 0.0000\,\dot{\psi} + 0.0000$

if ψ_ϵ is Z and $\dot{\psi}$ is P then $\delta = -0.4542\psi_\epsilon - 0.1090\,\dot{\psi} - 0.7879$

if ψ_ϵ is P and $\dot{\psi}$ is N then $\delta = -0.5074\psi_\epsilon - 0.8987\,\dot{\psi} + 0.6972$

if ψ_ϵ is P and $\dot{\psi}$ is Z then $\delta = -0.4916\psi_\epsilon - 0.8833\,\dot{\psi} + 0.0502$

if ψ_ϵ is P and $\dot{\psi}$ is P then $\delta = -1.4619\psi_\epsilon - 0.8922\,\dot{\psi} - 0.6559$

Again, it should be noted that the only parameters for adaption within the chemotaxis tuned autopilot were the premise parameters, and thus the syntax for the final fuzzy autopilot was the same as the fixed rule based fuzzy autopilot:

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 $\delta = +12.5$

if ψ_ϵ is Z and $\dot{\psi}$ is N then $\delta = +6.25$

if ψ_ϵ is Z and $\dot{\psi}$ is Z then $\delta = 0.00$

if ψ_ϵ is Z and $\dot{\psi}$ is P then $\delta = -6.250$

if ψ_ϵ is P and $\dot{\psi}$ is N then $\delta = -12.5$

if ψ_ϵ is P and $\dot{\psi}$ is Z then $\delta = -18.75$

if ψ_ϵ 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.

Given sufficient training time, the resulting input fuzzy sets for all three autopilots were as shown in Fig. 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.

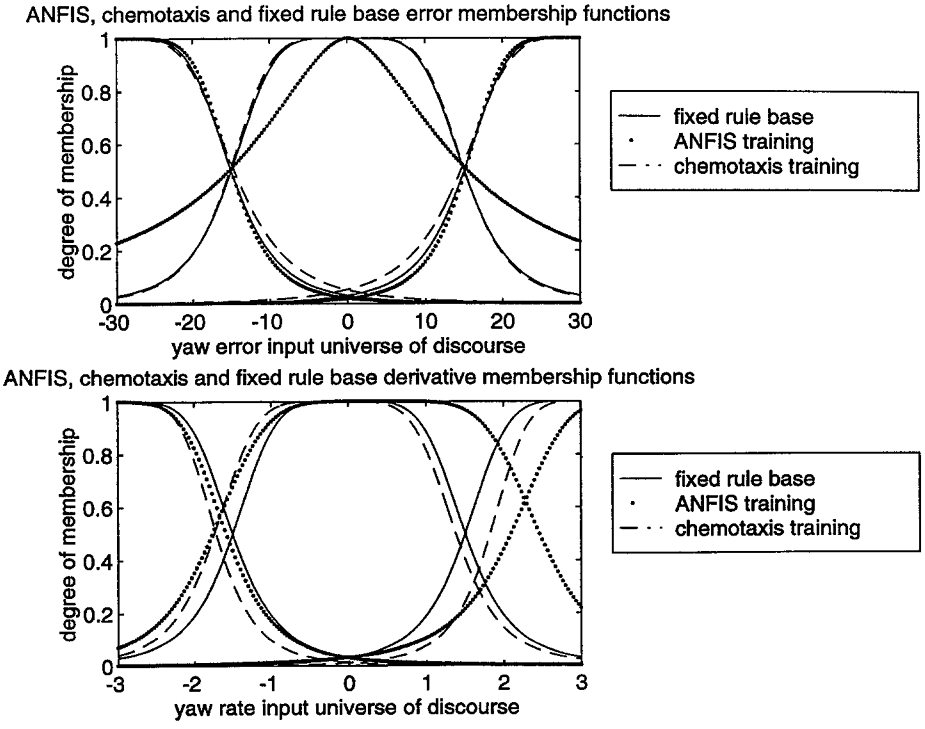

Fig. 3. Input fuzzy sets for the three autopilots.

