JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 18, No. 1, January–February 1995

On-Line Learning Nonlinear Direct Neurocontrollers for Restructurable Control Systems

Marcello R. Napolitano,* Steve Naylor,† Charles Neppach,† and Van Casdorph†
West Virginia University, Morgantown, West Virginia 26506-6101

This paper describes an innovative approach to the problem of the on-line determination of a control law in order to achieve a dynamic reconfiguration of an aircraft that has sustained extensive damage to a vital control surface. The approach consists of the use of on-line learning neural network controllers that have the capability of bringing an aircraft, whose dynamics can become unstable after a substantial damage, back to an equilibrium condition. This goal has been achieved through the use of a specific training algorithm, the extended back-propagation algorithm (EBPA), and proper selection of the architectures for the neural network controllers. The EBPA has recently shown remarkable improvements over the back-propagation algorithm in terms of convergence time and local minimum problems. The methodology is illustrated through a nonlinear dynamic simulation of a typical combat maneuver for a high-performance aircraft.

Nomenclature

a_z = output of accelerometer aligned with aircraft z body axis, g
J = performance index
K = discrete time index
L = lower bound of modified sigmoid function
M = measurement vector
O = neural network output vector
p = aircraft angular velocity around x body axis, rad/s
q = aircraft angular velocity around y body axis, rad/s
r = aircraft angular velocity around z body axis, rad/s
T = slope of modified sigmoid function
t = time
U = upper bound of modified sigmoid function
α = angle of attack, rad
β = angle of sideslip, rad
θ = pitch angle, rad
ϕ = angle of roll, rad
ψ = angle of yaw, rad
δ = surface deflection, rad or deg

Subscripts

A = aileron
D = differential stabilator
H = symmetric stabilator
R = rudder

Introduction

THE ever-rising costs of new military aircraft and their relatively low procurement have been the reasons for an increasing interest, in the past few years, for the design of a flight control system able to accomplish the reconfiguration of the aircraft following battle damage.[1–3] Battle damage refers to damage sustained during a typical air-to-ground maneuver with ground fire or during an air-duel combat situation. Contemporary fly-by-wire control systems automatically detect sensor failures and reconfigure to mask the effects of these failures. They do not, however, cope with damaged aircraft surfaces.

In order to implement a self-accommodation strategy, we may introduce a variety of control surfaces (speed brakes, wing flaps, differential dihedral canards, spoilers, rudder below fuselage) and thrust control mechanisms (differential thrust, thrust vectoring, canted engines). The selection of the control mechanism to be used is a function of several factors: control effectiveness, increased aircraft complexity and costs, weight penalties, increased aerodynamic drag due to the increased wetted area, and applicability depending on aircraft type. Today self-accommodation requirements for relaxed static stability aircraft with a flight envelope that extends to nonlinear angles of attack call for an increase in the number of independent control surfaces. The traditional flight data, the actuator position for each surface, along with a fully operational flight computer are assumed to be available.

We can classify the failures and/or battle damage of a control surface in two categories: locked surface and missing surface. Generally, we can say that a locked surface corresponds to a failure in the control surface's actuator. Battle damage mainly implies missing surface or, more realistically, both missing and locked surface.[2,3] In addition to any nonlinear dynamic and aerodynamic conditions that may occur at the time of damage, the dynamic coupling between longitudinal and lateral directional dynamics that exists following any type of control surface failure must be considered. This may lead to a loss of stability and, possibly, to unrecoverable flight conditions. This is especially true for those aircraft that, being open-loop unstable, rely on a feedback flight control system to coordinate movement of various control surfaces to maintain stability and to achieve level 1 handling qualities throughout the entire flight envelope. If properly and successfully implemented, a self-accommodation process will achieve, in increasing order of importance, 1) a lower damage-induced handling qualities degradation, 2) a lower mission abort rate, and 3) a lower aircraft loss rate.

There is a fine distinction between reconfigurable and restructurable flight control systems. In reconfigurable flight control systems the types of control surface failures are classified and anticipated and the relative change in control gains are calculated off-line, using any selected control scheme, and stored in the memory of the flight control computer, ready for on-line use whenever needed. On the contrary, in restructurable flight control systems, the reconstruction of the control law is performed on-line. In this case, the availability of sufficient computational power, as opposed to memory, can potentially become a critical issue. The battle damage and/or generic failure accommodation includes the following two tasks: actuator failure detection and identification (AFDI) and actuator failure accommodation (AFA).

Received Oct. 29, 1993; revision received April 28, 1994; accepted for publication May 4, 1994.

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

One of the contributions of this paper is to propose a restructurable flight control system where the AFA task is divided into the following steps:

Step 1. A nonlinear control law is formulated on-line to bring the aircraft back to a new equilibrium condition before the aircraft enters unrecoverable flight conditions.

Step 2. Once back to stable and linear conditions, a linear control method is implemented for restructuring the control laws in order to continue the mission with the least amount of degradation to the handling qualities.

In the last decade there has been the introduction of several control strategies related to reconfigurable flight control systems whereas only more recently restructurable flight control systems have received more attention.

A partial list of control methodologies for reconfigurable flight control systems is given by pseudo-inverse controller or control-mixer method,[4,5] multiple model adaptive control,[6,7] and quantitative feedback theory.[8]

A disadvantage of reconfigurable control methods is that they involve an extensive design to accommodate all possible control system failures at different conditions in the aircraft flight envelope. This in turn requires extensive memory space in the on-board computer. Also, these are linear techniques and, therefore, not perfectly suitable to the self-accommodation problem characteristics.

