Comparative analisys of neural control systems within the NASA IFCS F-15 WVU simulator

COMPARATIVE ANALISYS OF NEURAL CONTROL SYSTEMS WITHIN THE NASA IFCS F-15 WVU SIMULATOR

M. Battipede[*], P.A. Gili[+], M. Lando[+], M.R. Napolitano[#], M.G. Perhinschi[#], G. Campa[#]
[*]*Department of Mechanics, Polytechnic of Turin, 10129 Turin, Italy*
[+]*Department of Aeronautical and Space Engineering*
Polytechnic of Turin, 10129 Turin, Italy
[#]*Department of Mechanical and Aerospace Engineering*
West Virginia University, Morgantown WV 26505, USA

ABSTRACT

This paper describes the results of a study comparing the performance of three different approaches for the fault tolerant flight control system of the F-15 aircraft within the *Gen 2* NASA IFCS project. The common feature of the 3 methods is that they are all based on neural algorithms, however neural networks (NN) are used in different ways. The first approach features a Non Linear Dynamic Inversion (NLDI) technique augmented with a neural network to cancel the dynamic inversion errors. The second approach is based on a robust controller using both the Stochastic Optimal Feedforward and Feedback Technique (SOFFT) and a neural network to compensate errors. Finally, the third approach performs a neural network controller based on an adaptive predictor-corrector control strategy. The comparison study has been carried out using the WVU IFCS F-15 simulation environment. The results are given in terms of performance comparison and control activity evaluation for different maneuvers.

List of Acronyms

ARE	Algebraic Riccati Equations
CAS	Control Augmentation System
EMFC	Explicit Model Following Control
FDC	Flight Dynamics and Control
ICA	Integral of the Control Activity
IFCS	Intelligent Flight Control System
LQR	Linear Quadratic Regulator
MIMO	Multi-Input-Multi-Output
MISO	Multi-Input-Single-Output
MLP	Multi Layer Perceptron
NLDI	Non Linear Dynamic Inversion
NN	Neural Network
PTNN	Pre-Trained Neural Network
RPLR	Recursive Pseudo-Linear Regression
SISO	Single-Input-Single-Output
SOFFT	Stochastic Optimal Feedforward and Feedback Technique
WVU	West Virginia University

List of Symbols

$[A,B,C,D,H]_x$	State-space reference plant model matrices
$[A,B,C,D,H]_z$	State-space command model matrices
B	State-space system control matrix
C_1, C_2, C_3	Sigma-Pi NN variables
$\boldsymbol{E}$	Quadratic error function
G	User selected specific gain
$\boldsymbol{K}_p$	Gain matrix
K_i	Integrative constant
K_p	Proportional constant
K_u, K_x, K_z	Feedforward matrices
K_y	Feedback matrix
L	User selected specific gain
L_1, M_1, N_1	Plant acceleration contributions
$\mathbf{P}$	Covariance matrix
Q, R	Hermitian matrices
$U_{p,\,q,\,r}$	Pseudo control acceleration commands
$U_{p,\,q,\,r_{ad}}$	Control augmentation commands (NN outputs)
$U_{p,\,q,\,r\_error}$	Proportional and integral acceleration errors
W	Sigma-Pi NN weights
$\boldsymbol{W1}$, $\boldsymbol{W2}$	RPLR weights matrices
$\boldsymbol{e}$	Error between reference and plant model
p	Roll angular rate
$\dot{p}$	Roll angular acceleration
q	Pitch angular rate
$\dot{q}$	Pitch angular acceleration
r	Yaw angular rate
$\dot{r}$	Yaw angular acceleration
$\boldsymbol{u}$	Input vector
$\boldsymbol{x}$	State vector
$\boldsymbol{y}$	Output vector
$\boldsymbol{z}$	Command model state vector
α_{min}, α_{max}	Min/Max eigenvalues of Covariance matrix
δ_a	Aileron deflection
δ_{dir}	Pedals deflection
δ_{lat}	Lateral stick deflection
δ_{lon}	Longitudinal stick deflection
δ_r	Rudder deflection
δ_s	Stabilator deflection
ϕ	Regressor vector

λ	Forgetting factor
Ψ	Gradient matrix
Θ	Weight vector

Subscripts

a	aileron
c	inverse model (controller)
com	command
fwr	forward
inv	inverse
lin	linear
m	forward model
p, q, r	angular rates
ref	reference
r	rudder
s	stabilator

