JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 33, No. 6, November–December 1995

On-Line Learning Neural-Network Controllers for Autopilot Systems

Marcello R. Napolitano* and Michael Kincheloe†
West Virginia University, Morgantown, West Virginia 26506-6106

This paper proposes the implementation of on-line learning neural controllers in the autopilot control laws of a modern high-performance aircraft. A first advantage of this design philosophy consists in avoiding the precomputation, storing, and interpolation between thousands of feedback gains of a typical flight control system. Another advantage is the ability to compensate for nonlinearities and model uncertainties. In addition, an on-line learning time-varying neural architecture will avoid the time-consuming gain recalculation following any modification to the aircraft or to its control system during its operative life. The implementation of these types of alternative controllers is made possible by the recent simultaneous advances in neural-network technology with regard to the availability of efficient and fast learning algorithms, along with progress in digital microprocessors. The approach is shown using a six-degree-of-freedom nonlinear simulation code. The traditional gain-scheduling-based control laws for typical autopilot functions are replaced by on-line learning neural architectures trained with the extended back-propagation algorithm. This algorithm has shown substantial improvements over the conventional back-propagation method in learning speed and accuracy. The results of the simulation with and without a man in the loop are presented and discussed.

Nomenclature

a_x = output of accelerometer aligned with the aircraft x body axis, g
a_z = output of accelerometer aligned with the aircraft z body axis, g
i = index of the neurons in the hidden layer(s)
J = performance index
j = index of the neuron(s) in the output layer
k = discrete time index
L = lower bound of the modified sigmoid function
O = output of a generic neuron
p = pattern for the neural-network input data
p = aircraft angular velocity around the x body axis, rad/s
q = aircraft angular velocity around the y body axis, rad/s
r = aircraft angular velocity around the z body axis, rad/s
T = slope of the modified sigmoid function
t = time, s
U = upper bound of the modified sigmoid function
U = forward speed, ft/s
x = argument of the sigmoid function in the back-propagation algorithm
α = angle of attack, rad
β = angle of sideslip, rad
δ = surface deflection, rad or deg
θ = pitch angle, rad
Φ = roll angle, rad
Ψ = yaw angle, rad

Subscripts

A = aileron
E = elevator
R = rudder
ref = desired conditions
T = throttle

Received Nov. 28, 1994; revision received May 9, 1995; accepted for publication May 12, 1995.

*Assistant Professor, Department of Mechanical and Aerospace Engineering. Member AIAA.
†Graduate Student, Department of Mechanical and Aerospace Engineering. Student Member AIAA.

I. Introduction

THE design of control laws for the flight control system for high-performance aircraft represents a challenge despite the availability of advanced tools in control theory and the increasing available onboard computational power. Modern high-performance aircraft are required to fly in a wide envelope with regard to speed, altitude, wing loading, and weight configuration. The problem is complicated by the artificial implementation of several different dynamic modes (normal, air-to-air gun, air-to-surface gun, air-to-surface bombing, etc.). Furthermore, modern high-performance military aircraft are required to operate at high angles of attack, that is, at nonlinear dynamic and aerodynamic conditions, where modeling still suffers from high levels of uncertainty.

Several classical and modern control-law methods have been proposed for designing longitudinal and lateral directional control laws for the aircraft. Typically these techniques, either time-domain or frequency-domain based, are applied to linearized, time-invariant aircraft mathematical models. However, these control-system design methods have a certain number of disadvantages. In fact their designs, performed off line at a limited number of linearized time-unvarying models representing different conditions in the flight envelope, require extensive gain-scheduling computations. Also, while the approach may be capable of handling mild nonlinearities, it is not suitable for highly nonlinear problems or those with inconsistencies between the actual aircraft dynamics and its mathematical model. Additional problems may arise for multiple-dynamic-mode control systems in order to ensure a smooth transition between different dynamic modes and/or different flight conditions in the flight envelope. Furthermore, there are unresolved issues related to available computational power, memory utilization, memory space, and the capability of utilizing the original gain values for substantial changes in the aircraft configuration.[1] A combination of all the problems listed above often lead to a long and costly process for the tuning of the gains in the flight computer software. Nevertheless, a majority of the flight control systems for today's military aircraft have been designed using different control techniques but with the same design guidelines based on gain scheduling.[2]

During the 1970s, adaptive control schemes[3,4] were introduced on a few experimental aircraft flight control systems for slowly time-varying dynamics between different regions in a flight envelope. However, these algorithms were still not capable of handling dynamic nonlinearities, and reservations existed about their applicability to actual operating aircraft, due to the required computational effort, and on their stability.

More recently, control design methods have been proposed with the purpose of addressing model uncertainties. In particular, H_∞-based control strategies[5–7] have shown robustness to modeling inaccuracy, but robustness to nonlinearities and time variation has not been guaranteed for the entire flight envelope, inducing the need for complicated gain-scheduling activities.

An approach that has shown excellent potential for the control of highly maneuverable aircraft is the dynamic inversion (DI) method. The application of this method to an aircraft operating at post-stall conditions and/or high angular velocities was first proposed in Ref. 8. While this approach involves only limited gain-scheduling activity, some disadvantages do exist. In fact, if the dynamic equations are not exactly known and/or if the dynamic states are not measured or estimated exactly, the cancellation of the nonlinear dynamics will not be exact. This, in turn, compromises robustness to modeling uncertainties and closed-loop performance. Recently, solutions to these disadvantages of the DI method have been proposed through the introduction of a robust outer-loop controller in addition to the DI inner loop.[9–12] This outer-loop controller is designed using structural singular-value (μ) techniques with the objective of achieving robustness and stability performance not guaranteed by the DI method alone.

A totally different approach is the use of neural network (NN) technology. References 13 and 14 provide clear descriptions of the basic principles of NN theory and a quite complete classification of different neural architectures and learning algorithms. One of the milestone works in exploring potential for NNs to be used as adaptive nonlinear controllers and estimators is described in Ref. 15. A key feature of these architectures is their ability to learn on line. This feature applies to simple or complex dynamic systems, time-unvarying or time-varying systems, noise-free or noise-corrupted systems, and linear or nonlinear systems.