A partial list of design approaches for restructurable flight control system is given by proportional-integral implicit model-following control,[9] explicit model-following control,[10] model reference adaptive control,[11,12] feedback linearization methods,[13] and eigenstructure assignment methods.[14]

Similarly, a disadvantage of each of these techniques is that they are linear in nature, and therefore, the degree of success of their applications to an aircraft system that has entered nonlinear dynamic conditions following damage is questionable. Nevertheless, with different degrees of importance, they all represent important steps into the solution of this complex control problem.

The main objective of this paper is to introduce a neural-network-based approach for the on-line design of a nonlinear controller that allows us to accomplish step 1 of the AFA process, that is, to regain control of the aircraft following damage before it enters unrecoverable flight conditions. This is achieved through an on-line learning neural network (NN) architecture trained with a new algorithm, the extended back-propagation algorithm (EBPA), which has shown improvements over the (standard) back-propagation algorithm (SBPA)[15,16] in terms of convergence time and local minimum problems. A complete description of the EBPA and a comparison of its performances vs the SBPA[15,16] performances is given in Ref. 17.

It is assumed that successful failure detection and identification (FDI) following a control surface damage has already been achieved. The authors have previously introduced another NN-based approach for the AFDI task. The results for the simulation of the FDI process at linear and highly nonlinear conditions are shown in Refs. 18 and 19.

This paper is organized as follows. The next section suggests the use of on-line learning NN controllers that can take advantage of the performance of the EBPA. Another section discusses the numerical simulation of the self-accommodation process following control surface damages typical of a combat situation for a high-performance military aircraft. A final section summarizes the paper with conclusions and recommendations.

Neural Network Controllers for Restructurable Flight Control Systems

Very recently NN theory has been proposed as an alternative approach for the design of flight control systems. It has been shown[15,16] that NN architectures, trainable with different algorithms, have the capabilities to learn the dynamics of complex physical systems. Furthermore, it was recently proved[20] that a NN with at least one hidden layer is guaranteed to be able to reproduce the input-output mapping of any arbitrary nonlinear function. This property can be extremely useful when the input-output data are related to an unknown, unmodeled system. The implications of this property with the restructurable flight control system problem are clear.

Although this capability is undoubtedly very attractive, the implementation of NN controllers must be analyzed very carefully. A key issue in the design of a NN controller is the selection of an on-line training vs off-line training working mode. On-line training implies that the NN has the capability of changing the values of the numerical components that make up its architecture in real time. Off-line training implies that the NN operating on-line had been previously trained and it has a frozen numerical architecture. Given that an aircraft that is suddenly damaged is a time-varying system and there are no previous models of damage to train with off-line, only on-line training can be considered for restructuring the flight control system.

For on-line learning, a point of particular concern is the length of the training, which is dependent on the required accuracy and, ultimately, on the training algorithm. Another point of concern is the complexity of the NN architecture, which is then related, along with the complexity of the training algorithm, to the required computational effort.

From the above discussion it is apparent that the performance and the acceptability of an on-line learning NN controller are strictly related to the performance of its training algorithm. To date, the majority of the training for the NN is being performed with the SBPA,[15,16] which is a gradient-based optimization method. At present, few studies related to the design of NN controllers for reconfigurable flight control systems have been presented.[21–23] Particularly, Refs. 21 and 22 describe the design of a NN controller with off-line training with the SBPA.

There are several drawbacks associated with the SBPA.[15,16] Among these, the learning speed is slow for large-order systems, implying long training processes. Also, in the presence of local minima, the SBPA is often not able to find a set of weights for the NN. These two problems are mutually related and interactive. For example, it has been shown that the learning speed can be improved by setting a larger learning rate. Unfortunately, this also increases the possibility for the SBPA to be trapped in a local minimum or to oscillate around the global minimum.

These and other problems may be solved by introducing a heterogeneous network,[17–19,24] meaning that each neuron in the hidden and output layer of the NN has its own output capability of updating free parameters such that 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). Figure 1 shows a three-layer NN architecture trained with the EBPA.

Numerical Simulation of Control Law On-Line Reconstruction

The proposed controller scheme can essentially be classified as a direct neurocontroller where the control action does not specifically follow a model but tries to reach specified equilibrium conditions set in the NN cost function. A set of NN controllers, trained on-line with the EBPA, combining the NN mapping capabilities at nonlinear conditions with the EBPA's quick training capabilities, will achieve the crucial goal of Step 1 of the AFA process. The block diagram of the overall self-accommodation process is shown in Fig. 2.

Step 1 of the flight control on-line reconstruction has been numerically simulated using the Fortran software recently distributed for the AIAA Control Design Challenge. This software provides a numerical nonlinear simulation of an aircraft model, with full-envelope nonlinear aerodynamics and full-envelope thrust with first-order engine response data.[25] The aircraft modeled is a high-performance, supersonic vehicle representative of current fighters. The aircraft's primary flight control surfaces consist of horizontal stabilators capable of symmetric or differential movement, conventional ailerons, and a single vertical rudder. There are a total of five actuators: two ailerons, two stabilators, and one rudder. The model includes identical actuators for all surfaces. The aerodynamics are modeled for the full aircraft envelope using multidimensional tables and linear interpolation to form nonlinear function generators.