Superscripts

$\wedge$	forward NN output
*	reference trajectory
$\sim$	deviations

1. Introduction

In recent years, adaptive control systems have been taken advantages of artificial intelligence techniques[1,2,3,4,5] and high performance on-line system identification algorithms[6,7] using especially neural networks (NN). These new methods provide alternatives to gain scheduling approaches as well as to handle a variety of primary control surface failures. In order to pursue these features, the main objective of the NASA Intelligent Flight Control System (IFCS) F-15 program[8] is to investigate and develop through flight tests innovative and promising control approaches. The purpose of this paper is to show a comparison among three different fault tolerant control law schemes within the NASA IFCS F-15 program using the WVU IFCS F-15 Simulator developed at WVU.

The first approach is based on an adaptive flight controller using a Non Linear Dynamic Inversion (NLDI) augmented with a neural network to compensate inversion errors and changes in aircraft dynamics due to damage or failures on primary control surfaces. Moreover, this scheme uses an additional pre-trained neural network (PTNN) providing updated values of the aerodynamic and stability derivatives, required by the dynamic inversion, within the whole flight envelope.

The second approach combines conventional robust control schemes with neural networks to improve the performance of the flight control system. This robust controller is based on the Stochastic Optimal Feedforward and Feedback Technique (SOFFT), which belongs to the class of State Feedback Linear Quadratic Optimal Control approaches. In addition, the SOFFT controller is augmented with a neural network to compensate errors due to system uncertainties and/or failures on primary control surfaces.

The third approach takes advantages of the NN capabilities in performing system identification. Precisely, it features two adaptive neural entities which identify the forward and the inverse F-15 model and are connected according to the *predictor-corrector* scheme.

These three schemes are used to implement a Multi-Input-Multi-Output (MIMO) Control Augmentation System (CAS) with the tracking task of the same reference model in order to directly compare their features. The performance are evaluated in terms of trajectory tracking error and control activity for simple maneuvers at different flight conditions and with failures involving different primary control surfaces.

2. Aircraft Model and Simulation Environment

A simulation environment based on a nonlinear approximate model of the F-15 aircraft has been developed by the WVU researchers[9]. This model is derived from a Fortran code of a high performance military aircraft distributed by NASA to academic institutions within the 1990 AIAA GNC Design Challenge[10]. The aerodynamic and thrust characteristics are provided through 42 look-up tables, that is 16 tables for the longitudinal dynamics as functions of Mach number, angle of attack and stabilator deflections; 20 tables for the lateral-directional dynamics as functions of Mach number, angle of attack, sideslip angle and rudder; 2 tables for engine thrust and fuel flow as functions of Mach number and altitude. Additional look-up tables have been added for the modeling of the canard surfaces on the IFCS F-15 aircraft. The look-up tables have been subdivided to isolate the contribution of individual control surfaces in order to be able to simulate control surface failures[11].

A failure modeling strategy has been developed and applied for longitudinal, lateral and directional control surface blockage and partial destruction[4,11]. The method is based on the assumption that when a control device failure occurs, there is an alteration of the aerodynamic forces and moments, which is equivalent to a net loss of "aerodynamic efficiency". The contribution of each individual control device to the total external forces and moments is isolated and expressed in terms of a single parameter which can be varied during the simulation.

The simulation package is based on the *Flight Dynamics and Control* (FDC) toolbox[12] within Matlab/Simulink environment. For graphic display and

pilot interaction the dynamic model is interfaced with the *Aviator Visual Design Simulator* (AVDS) simulation package[13]. Particularly, the aircraft dynamic model is flown through a joystick device; however, pre-loading of command histories is also possible. In the open loop mode, the inputs given through the joystick are supplied directly to the stabilators, ailerons and rudders actuators, while the collective canard deflection is scheduled as a function of the Mach number and the angle of attack.

3. Controllers Description

3.1 Method #1: Non Linear Dynamic Inversion (NLDI) Scheme

The general scheme of this adaptive neural controller is shown in Figure 1. An on-line neural network algorithm (*Sigma-Pi*) is used to cancel the errors associated with the dynamic inversion of the model[1,2,3] and provide consistent handling qualities without requiring computational effort in gain-scheduling. In addition, a pre-trained neural network (PTNN) is employed to supply updated values of the aerodynamic and stability derivatives required by the dynamic inversion calculations, as the aircraft moves throughout its flight envelope. Finally, desired handling qualities are achieved with *ad hoc* reference models.