In recent years the implementation of neural architectures in flight control laws, both as controllers and as estimators, has been proposed in several investigations. Reference 16 clearly defines all the issues related to the implementation of neural architectures to flight control problems. References 17 and 18 describe the results of the application of neural controllers in an autopilot model for a high-performance aircraft. The described applications involved both longitudinal and lateral directional control laws. However, all these investigations fell short of proposing on-line learning neural controllers as a replacement for gain-scheduling-based control schemes. Also, they all utilized neural architectures trained previously off line with linear mathematical models based on the classic back-propagation algorithm (BPA) and then simulated. The first author has proposed on-line learning neural architectures for very specific flight control problems, namely, sensor failure detection, identification, and accommodation actuator failure detection, identification, and accommodation.

The objective of this paper is to present results relative to the first phase of an overall study that has the goal of showing the suitability of on-line learning neural controllers for on-line control laws in a high-performance aircraft control system. This first phase of the investigation relates to on-line learning neural controllers in the control laws of the autopilot functions; the second phase relates to such controllers in the control laws of stability augmentation systems for both longitudinal and lateral directional dynamics.

Another purpose of this study is to provide an additional successful implementation of the extended BPA (EBPA),[20,22] a particularly effective algorithm in terms of accuracy and learning speed, for the on-line learning. It will be shown that these neural-based control laws for the different autopilot functions do not need any gain scheduling and are generally robust to system nonlinearities.

The paper is organized as follows. Section II reviews NN design methodology and introduces the key characteristics of the EBPA. Then Sec. III briefly discusses the aircraft mathematical model used in this study. Sections IV and V describe the design of different NN-based autopilot functions, with a discussion of the results of the application of neural controllers for the different autopilot functions. The effects of different parameters for the NN architectures and the effects of the on-line learning, at linear and nonlinear conditions, are also discussed. Section VI concludes the paper.

II. Neural Architectures and the Extended Back-Propagation Algorithm

A complete review of NN theoretical principles is available in Refs. 13 and 14. In general, a neural architecture is defined as a network system with input and output parameters interconnected through one or more layers of neurons, also called processing elements, which provide a nonlinear activation. A typical example of a three-layer NN (one input, one hidden, and one output layer) is shown in Fig. 1. A formal proof of the capabilities of neural architectures to operate at nonlinear conditions is given in Ref. 23, where it is shown that a NN with a single hidden layer with an arbitrary number of neurons is capable of mapping any nonlinear dynamics following training. This property can be extremely useful when the input–output data are related to a nonlinear time-varying system.

While this capability is undoubtedly very attractive, the implementation of NN controllers must be analyzed very carefully. The

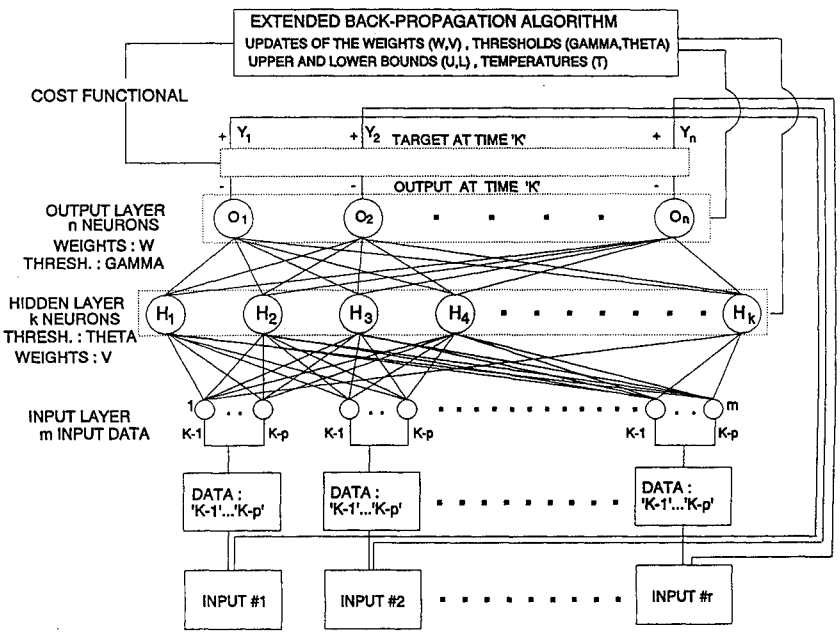

Fig. 1 Block diagram of a generic neural network.

first key issue in the design of an NN controller is the selection of on-line learning vs off-line training as the working mode. On-line learning implies that the NN has the capability of changing in real time the values of the numerical components that make up its architecture. Off-line training implies that the NN operating on line has been previously trained and has a frozen numerical architecture. Given that an aircraft is a time-varying system operating at linear and/or nonlinear conditions, on-line training is a very desirable feature and is the approach required to avoiding gain scheduling.

For on-line learning points of particular concern are 1) the amount of time needed to acquire a substantial learning level and 2) the complexity of the neural architectures.

Both these issues are related to the learning algorithm. Therefore the performance and the acceptability of an on-line learning NN controller are strictly related to the performances of its training algorithm. To date, the majority of the training for the NN is being performed with the BPA,[13,14] which is a gradient-based optimization method. Slow learning, especially for large-order systems, and local minima are classical disadvantages of the BPA.[13,14] An approach to solving these problems is given by the introduction of a heterogeneous network,[20,22] meaning that each neuron in the hidden and output layers has its own output capability of updating some new parameters, giving the overall architecture increased mapping and adaptation capabilities. Specifically, in a heterogeneous network each neuron is able to change its output range (upper and lower bounds: U, L) and the slope of the sigmoid activation function (temperature: T). The new nonlinear activation function will be given by

$$f(x_{i,j}, U_{i,j}, L_{i,j}, T_{i,j}) = \frac{U_{i,j} - L_{i,j}}{1 + e^{-x_{i,j}/T_{i,j}}} + L_{i,j} \qquad (1)$$

where i, j are the indices for the generic neurons of the hidden and output layers and x is the argument of the classic sigmoid function of the BPA. This function is implemented at each processing element of the hidden layer(s) and output layer. The gradient is calculated to perform the steepest descent as in the BPA. The only difference is that the gradient descent will be found with respect to each of the independent variables x, U, L, T, rather than x only. The training algorithm associated with this heterogeneous network, the EBPA, is described in detail in Refs. 20 and 22. Several studies comparing the EBPA and the BPA were performed by the first author and others. The results show that, for similar NN architectures trained with the EBPA and BPA with the same CPU time, the EBPA clearly outperformed the BPA.[21] A three-layer neural architecture trained with the EBPA is shown in Fig. 1.