Two types of damages were considered. The first type assumes that a damage occurs to the left stabilator involving a stuck actuator

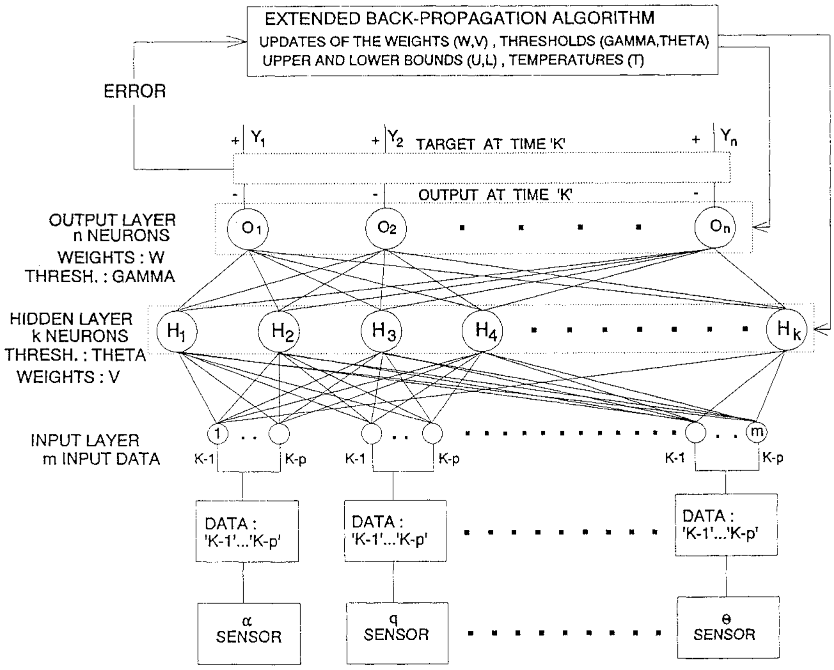

Fig. 1 Three-layer neural network implemented for failure detection.

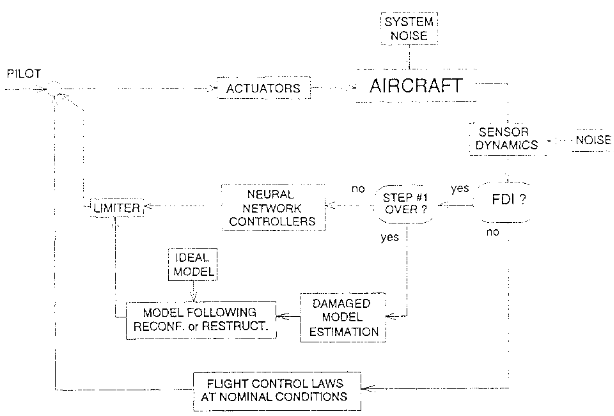

Fig. 2 Block diagram of flight control reconfiguration or on-line reconstruction process.

at −20 deg and a reduction of 50% of the aerodynamic surface. The second type assumes that a damage occurs to the right aileron involving a stuck actuator to +15 deg and a 50% surface reduction. In order to have a realistic simulation, it was necessary to model the aerodynamic effects of the damage in terms of modifications to the values of all the relative aircraft stability derivatives. The aerodynamic modeling procedures are described in Refs. 1–3.

Following both types of damages, the immediate goals would be to regain equilibrium and to compensate for the damage-induced pitching and rolling moments. Therefore, two separate NN controllers are introduced: the first for the pitch control and the second for the roll control. Control over the yawing dynamics proved to be unnecessary but could also be implemented.

For the NN pitch controller the input data are given by the elements of the vector M_{pitch}:

$$M_{\text{pitch}} = [a_z, \alpha, \theta, q, \dot{q}]^T \tag{1}$$

from time instant $K - 1$ down to $K - 2$. The two hidden layers of the NN architecture contain 10 and 18 neurons. The output of the NN pitch controller is the compensating deflection for the remaining healthy right stabilator (for type 1 damage) and the symmetric stabilators (for type 2 damage). The training for the NN pitch controller is turned on as the simulation starts. At nominal conditions the pitch NN controller is trained to emulate the actual control deflection for the symmetric stabilators, and therefore, it minimizes the cost function:

$$J_{\text{pitch}_{\text{nom}}} = (\delta_{H_{L,R}} - \hat{\delta}_{H_{L,R}}) \tag{2}$$

When the damage has been detected and isolated, the on-line learning NN pitch controller will switch its target and will start to minimize the cost function:

$$J_{\text{pitch}_{\text{dam}}} = \tfrac{1}{10}(q - q_{\text{des}}) + \tfrac{1}{10}(\theta - \theta_{\text{des}}) + \tfrac{1}{10}(\dot{q} - \dot{q}_{\text{des}}) \tag{3}$$

with all the desired values selected to be zero except for $\theta_{\text{des}} = 2$ deg $= 0.035$ rad. It can be said that this cost function resembles a controller with a proportional-integral-derivative (PID) error formulation.

Note that the on-line training at nominal conditions has no physical meaning; in fact, the NN pitch controller is just "mimicking" control deflections at nominal conditions. This has the benefit of having the NN output within the same order of magnitude of the NN output necessary when the on-line reconstruction will be turned on following a positive FDI.

A similar NN roll controller will operate in the same fashion. The input data are given by the elements of the vector M_{roll},

$$M_{\text{roll}} = [\phi, p, \dot{p}, r, \beta, \psi]^T \tag{4}$$

from time instant $K - 1$ down to $K - 2$. The two hidden layers contain 10 and 18 neurons. The output of the NN roll controller is the compensating deflection for the ailerons (for type 1 damage) and the asymmetric stabilator deflections (for type 2 damage). At nominal conditions the NN roll controller is trained to emulate the actual aileron deflections, and therefore, it minimizes the cost function:

$$J_{\text{roll}_{\text{nom}}} = \delta_A - \hat{\delta}_A \tag{5}$$

Following a positive FDI the on-line learning NN roll controller will switch its target and will start to minimize the cost function:

$$J_{\text{roll}_{\text{dam}}} = \tfrac{1}{10}(p - p_{\text{des}}) + \tfrac{1}{10}(\phi - \phi_{\text{des}}) + \tfrac{1}{10}(\dot{p} - \dot{p}_{\text{des}}) \tag{6}$$

with all the desired values selected to be zero. The on-line training at nominal conditions serves the same purpose as the on-line training at nominal conditions for the NN pitch controller. It also has to be stressed that in the following simulations the compensating control deflections calculated by the two NN controllers are within the maximum and minimum deflection and deflection rates, as reported in Ref. 25. This is the task of the limiter in Fig. 2. Therefore, the modeling of control saturation problems is ensured.