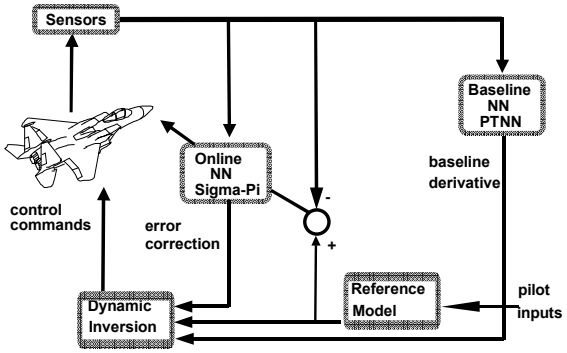

Figure 1. General Block Diagram of the *NLDI* scheme

Flight commands are generated by the pilot through longitudinal/lateral stick and pedals ($\delta_{lon_{stick}}$, $\delta_{lat_{stick}}$, $\delta_{dir_{pedal}}$), then these displacement commands are converted[5,14] into corresponding roll, pitch and yaw rate commands (p_{com}, q_{com}, r_{com}). The reference model provides filtered rate commands (p_{ref}, q_{ref}, r_{ref}) and acceleration commands ($\dot{p}_{ref}$, $\dot{q}_{ref}$, $\dot{r}_{ref}$) using first order roll rate and second order pitch and yaw rate transfer functions.

The inputs to dynamic inversion ($\dot{p}_c$, $\dot{q}_c$, $\dot{r}_c$) are computed using the expression:

$$\begin{bmatrix} \dot{p}_c \\ \dot{q}_c \\ \dot{r}_c \end{bmatrix} = \begin{bmatrix} U_p \\ U_q \\ U_r \end{bmatrix} - \begin{bmatrix} U_{p_{ad}} \\ U_{q_{ad}} \\ U_{r_{ad}} \end{bmatrix} \tag{1}$$

where ($U_{p_{ad}}$, $U_{q_{ad}}$, $U_{r_{ad}}$) are augmentation commands generated by adaptive NNs in order to compensate for the estimated errors e_p, e_q, e_r from the difference between reference and plant angular rates. Furthermore, the pseudo control acceleration commands (U_p, U_q, U_r) are computed using the following expressions:

$$U_x = \left(K_{p_x} + \frac{K_{i_x}}{s} \right) \cdot e_x + s \cdot x_{ref} \quad \text{with } x = p, q, r \tag{2}$$

where K_p and K_i are proportional and integrative constants.

The dynamic inversion is used to determine the necessary control surface deflections ($\delta_a, \delta_s, \delta_r$). These values will be reallocated to obtain actual collective and differential stabilator, differential canard, aileron and rudder deflections. Initially, control surface commands ($\delta_{a_{com}}$, $\delta_{s_{com}}$, $\delta_{r_{com}}$) are obtained with the following equation:

$$\begin{bmatrix} \delta_{a_{com}} \\ \delta_{s_{com}} \\ \delta_{r_{com}} \end{bmatrix} = B^{-1} \begin{bmatrix} \dot{p}_c - L_1 \\ \dot{q}_c - M_1 \\ \dot{r}_c - N_1 \end{bmatrix} \tag{3}$$

where *B* is the state-space system control matrix and the terms ($\dot{p}_c - L_1$, $\dot{q}_c - M_1$, $\dot{r}_c - N_1$) are the differences between input acceleration commands and actual plant acceleration contributions (L_1, M_1, N_1). These plant contributions are function of inertial and geometric characteristics, aerodynamic derivatives, angular rates and aerodynamic angles. Finally, the control surface deflections are computed from $\delta_{a_{com}}$, $\delta_{s_{com}}$ and $\delta_{r_{com}}$ in order to consider the modeling of the actuator dynamics (first and second order transfer functions).