III. Simulation Code and Autopilot Functions

The process described in the previous section has been numerically simulated using the software recently distributed for the AIAA Control Design Challenge.[24] This software provides a numerical nonlinear simulation of an aircraft model, with full-envelope nonlinear aerodynamics and full-envelope thrust for first-order engine response data. The aircraft modeled is a high-performance supersonic fighter. The aircraft primary flight control surfaces consist of horizontal stabilators that are capable of symmetric or differential movement, conventional ailerons, and a single vertical rudder. There are a total of five actuators (for the two ailerons, two stabilators, and one rudder), each with a rate of 24 deg/s. The maximum deflections for the elevators and the rudder are ±30 deg, and those for the ailerons are ±21.5 deg. The aerodynamics are modeled for the full aircraft envelope using multidimensional tables and linear interpolation to form nonlinear function generators. In the original configuration, the software does not allow the user to input a pilot maneuver; it only allows trimming of the aircraft at a specified flight condition starting from another specified initial condition. The original software also features four single-point autopilot functions (altitude hold, airspeed hold, roll-angle control, direction control) designed with classical control theory (PI, PID controllers).

The objective of this paper is to present the design of autopilot systems using NN theory. The main advantage of this approach is that on-line learning neural controllers will replace gain-scheduling activities, which can potentially be massive. Furthermore, numerical simulations will show that these controllers are robust to dynamic nonlinearities. Altitude hold, airspeed hold, pitch hold, and roll-angle and direction control systems were considered. Note that the original AIAA Control Design Challenge simulation code did not include a pitch hold system.

The evaluation of the performance of these neural controllers was performed in two different stages. In the first stage the on-line learning capabilities were investigated in a study where the aircraft was automatically flown between different flight conditions by the conventional autopilot. In the second stage the robustness of the autopilot design to nonlinearities was evaluated in a study where the aircraft was flown manually using keyboard commands. This stage also provided a primitive interface between the neural logic and the man in the loop.

IV. Analysis of the On-Line Learning Capabilities

In this stage the performance of different neural controllers at different points in the aircraft flight envelope were analyzed. The flight envelope for the AIAA Control Design Challenge aircraft is shown in Ref. 24. Typical design processes using classical compensation theory for these autopilot systems are outlined in Refs. 2 and 25. Typically these systems feature multiple feedback loops to achieve desirable transient characteristics and desirable steady-state error performance. Table 1 reviews typical architectures for these autopilot systems.[2,25]

It is known that the systems in Table 1 display multiple gains, requiring some gain-scheduling activities—which can be intensive—in order to have acceptable performance throughout the flight envelope.

The design of on-line learning neural controllers can be complicated by the large number of degrees of freedom (number of hidden layers, number of neurons per hidden layer, size of input data window, learning rate) in the selection of the neural architectures. Previous work by the first author and others[21,27] provided the following guidelines for the selection of suitable neural architectures:

1) On-line learning neural controllers are generally very robust to different numbers of neurons per hidden layer and a different input data window.

2) Medium-high learning rates, either constant or slowly decreasing with time, perform better for neural controllers.

3) Simple single-hidden-layer architectures perform as well as more complicated multiple-hidden-layer architectures.

Table 1 Architectures for considered autopilot systems

Altitude hold system	
System input: h_{ref}	
System output: elevator or stabilator deflection	
Type 1	feedback loops: h, a_x
Type 2	feedback loops: h, q, θ
Airspeed hold system	
System input: U_{ref}	
System output: throttle deflection	
	feedback loops: U, a_x
Pitch hold system	
System input: θ_{ref}	
System output: elevator or stabilator deflection	
Type 1	feedback loop: θ
Type 2	feedback loops: θ, q, $\dot{q}$
Roll-angle control system	
System input: Φ_{ref}	
System output: aileron deflection	
Type 1	feedback loop: Φ
Type 2	feedback loops: Φ, p
Type 3	feedback loops: Φ, p, r
Direction control system	
System input: Ψ_{ref}	
System output: rudder deflection	
Type 1	feedback loop: Ψ
Type 2	feedback loops: Ψ, r
Type 3	feedback loops: Ψ, r, Φ, p

Table 2 Architectures for the neural autopilot controllers

Altitude hold system
NN input: $\theta, q, h, \dot{h}, \delta_E$
NN input window: 3
NN output: δ_E
Number of hidden-layer neurons: 18
Learning rate: 0.5
Neural architecture: 15, 18, 1
Performance index: $0.25\ (h - h_{\text{ref}}) + 0.7(\dot{h} - \dot{h}_{\text{ref}}) + 0.05(q - q_{\text{ref}})$;
h_{ref} is user-defined; $\dot{h}_{\text{ref}} = q_{\text{ref}} = 0$

Airspeed hold system
NN input: a_x, U, δ_T
NN input window: 3
NN output: δ_T
Number of hidden-layer neurons: 12
Learning rate: 0.5
Neural architecture: 9, 12, 1
Performance index: $0.7\ (U - U_{\text{ref}}) + 0.3(a_x - a_{x\,\text{ref}})$;
U_{ref} is user-defined; $a_{x\,\text{ref}} = 0$

Pitch hold system
NN input: θ, q, δ_E
NN input window: 3
NN output: δ_E
Number of hidden-layer neurons: 12
Learning rate: 0.5
Neural architecture: 9, 12, 1
Performance index: $0.7\ (\theta - \theta_{\text{ref}}) + 0.3(q - q_{\text{ref}})$;
θ_{ref} is user-defined; $q_{\text{ref}} = 0$

Roll-angle control system
NN input: $\Phi, p, \delta_A, \delta_R$
NN input window: 3
NN output: δ_A
Number of hidden-layer neurons: 15
Learning rate: 0.5
Neural architecture: 12, 15, 1
Performance index: $0.5\ (\Phi - \Phi_{\text{ref}}) + 0.5(p - p_{\text{ref}})$;
Φ_{ref} is user-defined; $p_{\text{ref}} = 0$

Direction control system
NN input: $r, \dot{r}, \beta, \delta_R$
NN input window: 3
NN output: δ_R
Number of hidden-layer neurons: 15
Learning rate: 0.5
Neural architecture: 12, 15, 1
Performance index: $0.5\ (r - r_{\text{ref}}) + 0.5(\dot{r} - \dot{r}_{\text{ref}})$;
r_{ref} is user-defined; $\dot{r}_{\text{ref}} = 0$

4) A proper selection of the data to be provided as input and an appropriate choice of the performance index to be minimized are, by far, the most effective degrees of freedom.