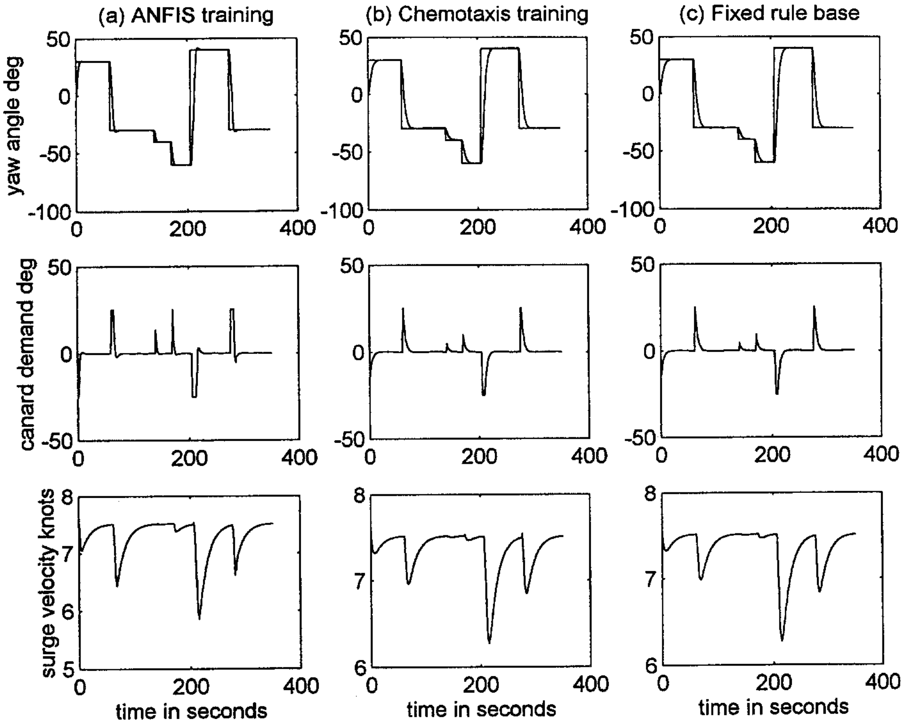

Fig. 4. Yaw responses, canard demands and velocity (surge) responses for the autopilots.

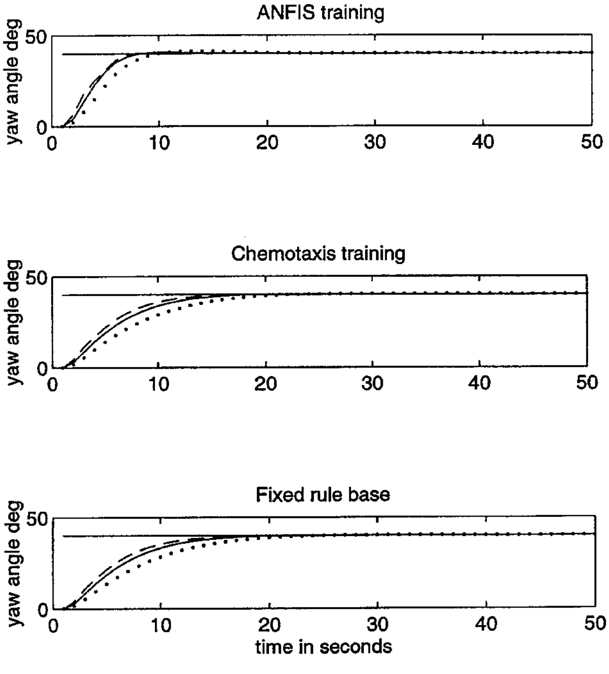

Fig. 5. Robustness testing of autopilots for yaw control over a speed range. • 5 knots; —— 7.5 knots; – • • 10 knots.

A qualitative assessment of the autopilot responses was provided by the AUV models responses to a series of random course changes, as shown in Fig. 4. It can be seen from these results that each course changing manoeuvre corresponds to a reduction in AUV surge velocity. In particular, the largest course change of 100° results in the greatest reduction in AUV surge velocity from the nominal value of 7.5 knots, thus highlighting the realism of the model used throughout these simulations.