The simulation starts at an altitude of 18,700 ft at a speed of 650 ft/s. A highly nonlinear maneuver is simulated with large deflections for the stabilators and the ailerons.

The responses achievable with an on-line learning NN controller trained with the EBPA are compared with a similar architecture SBPA-trained NN controller and with conventional autopilot pitch- and roll-hold control systems, assumed to be turned on following a positive AFDI. Of course, it is not expected that the autopilot control laws, designed for nominal conditions and linear flight regimes, can perform acceptably, but the intention was just to compare the NN controller performance with some other control scheme. Of much more interest is the comparison of the performances of the SBPA- vs EBPA-trained NN controllers. The architecture of the SBPA-trained NN controllers is similar to the EBPA-trained NN in the sense that they have the same number of inputs, outputs, and neurons in the hidden layers. The complete architectures of the two NN controllers trained with the EBPA is shown in Tables 1 and 2.

The type 1 and type 2 damage simulations and relative on-line reconstructions are described next.

Table 1 Architecture of the NN pitch controller

	EBPA	SBPA
Architecture		
Number of input parameters	5	5
Window size (P)	2	2
Number of input data (m)	10	10
Number of elements in the HL_1 (k_1)	10	10
Number of elements in the HL_2 (k_2)	18	18
Number of elements in the OL (n)	1	1
Learning rates		
ηs[a]	0.1	0.1
Momentum coefficients		
αs[a]	0	0
Parameters to be updated		
Weights: IL to HL_1 ($m \times k_1$)	100	100
Thresholds: HL_1 (k_1)	10	10
Upper bounds for HL_1 (k_1)	10	—
Lower bounds for HL_1 (k_1)	10	—
Temperatures for HL_1 (k_1)	10	—
Weights: HL_1 to HL_2 ($k_1 \times k_2$)	180	180
Thresholds: HL_2 (k_2)	18	18
Upper bounds for HL_2 (k_2)	18	—
Lower bounds for HL_2 (k_2)	18	—
Temperatures for HL_2 (k_2)	18	—
Weights: HL_2 to OL ($k_2 \times n$)	18	18
Thresholds: OL (n)	1	1
Upper bounds for OL (n)	1	—
Lower bounds for OL (n)	1	—
Temperatures for OL (n)	1	—
Total	414	327

[a]From Refs. 15–17.

Table 2 Architecture of NN roll controller

	EBPA	SBPA
Architecture		
Number of input parameters	6	6
Window size (p)	2	2
Number of input data (m)	12	12
Number of elements in HL_1 (k_1)	10	10
Number of elements in HL_2 (k_2)	18	18
Number of elements in OL (n)	1	1
Learning rates		
ηs[a]	0.1	0.1
Momentum coefficients		
αs[a]	0	0
Parameters to be updated		
Weights: IL to HL_1 ($m \times k_1$)	120	120
Thresholds: HL_1 (k_1)	10	10
Upper bounds for HL_1 (k_1)	10	—
Lower bounds for HL_1 (k_1)	10	—
Temperatures for HL_1 (k_1)	10	—
Weights: HL_1 to HL_2 ($k_1 \times k_2$)	180	180
Thresholds: HL_2 (k_2)	18	18
Upper bounds for HL_2 (k_2)	18	—
Lower bounds for HL_2 (k_2)	18	—
Temperatures for HL_2 (k_2)	18	—
Weights: HL_2 to OL ($k_2 \times n$)	18	18
Thresholds: OL (n)	1	1
Upper bounds for OL (n)	1	—
Lower bounds for OL (n)	1	—
Temperatures for OL (n)	1	—
Total	434	347

[a]From Refs. 15–17.

Type 1 Damage

At a certain discrete time instant $K = K_{\text{dam}}(t = K_{\text{dam}}, \Delta t = 180$ s) type 1 damage is simulated, causing instantaneous change of the lift, pitching, and rolling moment coefficients. Following a positive FDI, the NN pitch and roll controllers will attempt to regain control of the aircraft as described above. Figures 3 and 4 show the pitch angle and the pitch rate time histories for the three different control schemes. It can be seen that the control law of the pitch-hold system in the autopilot makes an attempt to regain control of the aircraft but without much success. The SBPA-trained NN pitch controller, with similar architecture and identical learning rates as in the EBPA pitch controller, also shows a tendency toward achieving the control goal until the simulated aircraft crashes to the ground (altitude becomes negative). The length of the transient

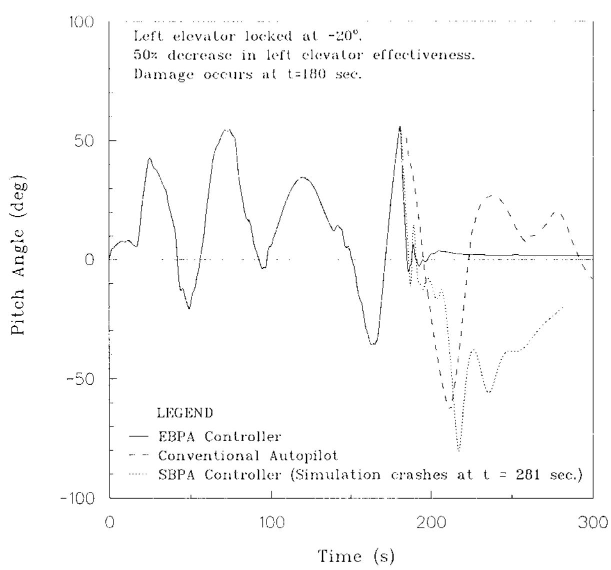

Fig. 3 Type 1 damage: plot of pitch angle vs time.