A parametric study on the architectures of the five neural controllers for the five different autopilot functions in terms of both convergence speed and accuracy confirmed the previous results. For the input data selection, an important conclusion is that, for each of the five different autopilot systems, the very same variables whose inclusion in a feedback loop helps to improve the performance of classic autopilot system have the same positive effect on the neural autopilot systems. This allows a conceptual parallelism to be drawn between neural and classic autopilot systems, which needs to be better explored and understood. For the choice of the performance indices to be minimized on line for each of the five systems, the results show that the neural controllers perform very well in terms of convergence speed and steady-state error when minimizing performance indices given by the difference between the desired and the actual output target. In addition, the same neural architectures for each of the five systems perform even better when the performance indices to be minimized resemble the error formulation in a PI, PD, or PID controller. The resulting architectures for the neural controllers for each of five autopilot functions are reported in Table 2. The on-line learning capabilities of two specific autopilot systems, the altitude hold system and the airspeed hold system, were tested in a dynamic simulation where 18 different flight conditions in the flight envelope were considered.

It is clear that the on-line learning for the neural controllers implements a gradient-based algorithm, and therefore neural controllers are not designed to have desirable performance in terms of classical time-domain specifications such as damping, natural frequency, overshoot, settling time, etc. Nevertheless, it is intuitive that there is a direct relationship between those classical time-domain specifications and the minimization of the performance index. In order to evaluate the on-line learning capabilities, a criterion had to be defined. It was decided to select as a criterion a hybrid settling time defined as follows:

1) For the airspeed hold system, elapsed time to go from ±50 ft/s from the target airspeed to oscillation around target speed of ±3 ft/s.

2) For the altitude hold system, elapsed time to go from ±100 ft from the target altitude to oscillation around target altitude of ±6 ft.

These settling times were calculated for each of the controllers for the two autopilot systems at each of the 18 flight conditions. For simulation purposes the conventional autopilot was used to fly the aircraft between different flight conditions. It should be emphasized that linear dynamic and aerodynamic conditions were preserved during this simulation.

A point to be demonstrated is the superiority of the EBPA vs the BPA. The aircraft was flown twice: once with the on-line learning performed with the EBPA, and once with the BPA. The length of these simulations was different due to the different performance of each of the two controllers for each of the two autopilot systems. Table 3 shows the settling times at different flight conditions using the EBPA and the BPA for airspeed and altitude. From the results it can be seen that the EBPA altitude neural controller provides better performance than the BPA controller for 16 of 18 flight conditions. A similar improvement in performance is achievable with the EBPA for the airspeed hold system. For both autopilot systems the difference between the EBPA and BPA settling times varies from minor to substantial. Another consideration is that the superior performance of the EBPA does not seem to be correlated with the specific flight condition (viz., low vs high speed, low vs high altitude).

The reason for the improved performance of the EBPA over the BPA algorithm has to be sought in the different learning characteristics. Essentially the EBPA has more degrees of freedom in the minimization of the cost functions shown in Table 2 through the use in the gradient method of the following partial derivatives:

$$\frac{\partial f(x_{i,j}, U_{i,j}, L_{i,j}, T_{i,j})}{\partial x_{i,j}} = -\frac{(O_{i,j} - U_{i,j})(O_{i,j} - L_{i,j})}{T_{i,j}(U_{i,j} - L_{i,j})} \tag{2}$$

$$\frac{\partial f(x_{i,j}, U_{i,j}, L_{i,j}, T_{i,j})}{\partial U_{i,j}} = \frac{1}{1 + e^{-x_{i,j}/T_{i,j}}} \tag{3}$$

$$\frac{\partial f(x_{i,j}, U_{i,j}, L_{i,j}, T_{i,j})}{\partial L_{i,j}} = 1 - \frac{1}{1 + e^{-x_{i,j}/T_{i,j}}} \tag{4}$$

$$\frac{\partial f(x_{i,j}, U_{i,j}, L_{i,j}, T_{i,j})}{\partial T_{i,j}} = \frac{x_{i,j}(O_{i,j} - U_{i,j})(O_{i,j} - L_{i,j})}{T_{i,j}^2(U_{i,j} - L_{i,j})} \tag{5}$$

as opposed to the derivative

$$\frac{\mathrm{d}f(x_{i,j})}{\mathrm{d}x_{i,j}} = \frac{e^{-x_{i,j}}}{(1 + e^{-x_{i,j}})^2} \tag{6}$$

in the BPA algorithm. $O_{i,j}$ represent the generic output of a neuron at the hidden layer(s) (i) and at the output layer (j).

From the previous simulation it is apparent that neural autopilot systems have excellent inherent online local learning capabilities. Another interesting point was to evaluate whether these local capabilities increase when the neural architectures are previously trained at flight conditions different from the ones used in the previous simulation. For this purpose the neural altitude and airspeed hold systems were first trained at 200 other flight conditions within the aircraft flight envelope, with altitude and airspeed randomly selected. Following this training, the neural architectures were presented again with the original 18 flight conditions. The resulting settling times, with and without pretraining, are presented in Table 4. It may be deduced that the pretraining improves to a certain degree the local

Table 3 Airspeed and altitude systems: comparison of EBPA vs BPA performance

Initial conditions: altitude = 17,500 ft; airspeed = 500 ft/s						
FC	Alt (ft)	Airspeed (ft/s)	$T_{s\,\mathrm{ALT(EBP)}}$	$T_{s\,\mathrm{ALT(BP)}}$	$T_{s\,\mathrm{AIRSP(EBP)}}$	$T_{s\,\mathrm{AIRSP(BP)}}$
1	15,000	700	26.42	28.68	5.94	38.10
2	18,000	800	15.04	17.72	7.08	7.44
3	20,000	1,200	12.54	13.54	8.26	9.26
4	22,000	1,400	48.53	71.10	18.89	20.88
5	24,000	800	12.67	20.15	23.26	29.86
6	22,000	1,300	4.19	13.61	14.25	15.61
7	26,000	1,600	11.03	12.21	19.13	18.73
8	30,000	1,400	4.60	4.04	6.46	5.92
9	28,000	1,800	15.81	18.61	28.52	32.15
10	31,000	1,600	12.79	13.01	7.20	9.69
11	34,000	900	12.55	12.13	15.31	21.86
12	28,000	1,500	12.49	12.99	14.05	15.25
13	34,000	1,400	10.87	12.61	5.94	6.18
14	36,000	1,800	11.23	11.55	25.88	25.90
15	25,000	1,600	11.69	11.77	6.46	7.00
16	22,000	1,300	13.31	13.40	5.90	6.69
17	25,000	800	11.45	18.29	18.77	24.26
18	20,000	1,500	10.49	10.83	17.46	17.81