Inversion errors and changes in aircraft dynamics are compensated through a neural algorithm based on a two-layer *Sigma-Pi* NN[5,14,15] for each angular acceleration ($\dot{p}, \dot{q}, \dot{r}$). These NNs use proportional and integral acceleration errors (U_{p_error}, U_{q_error},

U_{r_error}) for on-line learning purpose. Inputs to the networks are pseudo control acceleration commands (U_p , U_q , U_r) described above, bias terms and sensor feedback. For each channel three terms C_1, C_2 and C_3 are computed as functions of input variables and previous-step network outputs ($U_{p_{ad}}$, $U_{q_{ad}}$, $U_{r_{ad}}$). All the neuron outputs are summed and multiplied to each other - hence the name of the network. The outputs of the neural networks are the control augmentation commands defined as:

$$U_{ad} = W^T f\left(C_1, C_2, C_3\right) \tag{4}$$

where f is computed from each signal inputs using a nested Kronecker product and the network weights W are determined by an adaptation law:

$$\dot{W} = -G\left(U_{error} \cdot f + L\left|U_{error}\right| W\right) \tag{5}$$

where G and L are user selected specific gains.

In addition, the Pseudo-Control Hedging (PCH)[1,2,3] has been added into the control system to prevent or relieve control saturation problems occurring when an actuator is commanded beyond its limit. The differences between control surface commands and control surface deflections are used to evaluate angular acceleration corrections that are then fed to the reference model in order to modify reference angular rates and reduce the possibility of control saturation.

3.2 Method #2: SOFFT Controller

The second approach is based on the so-called Stochastic Optimal Feedforward and Feedback Technique (SOFFT) descending from optimal control techniques and in particular from Explicit Model Following Control (EMFC)[16]. In addition, the robust SOFFT controller is augmented with a neural network to compensate errors due to system uncertainties, unknown disturbances, inexact inversion of the plant dynamics and failures on primary control surfaces.

Differently from EMFC, the SOFFT approach[8,17,18] decouples the feedforward and feedback control design process avoiding conflicting demands on the control laws. In other words, there is not just the optimization of a single criterion in which the performance of the feedforward and feedback control laws are jointly evaluated with some compromises but a separate optimization of the feedforward and feedback control objectives is carried out.

The SOFFT control scheme can be considered both as an Adaptive Control System and a Gain Scheduling Approach since as the aircraft moves throughout the flight envelope, the matrices describing its linearized model are continuously updated and Algebraic Riccati Equations (ARE) are solved to yield the most updated control gains. The general structure of SOFFT controller is shown in Figure 2 with one feedback matrix (K_y) and three feedforward matrices (K_u, K_x, K_z) as result of the design process.

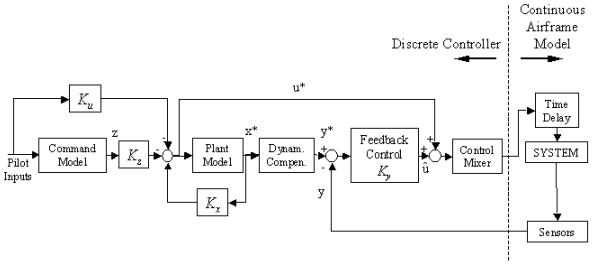

Figure 2. *SOFFT* Controller Scheme

A peculiar characteristic of this approach is the fact that, for the design of the feedforward control laws, two models are used: a *command* model and a *reference* plant model. The reference plant model is then forced to follow the signals coming from the command model by designing a control law that minimizes the error between the dynamics of the two models. There are two different methods in order to compute the feedforward gains K_u, K_x, K_z : *EMFC* and *Perfect Tracking*. According to the *EMFC* approach:

$$\tilde{u} = \min \int_0^\infty \left(\left[H_x \tilde{x} - H_z \tilde{z}\right]^T Q \left[H_x \tilde{x} - H_z \tilde{z}\right] + \tilde{u}^T R \tilde{u} \right) dt \tag{6}$$

constrained to:

$$\begin{bmatrix} \dot{\tilde{x}} \\ \dot{\tilde{z}} \end{bmatrix} = \begin{bmatrix} A_x & 0 \\ 0 & A_z \end{bmatrix} \begin{bmatrix} \tilde{x} \\ \tilde{z} \end{bmatrix} + \begin{bmatrix} B_x & 0 \\ 0 & B_z \end{bmatrix} \begin{bmatrix} \tilde{u} \\ \tilde{u}_z \end{bmatrix} \tag{7}$$

where A_x B_x C_x D_x H_x is the reference plant model and A_z B_z C_z D_z H_z is the command model. The state and input variables represent deviations from a reference trajectory that can usually be a set point condition (i.e $H_x x^* = H_z z^*$):

$$\tilde{x} = x - x^*,\ \tilde{z} = z - z^*,\ \tilde{u} = u - u^*,\ \tilde{u}_z = u_z - u_z^* \tag{8}$$

where:

$$z^* = -A_z^{-1} B_z u_z^* \quad \text{and} \quad x^* = -A_x^{-1} B_x u^* \tag{9}$$