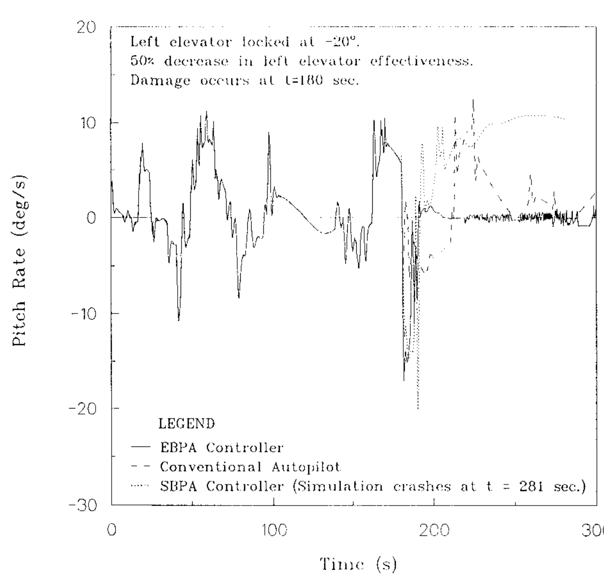

Fig. 4 Type 1 damage: plot of pitch rate vs time.

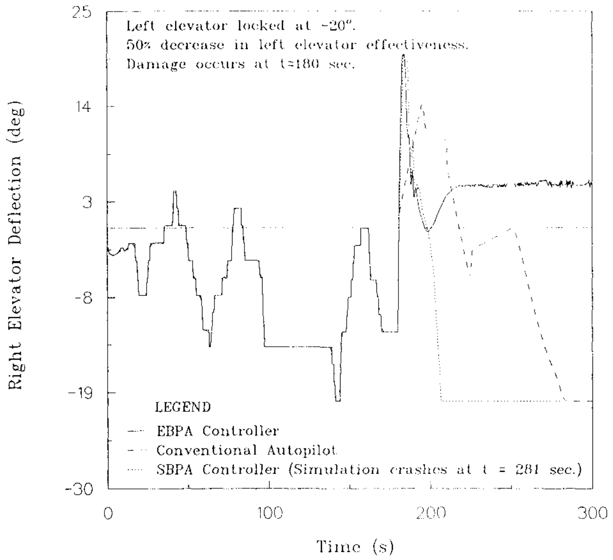

Fig. 5 Type 1 damage: plot of right stabilator deflections vs time.

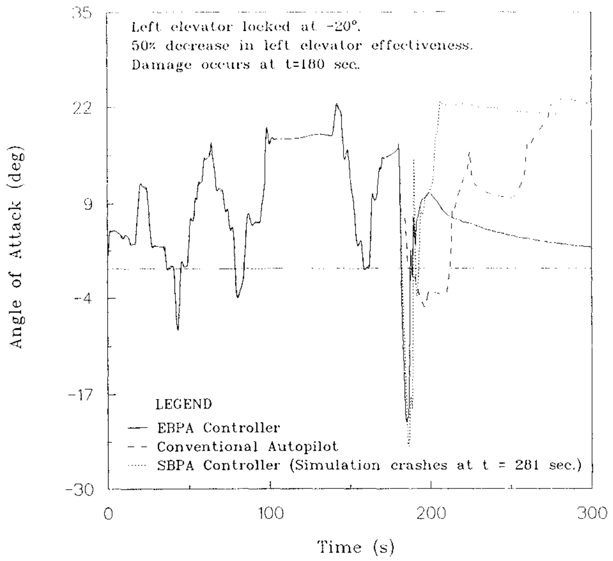

Fig. 6 Type 1 damage: plot of angle of attack vs time.

is consistent with the well-known problem of long convergence times associated with the SBPA. The superior performance of the EBPA-trained NN pitch controller is clear in these figures. Such a controller will achieve θ_{des} with a reasonable transient (about 20 s) without excessive oscillations. The associated compensating deflections for the "healthy" right stabilator are shown in Fig. 5 for the three control schemes. It is clear that the pitch autopilot control laws quickly induce a saturation for the compensating control deflection. To a smaller extent a similar problem is experienced by the SBPA-trained NN pitch controller. These problems do not apply for the compensating deflections calculated by the EBPA-trained NN pitch controller. The EBPA deflections exhibit reasonable transient characteristics with only a moderate level of activities for the limiter. The angle-of-attack time history for the different control schemes is shown in Fig. 6. The unacceptable responses associated with both the autopilot pitch-hold control laws and the SBPA-trained NN controller are also evident, whereas the EBPA controller confirms its desirable performance.

A similar comparison has been performed between the EBPA- and SBPA-trained NN roll controllers and the standard autopilot roll control system. The objective was to control the damage-induced rolling moment before it causes inertial coupling problems. Figures 7 and 8 show the bank angle and the roll rate time histories. Especially from Fig. 8, it can be seen again that the nominal autopilot control laws and the SBPA NN roll control do not provide any real compensating

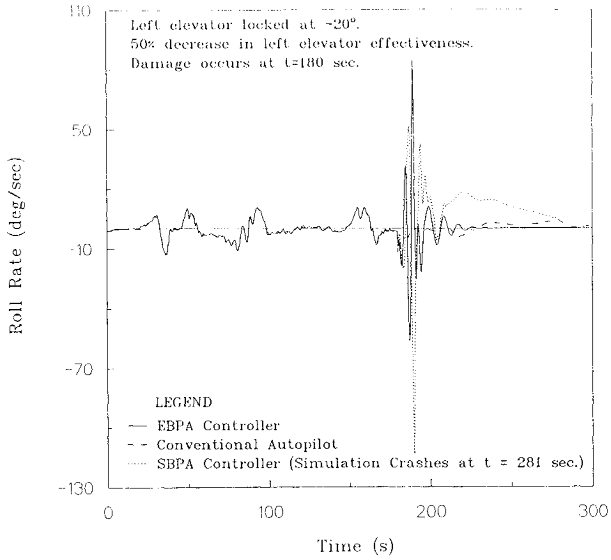

Fig. 7 Type 1 damage: plot of bank angle vs time.