Table 4 Airspeed and altitude systems with and without pretraining (EBP algorithm)

Initial conditions: altitude = 17,500 ft; airspeed = 500 ft/s						
FC	Alt (ft)	Airspeed (ft/s)	$T_{s\,\mathrm{ALT}}$	$T_{s\,\mathrm{ALT(pre\text{-}tr.)}}$	$T_{s\,\mathrm{AIRSP}}$	$T_{s\,\mathrm{AIRSP(pre\text{-}tr.)}}$
1	15,000	700	26.42	25.22	6.16	5.94
2	18,000	800	15.04	14.84	7.08	7.07
3	20,000	1,200	12.54	11.50	8.28	8.26
4	22,000	1,400	48.53	33.12	19.63	18.89
5	24,000	800	12.67	20.87	23.26	23.26
6	22,000	1,300	4.19	12.33	14.29	14.25
7	26,000	1,600	11.03	10.12	19.49	19.13
8	30,000	1,400	4.60	13.13	6.48	6.46
9	28,000	1,800	15.81	14.85	28.50	28.52
10	31,000	1,600	12.79	11.77	7.23	7.21
11	34,000	900	12.55	11.75	15.41	15.31
12	28,000	1,500	12.49	11.17	14.03	14.05
13	34,000	1,400	10.87	4.59	5.89	5.94
14	36,000	1,800	11.23	10.55	26.24	25.88
15	25,000	1,600	11.69	11.13	6.48	6.46
16	22,000	1,300	13.31	11.71	5.90	5.90
17	25,000	800	11.45	11.97	18.79	18.78
18	20,000	1,500	10.49	16.59	17.46	17.46

learning capabilities of the neural controller for the altitude hold autopilot systems, but the performance of the neural airspeed hold system is practically unchanged. An investigation with a longer training was not deemed necessary, since for neural controllers, unlike neural estimators, the local learning capabilities are those of main interest.

V. Neural Autopilot Controllers at Nonlinear Conditions

The simulation previously described proved the existence of local on-line learning capabilities for neural autopilot controllers at linear conditions. As stated in the previous section, these simulations were linear, with the conventional autopilot in the original code flying the aircraft between different set points. The next objective was to test the neural control schemes at nonlinear conditions and, at the same time, to provide some sort of interface, even without graphic simulation capabilities, between the control scheme and the man in the loop. Therefore, particular subroutines were designed to provide control inputs for the aircraft simulation from the keyboard using a DOS interrupt command. The setup gives the user the option to fly the aircraft manually (with commands for thrust control and control-surface deflection from specific keystrokes) or automatically (with autopilot type to be selected between the conventional autopilot in the original code and EBPA neural autopilots). The aircraft was then flown several times, and the performance of the five different EBPA neural autopilots with the man in the loop were separately evaluated at linear and nonlinear conditions. While it may not be realistic to activate autopilot systems while the aircraft is at nonlinear dynamic and aerodynamic conditions, it is nevertheless important to explore whether the on-line learning capabilities of neural autopilot controllers are retained at nonlinear conditions. In fact, even though nonlinear conditions are not typical operating conditions for autopilot systems, good performance at those conditions could make neural controllers a very attractive approach for more complex stability augmentation systems for high-performance aircraft.

During the entire fly-by-keyboard phase for testing the neural autopilots, diary files were kept for each autopilot. The most important dynamic data at the instant when the autopilot functions were activated—the activation and deactivation times and the settling times, defined as the time to achieve the desired target starting from the activation time—were recorded.

The results of the simulation for the neural altitude autopilot are reported in Fig. 2. In the selected setup configuration for the autopilot functions, the neural autopilot, when activated, is instructed to hold the current altitude. Similarly, Fig. 3 shows the graphical results of the neural airspeed autopilot. In the selected setup, the neural autopilot, when activated, holds the current airspeed. Figure 4 shows the results relative to the pitch neural autopilot. Unlike the previous autopilot functions, the pitch neural autopilot, when activated, is instructed to hold a user-defined pitch angle. Arbitrary small values for the target pitch angle were selected during the simulation each time the neural autopilot was activated. The neural control scheme is tested at low and high angles of attack.

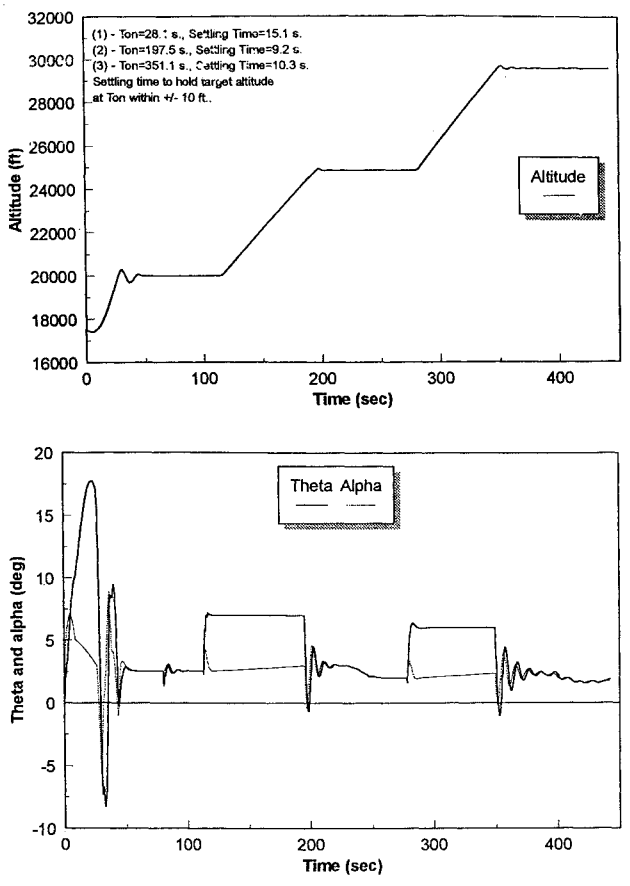

Fig. 2 Neural altitude autopilot—low angles of attack. Upper part: altitude vs time; lower part: α and θ vs time.