The feedforward part of the control law can be defined as:

$$u = K_u u_z^* - K_x x - K_z z \tag{10}$$

The three feedforward matrices K_u, K_x, K_z are computed by solving two ARE's and one Lyapunov equation. If this is to be done in real time, the computational cost can easily become prohibitive and another way of computing the feedforward gains can be carried out through the *Perfect Tracking* control method based on Dynamic Inversion control approaches[19]. Defining the error to be minimized as:

$$\overset{\Delta}{e} = H_x x - H_z z \tag{11}$$

the *Perfect Tracking* method consists in calculating the three feedforward matrices K_u, K_x, K_z by forcing the time derivative of the error to remain zero:

$$\begin{aligned} \dot{e} &= H_x \dot{x} - H_z \dot{z} = \\ & H_x A_x x + H_x B_x u - H_z A_z z - H_z B_z u_z = 0 \end{aligned} \tag{12}$$

If $H_x B_x$ is full column rank, the control law:

$$u = \left(H_x B_x\right)^{+} \left(-H_x A_x x + H_z A_z z - H_z B_z u_z\right) \tag{13}$$

is such that the error remains at zero and the three feedforward gains can be written as:

$$\begin{aligned} K_x &= +\left(H_x B_x\right)^{+} H_x A_x \\ K_z &= -\left(H_x B_x\right)^{+} H_z A_z \\ K_u &= -\left(H_x B_x\right)^{+} H_z B_z \end{aligned} \tag{14}$$

The price to pay for this gain in computational resources is that the perfect tracking approach, as any inverting control law, works only for minimum phase plants.

The design of the feedback control is performed independently from the feedforward control design by using a standard output feedback LQR synthesis on the feedforward plant model A, B, C, D. The real plant is forced to follow the dynamics of the reference plant model:

$$\tilde{u} = \min \int_0^{\infty} \left(\tilde{y}^T Q \tilde{y} + \tilde{u}^T R \tilde{u}\right) dt \tag{15}$$

constrained to:

$$\begin{bmatrix} \dot{\tilde{x}} \\ \tilde{y} \end{bmatrix} = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} \tilde{x} \\ \tilde{u} \end{bmatrix} \tag{16}$$

where the input, state and output variables are deviations from a reference trajectory u^*, x^*, y^* supplied by the reference plant model:

$$\tilde{x} = x - x^*, \quad \tilde{u} = u - u^*, \quad \tilde{y} = y - y^* \tag{17}$$

The feedback matrix K_y is obtained by solving the ARE involving the matrices A, B, C, D, Q and R.

The SOFFT scheme augmented through an on-line neural network provides further benefits of adaptation without requiring substantial modifications to the existing linear control architecture[20]. In this setting, the neural network extends the operational range of the aircraft beyond the capacity of the linear controller maintaining command tracking and stable flight conditions. This neural augmentation is accomplished through a single hidden layer NN trained on-line by the EMRAN (*Extended Minimal Resource Allocating Network*) algorithm[21]. The network output is composed of two vectors: the adaptive term which aims to compensate for the uncertainties and the robustifying term which ensures that the network weights remain bounded during training phase.

3.3 Method #3: Predictor-Corrector Control Strategy

This adaptive neural network controller takes advantage of the NN capabilities in performing system identification. The control activity is handled by two adaptive neural entities which identify the forward and the inverse F-15 model and are connected according to the *predictor-corrector* scheme[22, 23]. The identification of the forward dynamics of the plant is accomplished to estimate on-line the plant Jacobian, which is then used in the inverse model adaptation process to implement the back propagation through the model.

The control scheme is based on the *reference model direct inverse* scheme (also known as *predictor-corrector*) shown in Figure 3.