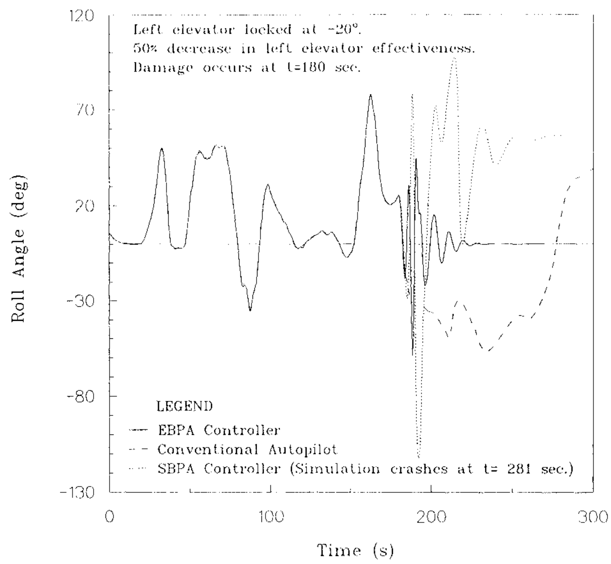

Fig. 8 Type 1 damage: plot of roll rate vs time.

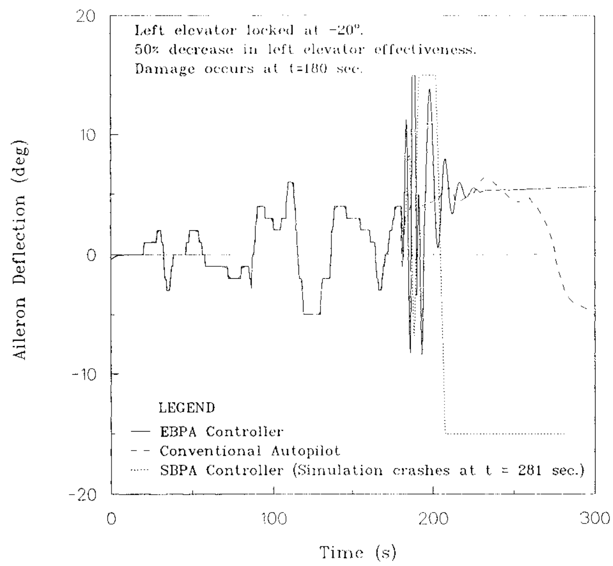

Fig. 9 Type 1 damage: plot of aileron deflections vs time.

action. However, the EBPA-trained NN roll controller still provides the best control action. The compensating aileron deflections, output of the roll controllers, are shown in Fig. 9.

Type 2 Damage

A similar set of plots show the results relative to the numerical simulation of type 2 damage, involving a damage to the right aileron. The burden of the on-line reconstruction was carried by the stabilators used asymmetrically in order not to saturate the healthy left-aileron surface.[1]

Due to the moment arms of the control surfaces with respect to the center of gravity, it is clear that for type 2 damage the induced pitching moment is much less critical than the rolling moment induced by type 1 damage. Again we will consider the NN pitch and roll controllers trained on-line with both SBPA and EBPA along with the autopilot pitch- and roll-hold schemes. Figures 10–12 show the roll rate, the bank angle, and the pitch angle with the damage starting at $K = K_{\mathrm{dam}}(t = K_{\mathrm{dam}}, \Delta t = 180\ \mathrm{s})$. Both the SBPA NN controllers and the autopilot control law do not seem to provide an appropriate control action. The SBPA confirms its problems associated with long training times. Once again, the EBPA NN pitch and roll controllers perform in a very desirable fashion. The differential deflections for the right and left stabilators, providing the compensating rolling and pitching moments, are shown in Figs. 13 and 14.

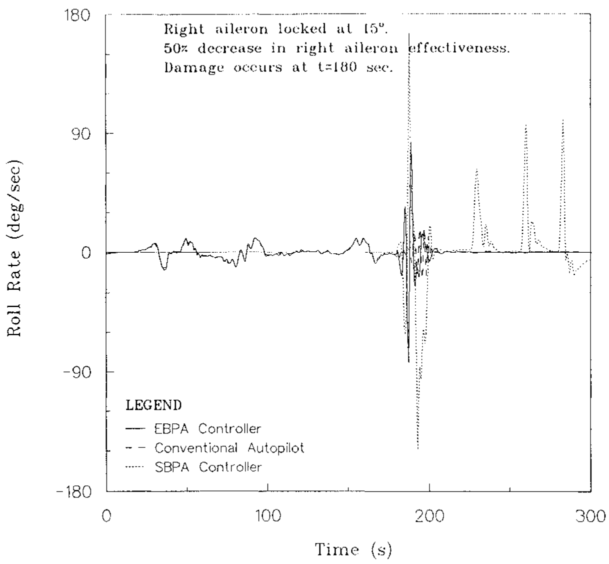

Fig. 10 Type 2 damage: plot of roll rate vs time.