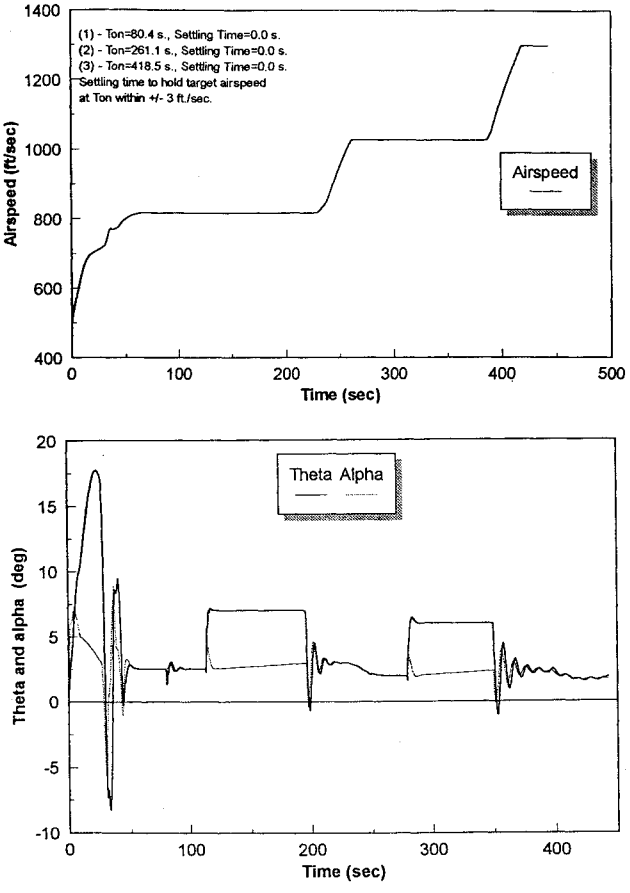

Fig. 3 Neural airspeed autopilot—low angles of attack. Upper part: airspeed vs time; lower part: α and $\theta z1$ vs time.

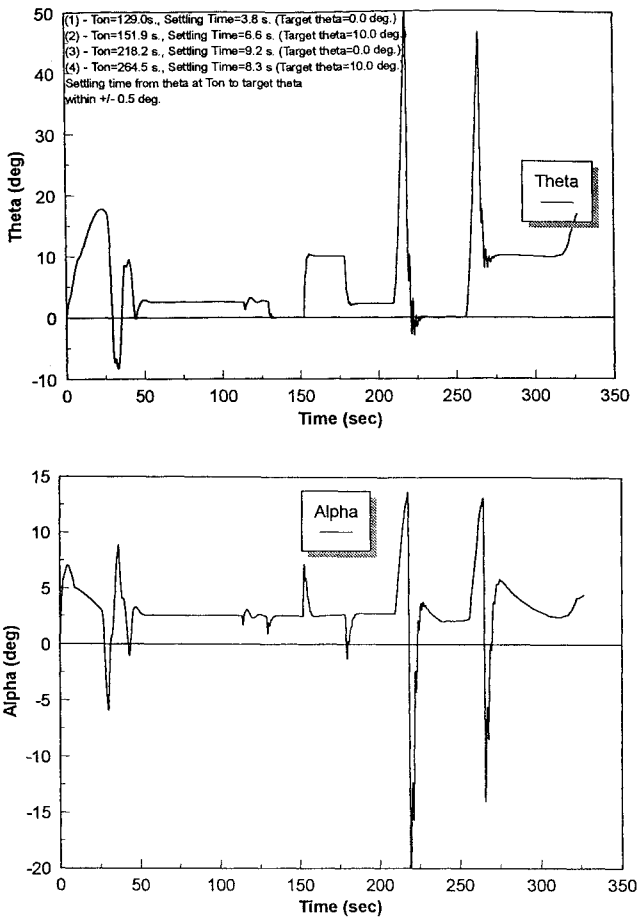

Fig. 4 Neural pitch autopilot—low and high angles of attack. Upper part: θ vs time; lower part: α vs time.

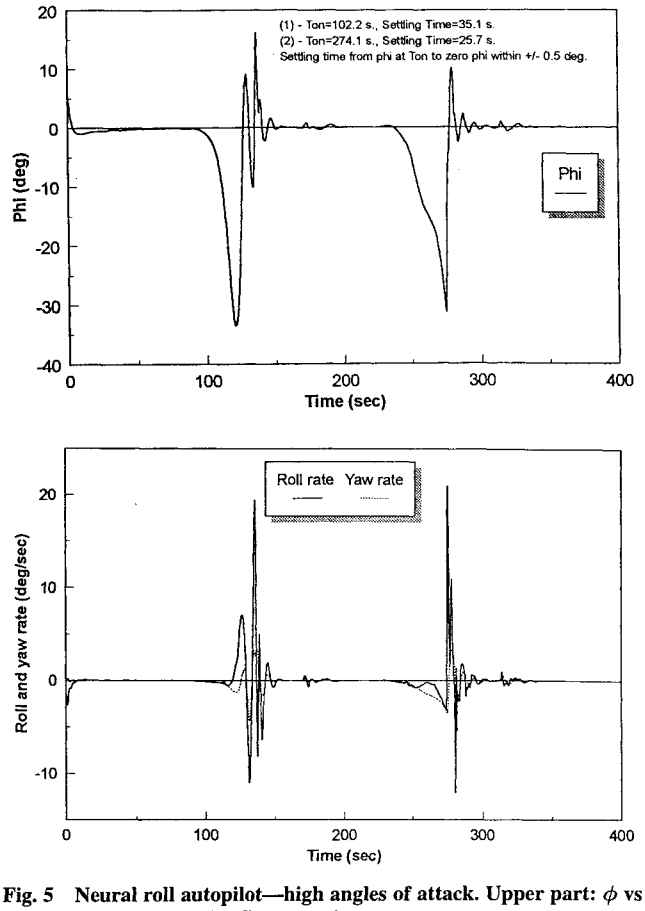

Fig. 5 Neural roll autopilot—high angles of attack. Upper part: ϕ vs time; lower part: α and roll rate vs time.