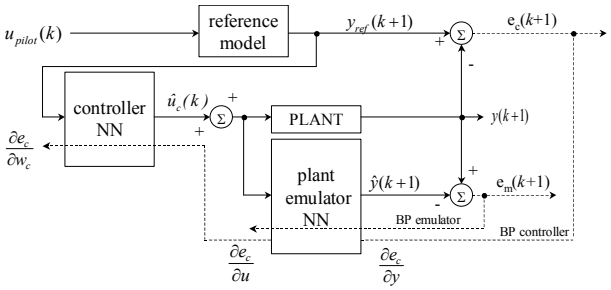

Figure 3. *Predictor-Corrector* scheme

Desired handling qualities are achieved through a reference model that is fed by pilot flight commands ($\delta_{lon_{stick}}$, $\delta_{lat_{stick}}$, $\delta_{dir_{pedal}}$) and provides filtered angular rate (p_{ref} , q_{ref} , r_{ref}) and angular acceleration commands ($\dot{p}_{ref}$, $\dot{q}_{ref}$, $\dot{r}_{ref}$) as describe in the subsection 3.1. These reference signals are processed by the neural controller, which calculates the required

control signals, reallocates them and feeds the actuators. In particular, the pitch control channel signal moves the collective stabilators, the roll channel commands the differential ailerons and the differential stabilators, the yaw control channel acts on the collective rudder and the differential canards. Collective canard deflections are scheduled as functions of Mach number and angle of attack.

As shown in Figure 3, the plant emulator represents the forward model while the controller action is carried out by the inverse model. The forward and the inverse models have both three input variables and three outputs as they identify the direct dynamic response of angular rates (*p, q, r*) to command inputs (δ_{lat} , δ_{lon} , δ_{dir}) and viceversa. The forward model is based on three Multi-Input-Single-Output (MISO) networks connected in a parallel structure, whereas the inverse model employs a SISO network for the longitudinal channel (*q*) and two MISO networks for the lateral and directional channels (*p*, *r*). Each neural network features a Multi Layer Perceptron (MLP) with a single hidden layer which implements the identification of NARX (Neural Auto Regressive with external inputs) systems according to the following scheme:

$$\begin{aligned} \mathbf{y} &= f_{NNfwr}(\phi_{fwr}) \\ \mathbf{u} &= f_{NNinv}(\phi_{inv}) \end{aligned} \tag{18}$$

where ϕ is the regressor vector whose structure is shown in detail in Table 1 for both forward and inverse model. The blocks pointed out by the dashed lines are repeated in sequence depending on the number of inputs of the neural system.

ϕ_{fwr}	ϕ_{inv}
y(*k*)	$y_{i\,ref}(k+1)$
⋮	$y_i(k)$
y(*k-n+1*)	⋮
u_i (*k-1*)	y_i (*k-n+1*)
⋮	u(*k-1*)
u_i (*k-n*)	⋮
	u(*k-n+1*)

Table 1. Regressor vector structure

In this implementation the input vector of the inverse NN is independent from the calculated output, that is the inverse model has no direct feedback. Avoiding this direct feedback of the NN output decreases the risk of oscillations during transient phases and allows greater time steps with a more real simulation time. The input signals *u(k-1), u(k-2), ..., u(k-n+1)* (with *n* = network order) are provided by a linear inverse model which feeds the neural entity with an estimation of the plant input at the previous time steps ($\delta_{a_{com}}$, $\delta_{s_{com}}$, $\delta_{r_{com}}$)$_{lin}$ according to the Eq. (3). Successively the NN inverse model can filter the signals and compensate for nonlinearities, modeling errors, model uncertainties and changes in dynamics due to failures and non-nominal flight conditions.

Initially, the forward and the inverse models are pre-trained using the back-propagation technique featuring the Levenberg-Marquardt method[24]. In addition the two models are trained on-line in order to achieve adaptive and fault tolerant characteristics. The error functions which are minimized for the forward and the inverse model on-line training are respectively:

$$\boldsymbol{E}_m = \tfrac{1}{2}\left(\boldsymbol{y}-\hat{\boldsymbol{y}}\right)^T \boldsymbol{K}_{p_m}\left(\boldsymbol{y}-\hat{\boldsymbol{y}}\right) \tag{19}$$

$$\boldsymbol{E}_c = \tfrac{1}{2}\left(\boldsymbol{y}_{ref}-\boldsymbol{y}\right)^T \boldsymbol{K}_{p_c}\left(\boldsymbol{y}_{ref}-\boldsymbol{y}\right) \tag{20}$$