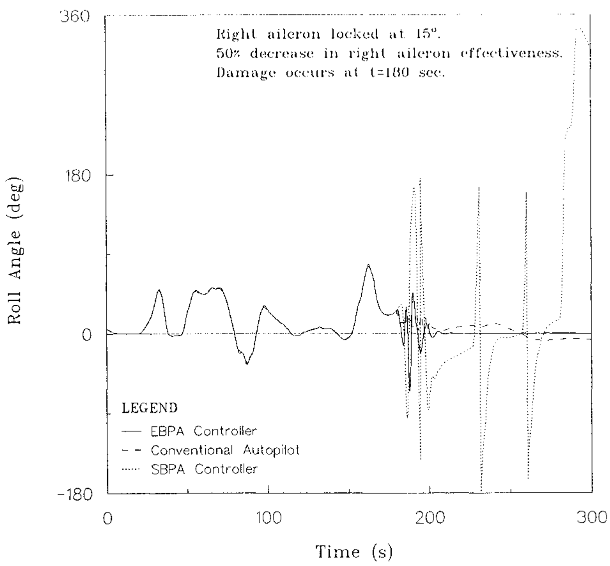

Fig. 11 Type 2 damage: plot of bank angle vs time.

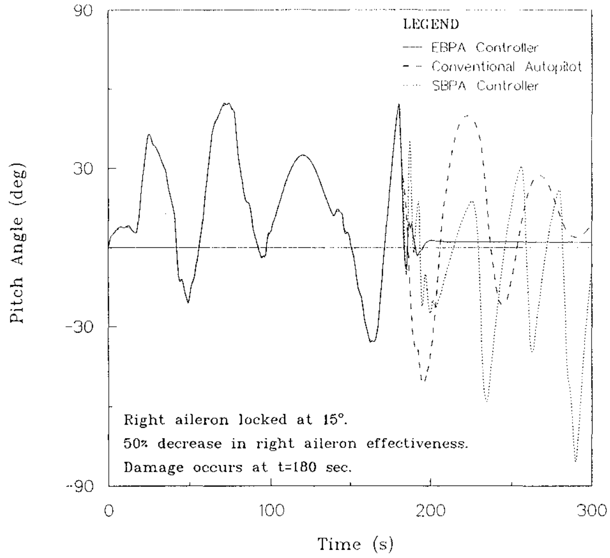

Fig. 12 Type 2 damage: plot of pitch angle vs time.

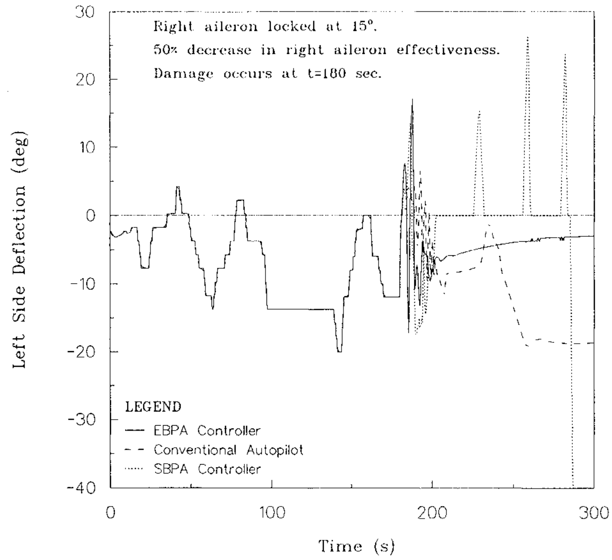

Fig. 13 Type 2 damage: plot of asymmetric stabilator deflections vs time, left-side surface deflection.

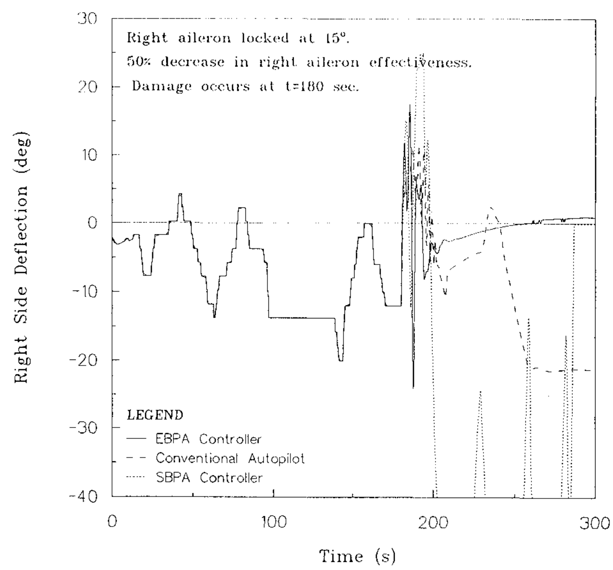

Fig. 14 Type 2 damage: plot of asymmetric stabilator deflections vs time, right-side surface deflection.

After having shown the results of the simulations with the implementation of NN controllers trained with both EBPA and SBPA, it is very important to try to speculate on the reasons behind the superior performances of the EBPA vs SBPA on-line training, even if the topic may be of more interest to the NN technical community. From an analysis of the NN architectures in Tables 1 and 2 it can be seen that the EBPA requires only a relatively modest increase in computational efforts with respect to the SBPA (precisely only 26 and 25% more parameters to update for the pitch and roll controllers, respectively). The capability of updating the upper and lower bounds for the output neurons with the EBPA should not be, in this case, a critical feature. In fact, the calculated compensating deflections are well within ± 1 rad, and there is no need to shift the bounds of the output neurons. Therefore, it is believed that most of the benefits of using the EBPA vs the SBPA come from the capability of adjusting the temperature parameters (T) for each neuron, which then directly affects the gradients, which in turn adjusts the error signals and the updating procedures, as shown in Refs. 17–19 and 24.