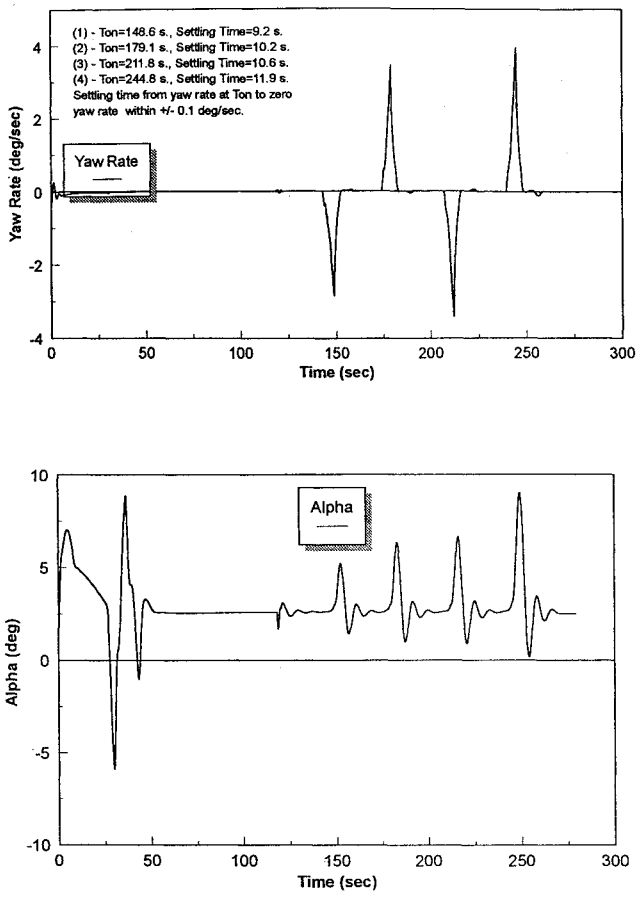

Fig. 6 Neural directional autopilot—low angles of attack. Upper part: yaw rate vs time; lower part: α vs time.

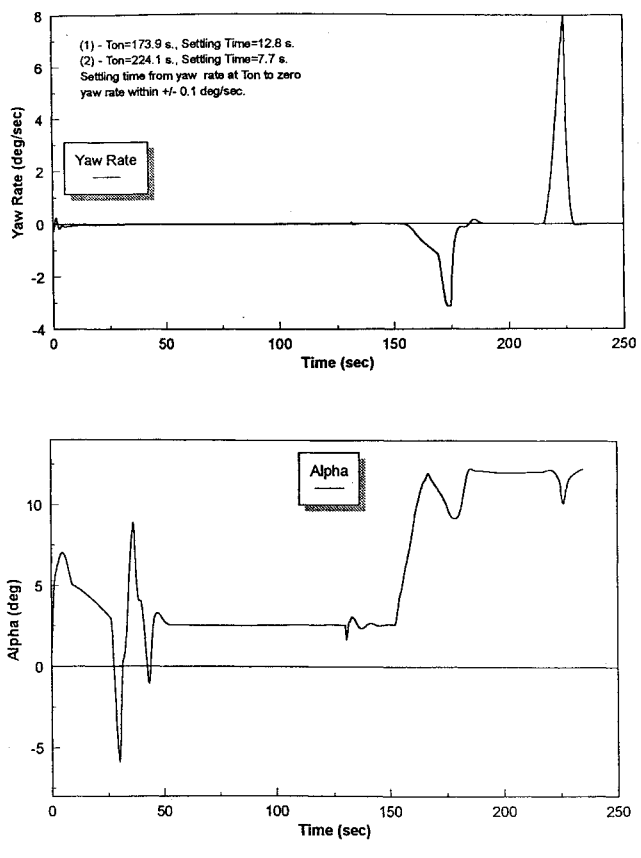

Fig. 7 Neural directional autopilot—high angles of attack. Upper part: yaw rate vs time; lower part: α vs time.

The latter are achieved with high nose-up elevator deflection at low thrust levels.

Figure 5 relates to the roll neural autopilot at high angles of attack. This roll autopilot is essentially used to regain wing-level flight conditions. Control over the directional dynamics was achieved through manual activation of the rudder and/or through the neural directional autopilot system. Again, the described use of the autopilot may not be typical of any flight phase; however, the overall goal is to test the behavior of the neural scheme at nonlinear conditions. It is likely that, at the conditions when the system is activated, some dynamic coupling effects (induced by the simultaneous high values for p, q, and r) are already present. Nevertheless, wing-level conditions are regained within a reasonable time.

Figures 6 and 7, finally, relate to the directional neural autopilot at low and high angles of attack. When activated the system is simply instructed to cancel any yaw rate and acceleration. Control over the rolling dynamics is retained through manual deflection of the ailerons and/or through the neural roll autopilot.

VI. Conclusions

This paper has presented the results of a preliminary investigation undertaken with the goal of showing the feasibility and the advantages of on-line learning neural controllers. A set of neural-based autopilot functions were simulated using the simulation code for an high-performance military aircraft.

Two studies were conducted. In the first study on-line learning neural controllers for airspeed and altitude autopilot functions trained with the extended back-propagation algorithm were compared with similar architectures trained with the back-propagation algorithm at 18 different set points within the flight envelope, showing that the first algorithm is capable of providing better performance. These simulations demonstrated the important on-line local learning capabilities of the neural controllers.

The second study had the objective of providing a primitive interface of the neural autopilots with a man in the loop through numerical simulation without graphic capabilities. This is a little-explored but very important issue with regard to true acceptability of neural controller schemes in flight control systems. Another objective of this study was to show the capabilities of neural controllers at nonlinear conditions. The results were very encouraging, showing good performance for the neural controllers at nonlinear conditions where conventional controllers do not have built-in robustness capabilities. This particular property can be very important for extending the application of neural controllers to more complex stability augmentation systems. Another factor that can, potentially, make the application of neural controllers very attractive is the possibility of implementing the neural schemes in parallel dedicated neural microprocessors.

In conclusion, this paper has shown the potential of on-line learning neural controllers and has provided an additional proof of the importance of using learning algorithms more advanced than the conventional back propagation for on-line learning purposes.

Acknowledgment

Partial support for the first author and support for the second author has been provided by NASA/West Virginia Space Consortium Grant NGT-40047.

References

[1]Anon., "AFTI/F-16 Development and Integration Program—DFCS Phase Final Technical Report," AFWAL-ET 84-3008, Vol. 1, Wright–Patterson AFB, OH, Dec. 1984.

[2]Stevens, B. L., and Lewis, F. L., *Aircraft Control and Simulation*, Wiley, New York, 1992.