Although Fig. 4 does not provide conclusive evidence of the ANFIS autopilot's superior performance over the chemotaxis and fixed rule base autopilots, it is apparent that the hybrid learning algorithm of the ANFIS technique leads to faster, more accurate responses. Additionally, the surge velocity profile of the ANFIS autopilot response is not significantly different to that of the fixed rule base fuzzy autopilot. 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 AUV speed envelope provided suitable insight into the robustness of each controller. Figure 5 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 dimin-

Table 2. Performance assessment of the autopilots.

AUV Model	ANFIS autopilot				Chemotaxis autopilot				Fixed rule autopilot			
	ψ_ϵ	δ_ϵ	T_R	$M_p(t)$	ψ_ϵ	δ_ϵ	T_R	$M_p(t)$	ψ_ϵ	δ_ϵ	T_R	$M_p(t)$
5 Knots	83.18	33.86	9.76	2.86	117.74	18.81	16.07	0.02	120.73	18.05	19.10	0.05
7.5 Knots	59.29	20.98	7.79	1.90	86.53	10.09	12.78	0	88.98	11.06	15.91	0.02
10 Knots	46.02	13.90	7.51	1.32	68.32	9.89	11.30	0	74.46	7.94	13.53	0.02

ished 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 fared 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, better than the chemotaxis tuned and fixed rule base autopilots.

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 (ISE) for the yaw error (ψ_ϵ) and the canard demand (δ_ϵ) are employed:

$$\psi_\epsilon = \int_{t_1}^{t_2} (\psi_d - \psi_a)^2 \, dt \tag{30}$$

$$\delta_\epsilon = \int_{t_1}^{t_2} (\delta_d - \delta_a)^2 \, dt \tag{31}$$

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% 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 2 contains figures pertaining to 5, 7.5 and 10 knots.

Summarised in Table 2 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 chemotaxis tuned and fixed rule base autopilots. However, the minimum rudder demand was exercised by the chemotaxis autopilot (coinciding with a reduction of course error by 2.75%), the activity being 8.77% less than the fixed rule fuzzy autopilot, highlighting the convergence of the chemotaxis algorithm towards a more optimal premise parameter set than those of the fixed rule base autopilot. 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 chemotaxis tuning.

At 5 knots, the autopilot developed using chemotaxis had approximately the same accuracy as the fixed rule base fuzzy autopilot. Conversely, 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 AUV behaviour.

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. Again, the ANFIS autopilot was deemed the best, with the most impressive rise times and quite acceptable overshoots.

8. Conclusions

The work described in this paper demonstrates that yaw 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, when compared to other neuro-fuzzy approaches, insofar as 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 AUV model, and for many helpful discussions concerning its implementation.

References

1. Cowling D. Full range autopilot design for an unmanned underwater vehicle. Proceedings of 1996 IFAC 13th Triennial World Congress, San Francisco, CA, 30 June–5 July 1996; Q: 339–344
2. Cristi R, Papoulias FA, Healey AJ. Adaptive sliding mode control of autonomous underwater vehicles in the dive plane. IEEE J Oceanic Eng 1990; 15(3): 152–160
3. Forsen TI, Fjellstad O-E. Robust adaptive control of underwater vehicles. Proceedings 3rd IFAC Workshop on Control Applications in Marine Systems, Trondheim, Norway, 10–12 May 1995; 66–74
4. Farbrother HNR, Stacey BA, Sutton R. A self-organising controller for an ROV. Proceedings IEE Control 91, Edinburgh, UK, 1991; 499–504
5. Smith SM, Rae GJS, Anderson DT, Shien AM. Fuzzy logic control of an autonomous underwater vehicle. Proceedings 1st IFAC International Workshop on Intelligent Autonomous Vehicles, Southampton, UK, 18–21 April 1993; 318–323
6. Fujii T, Ura T. Development of motion control system for AUV using neural nets. Proceedings of AUV 90, Washington, 1990; 81–86
7. Yuh J. A neural network controller for underwater robotic vehicles. IEEE J Oceanic Engineering 1990; 15(3): 161–166
8. Jang JSR. ANFIS: adaptive network-based fuzzy inference system. IEEE Trans Systems, Man Cybern 1993; 23(3): 665–685
9. Koshland DE. Bacterial chemotaxis in relation to neurobiology. Ann Rev Neurosci 1980; 3: 45–75
10. Marshfield WB. 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

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
$N_{UU\delta ru}$ 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