Conclusions

This paper has presented the implementation of a NN based approach for the first step of the actuator failure accommodation in restructurable flight control systems. The approach consists of the on-line design of a nonlinear control law immediately following control surface damages involving nonlinear dynamic and aerodynamic conditions. The paper presents results from nonlinear simulations with two types of very critical control surface damages. The achievements of the design goals have been made possible through a combination of the inherent capabilities of a NN architecture for the mapping of nonlinear dynamics with the fast and accurate training capabilities offered by the EBPA. The proposed algorithm is relatively simple to implement in a software code and should not require major computational efforts. It is expected that in the near future the flight control system community will consider the use of modern superfast NN-based microprocessors for parallel NN computations.

As a concluding statement, the authors believe that, given the level of effectiveness shown in this study, the approach of using on-line learning NN controllers at highly nonlinear conditions opens a wide range of possible applications for the restructurable controls problem and for the design of adaptive flight control systems.

References

[1]Napolitano, M. R., "A New Approach to the Flight Control System Reconfiguration Following a Battle Damage and/or a Generic Failure on a Control Surface," Ph.D. Dissertation, School of Mechanical and Aerospace Engineering, Oklahoma State University, Dec. 1989.

[2]Napolitano, M. R., and Swaim, R. L., "New Technique for Aircraft Flight Control Reconfiguration," *Journal of Guidance, Control, and Dynamics*, Vol. 14, No. 1, 1991, pp. 184–190.

[3]Napolitano, M., and Swaim, R., "New Techniques for Aircraft Flight Control Reconfiguration," *Control and Dynamic Systems*, Vol. 52, 1992, pp. 155–228.

[4]Rattan, K. S., "Evaluation of Control Mixer Concept for Reconfiguration of Flight Control System," *Proceedings of the 22nd IEEE Conference on Decision and Control*, Inst. for Electrical and Electronics Engineers, Piscataway, NJ, 1985, pp. 560–569.

[5]Raza, S. J., and Silverthorn, J. T., "Use of the Pseudo-Inverse for Design of a Reconfigurable Flight Control System," AIAA Paper 85-1900, Aug. 1985.

[6]Maybeck, P. S., and Pogoda, D. L., "Multiple Model Adaptive Controller for the STOL/F-15 with Sensor Actuator Failures," *Proceedings of the 28th Conference on Decision and Control*, Tampa, FL, Dec. 1989, pp. 1566–1572.

[7]Maybeck, P. S., and Stevens, R. D., "Reconfigurable Flight Control Via Multiple Model Adaptive Control Methods," *Proceedings of the 29th Conference on Decision and Control* (Honolulu, HI), 1990, pp. 3351–3356.

[8]Horowitz, I., Arnold, P. B., and Houpis, C. H., "YF-16—CCV Flight Control System Reconfiguration Design Using Quantitative Feedback Theory," *Proceedings of the National Aerospace and Electronics Conference*, Inst. of Electrical and Electronic Engineers, New York, 1985.

[9]Huang, C. Y., and Stengel, R. F., "Restructurable Control Using Proportional-Integral Implicit Model Following," *Journal of Guidance, Control, and Dynamics*, Vol. 13, No. 2, 1990, pp. 303–309.

[10]Robinson, A. C., "Totally Robust Control—A New Concept for Design of Flight Control Systems," AIAA Paper 85-1974, Aug. 1985.

[11]Morse, W., and Ossman, K., "Flight Control Reconfiguration Using Model Reference Adaptive Control," *Proceedings of the 1988 American Control Conference*, June 1988, pp. 2219–2224.

[12]Ha, C. M., "Reconfigurable Aircraft Flight Control System Via Robust Direct Adaptive Control," AIAA Paper 90-3226, Sept. 1990.

[13]Ochi, Y., "Application of Feedback Linearization Method in a Digitally Restructurable Flight Control System," *Journal of Guidance, Control, and Dynamics*, Vol. 16, No. 1, 1993.

[14]Napolitano, M. R., and Swaim, R. L., "Redesign of the Feedback Structure Following a Battle Damage and/or a Failure on a Control Surface by Eigenstructure Assignment," AIAA Paper 91-2626, Aug. 1991.

[15]Rumelhart, D., and McClelland, J., *Parallel Distributed Processing*, Massachusetts Inst. of Technology Press, Cambridge, MA, 1986.

[16]Simpson, P. K., *Artificial Neural System*, Pergamon, Fairview Park, NY, 1990.

[17]Napolitano, M. R., Neppach, C., and Casdorph, V., "A New Training Algorithm for an On-Line Neural Network Based State Estimation," *IEEE Transactions on Neural Networks* (submitted for publication).

[18]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, 1993, pp. 999–1008.

[19]Napolitano, M. R., Neppach, C., Casdorph, V., and Naylor, S., "Failure Detection and Identification Following Battle Damage in a Combat Situation," *IEEE Transactions on Neural Networks* (submitted for publication).

[20]Lapedes, A., and Farber, R., "How Neural Nets Work," *Proceedings of the International Joint Conference on Neural Networks*, IJCNN '88, Denver, CO, Nov. 1988.

[21]Huang, C., Tylock, J., Engel, S., Whitson, J., and Eilbert, J., "Failure-Accomodating Neural Network Flight Control," AIAA Paper 92-4394, Aug. 1992.

[22]Ha, C. M., Wei, Y. P., and Bessolo, J. A., "Reconfigurable Aircraft Flight Control System Via Neural Networks," AIAA Paper 92-1075, Feb. 1992.

[23]Mears, M., Smith, R., Chandler, P., and Patcher, M., "Hopfield Neural Network for Adaptive Control," AIAA Paper 93-3729, Aug. 1993.

[24]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, MD, June 1992.

[25]Brumbaugh, R. W., "*An Aircraft Model for the AIAA Controls Design Challenge*," NASA Contractor Report 186019, Dec. 1991.