[3]Athans, M., et al., "The Stochastic Control of the F-8C Aircraft Using a Multiple Model Adaptive Control (MMAC) Method—Part I: Equilibrium Flight," *IEEE Transactions on Automatic Control*, Vol. AC-22, No. 5, 1977, pp. 767–780.

[4]Elliott, J. R., "NASA's Advanced Control Law Program for the F-8 Digital Fly-by-Wire Aircraft," *IEEE Transactions on Automatic Control*, Vol. AC-22, No. 5, 1977, pp. 753–757.

[5]Kaminer, I., Khargonekar, P. P., and Robel, G., "Design of Localizer Capture and Track Modes for a Lateral Autopilot Using H_∞ Synthesis," *IEEE Control System Magazine*, Vol. 10, No. 4, 1990, pp. 13–21.

[6]Chiang, R. Y., Safonov, K. P., and Tekawy, J. A., "A Fixed H_∞ Controller for a Supermaneuverable Fighter Performing the Herbst Maneuver," *Proceedings of the 29th Conference on Decision and Control* (Honolulu, Hawaii), Dec. 1990, pp. 2599–2606.

[7]Voulgaris, P., and Valvani, L., "High Performance Linear Quadratic and H_∞ Designs for a 'Supermaneuverable' Aircraft," *Journal of Guidance, Control, and Dynamics*, Vol. 14, No. 1, 1991, pp. 157–165.

[8]Snell, S. A., Enns, D. F., and Garrard, W. L., "Nonlinear Inversion Flight Control for a Supermaneuver Aircraft," *Journal of Guidance, Control, and Dynamics*, Vol. 15, No. 4, 1992, pp. 976–984.

[9]Buffington, J., Adams, R., and Banda, S., "Robust, Nonlinear, High-Angle-of-Attack Control for a Supermaneuverable Vehicle," *Proceedings of the AIAA Guidance, Navigation, and Control Conference* (Monterey, CA), AIAA, Washington, DC, 1993 (AIAA Paper 93-3774).

[10]Balas, G. J., Reiner, J., and Garrard, W. L., "Robust Dynamic Inversion Control Law for Aircraft Control," *Proceedings of the AIAA Guidance, Navigation, and Control Conference* (Hilton Head, SC), AIAA, Washington, DC, 1992.

[11]Balas, G. J., Reiner, J., and Garrard, W. L., "Design of a Flight Control System for a Highly Maneuverable Aircraft Using μ Synthesis," *Proceedings of the AIAA Guidance, Navigation, and Control Conference* (Monterey, CA), AIAA, Washington, DC, 1993 (AIAA Paper 93-3774).

[12]Reiner, J., Balas, G. J., and Garrard, W. L., "Design of a Flight Control System for a Highly Maneuverable Aircraft Using Robust Dynamic Inversion," *Proceedings of the AIAA Guidance, Navigation, and Control Conference* (Scottsdale, AZ), AIAA, Washington, DC, 1994 (AIAA Paper 94-3682).

[13]Rumelhart, D., and McClelland, J., *Parallel Distributed Processing*, MIT Press, Cambridge, MA, 1986.

[14]Simpson, P. K., *Artificial Neural Systems*, Pergamon, Fairview Park, NY, 1990.

[15]Narendra, K. S., and Partasarathy, K., "Identification and Control of Dynamical Systems Using Neural Networks," *IEEE Transactions on Neural Networks*, Vol. 1, No. 1, 1990.

[16]Baker, W. L., and Farrell, J. A., "Learning Augmented Flight Control for High Performance Aircraft," *Proceedings of the AIAA Guidance, Navigation, and Control Conference* (New Orleans), AIAA, Washington, DC, 1991 (AIAA Paper 91-2836).

[17]Troudet, T., Garg, S., and Merrill, W. C., "Neural Network Application to Aircraft Control System Design," *Proceedings of the AIAA Guidance, Navigation, and Control Conference* (New Orleans), AIAA, Washington, DC, 1991 (AIAA Paper 91-2715).

[18]Ha, C. M., "Neural Network Approach to AIAA Aircraft Control Design Challenge," *Proceedings of the AIAA Guidance, Navigation, and Control Conference* (New Orleans), AIAA, Washington, DC, 1991 (AIAA Paper 91-2672).

[19]Napolitano, M. R., Neppach, C., Casdorph, V., Naylor, S., Innocenti, M., and Bini, F., "Sensor Failure Detection, Identification and Accommodation Using On-Line Learning Neural Architectures," *Proceedings of the AIAA Guidance, Navigation, and Control Conference* (Scottsdale, AZ), AIAA, Washington, DC, 1994 (AIAA Paper 94-3598).

[20]Napolitano, M. R., Chen, C. I., and Naylor, S., "Aircraft Failure Detection and Identification Using Neural Networks," *Journal of Guidance, Control, and Dynamics*, Vol. 16, No. 6, pp. 999–1008.

[21]Napolitano, M. R., Neppach, C., Casdorph, V., and Naylor, S., "On-Line Learning Nonlinear Direct Neuro Controllers for Restructurable Flight Control Systems," *Journal of Guidance, Control, and Dynamics*, Vol. 18, No. 1, pp. 170–176.

[22]Chen, C. L., and Nutter, R. S., "An Extended Back-Propagation Learning Algorithm by Using Heterogeneous Processing Units," *Proceedings of the International Joint Conference on Neural Network (IJCNN '92)* (Baltimore), June 1992.

[23]Cybenko, G., "Approximation by Superposition of Sigmoidal Functions," *Mathematics of Control Signals and Systems*, Vol. 2, No. 4, 1989, pp. 303–314.

[24]Brumbaugh, R. W., "An Aircraft Model for the AIAA Controls Design Challenge," *Proceedings of the AIAA Guidance, Navigation, and Control Conference* (New Orleans), AIAA, Washington, DC, 1991 (AIAA Paper 91-2631).

[25]Roskam, J., *Airplane Flight Dynamics and Automatic Flight Controls*, Roskam Aviation and Engineering Corp., 1979.

[26]Napolitano, M. R., Silvestri, G., Innocenti, M., Neppach, C., Naylor, S., and Casdorph, V., "Implementation of Hardware-Based On-Line Learning Neural Architectures for Sensor Failure Detection, Identification, and Accommodation" (submitted for publication).

[27]Naylor, S., "On-Line Learning Nonlinear Neural Controllers for Restructurable Flight Control Systems," Master Thesis in Aerospace Engineering, Dept. of Mechanical and Aerospace Engineering, West Virginia Univ., Aug. 1994.