The on-line training algorithm belongs to the Recursive Identification methods category and it is an extension of the *Recursive Pseudolinear Regression* (RPLR) algorithm[25]. This technique is based on the step by step updating of the $\boldsymbol{\Theta}_{fwr}$ and $\boldsymbol{\Theta}_{inv}$ vectors, which group in vector shape respectively the couples of weight matrices $\boldsymbol{W1}_{fwr}$, $\boldsymbol{W2}_{fwr}$ and $\boldsymbol{W1}_{inv}$, $\boldsymbol{W2}_{inv}$. Omitting subscripts for simplicity, the equations below represent the k^{th} time step of the RPLR algorithm:

$$\begin{aligned} \mathbf{K}(k) &= \mathbf{P}(k-1)\boldsymbol{\Psi}(k)\left[\lambda \boldsymbol{I} + \boldsymbol{K}_p \boldsymbol{\Psi}^T(k)\mathbf{P}(k-1)\boldsymbol{\Psi}(k)\right]^{-1} \\ \boldsymbol{\Theta}(k) &= \boldsymbol{\Theta}(k-1) + \mathbf{K}(k)\boldsymbol{K}_p \boldsymbol{e}(k) \\ \bar{\mathbf{P}}(k) &= \frac{1}{\lambda}\left[\boldsymbol{I} - \mathbf{K}(k)\boldsymbol{K}_p \boldsymbol{\Psi}^T(k)\right]\mathbf{P}(k-1) \\ \mathbf{P}(k) &= \frac{\alpha_{max}-\alpha_{min}}{tr\left(\bar{\mathbf{P}}(k)\right)}\bar{\mathbf{P}}(k) + \alpha_{min}\boldsymbol{I} \end{aligned} \tag{21}$$

applied according to the *constant-trace* technique. The $\boldsymbol{\Psi}$ matrix determines the gradient descent direction; for the forward model it is simply:

$$\boldsymbol{\Psi}_{fwr_m}(k) = -\nabla_{\boldsymbol{\Theta}_{fwr_m}} \boldsymbol{e}_{\hat{y}_m}(k) = \nabla_{\boldsymbol{\Theta}_{fwr_m}} \hat{\boldsymbol{y}}_m(k) \tag{22}$$

with m = 1,...,3, while for the inverse model it is:

$$\begin{aligned} \boldsymbol{\Psi}_{inv_m}(k) &= -\nabla_{\boldsymbol{\Theta}_{inv_m}} \boldsymbol{e}_{c_m}(k) \cong \nabla_{\boldsymbol{\Theta}_{inv_m}} \hat{\boldsymbol{y}}_m(k) = \\ &= \nabla_{\boldsymbol{u}_{m(k-1)}} \hat{\boldsymbol{y}}_m(k) \cdot \left(\nabla_{\boldsymbol{\Theta}_{inv_m}} \boldsymbol{u}_m(k-1)\right) \end{aligned} \tag{23}$$

with m = 1,...,3 and where the generic (*j,i*) element can be written as follows:

$$\boldsymbol{\Psi}_{inv_m}(j,i) = \frac{\partial \hat{\boldsymbol{y}}(i)}{\partial \boldsymbol{\Theta}_{inv_m}(j)} = \frac{\partial \hat{\boldsymbol{y}}(i)}{\partial \boldsymbol{u}_m}\frac{\partial \boldsymbol{u}_m}{\partial \boldsymbol{\Theta}_{inv_m}(j)} \tag{24}$$

4. Results of the Comparative Study

The performance comparison of the neural controllers has been carried out through three different maneuvers of 200 seconds each involving nominal flight conditions as well as stabilator and canard failures. Particularly, each maneuver consists of three phases:
- starting from a steady level flight at *6000 m* and Mach = *0.75*, there are three short doublets on the roll, pitch and yaw channels, respectively;
- after 45 seconds there is a descent to about *1500 m* and Mach = *0.5* which must be over within *80 s*;
- after the trim into the new flight condition, the set of doublets on the three channels are repeated as described above.

The first maneuver (*Man #1*) has been conducted in nominal flight conditions (no failures), whereas the second and the third maneuvers (*Man #2* and *Man #3*) present a failure respectively on the right stabilator (jammed at – *5 deg*.) and on the right canard (jammed at – *5 deg*.), both occurring in the first phase after *30 s*.

The performance of the different approaches have been illustrated by comparing the following parameters:
- *tracking performance*, in terms of mean value, maximum value and standard deviation of the errors between reference model angular rates and actual aircraft angular rates;
- *control activity* (ICA), in terms of the integral of the absolute value of the primary control surface deflections (collective and differential stabilators, differential ailerons, collective and differential canards, rudders).

Results listed in Table 2 and Table 3 show the performance of the three methods for *Man #1* in non failure condition; Table 4 and Table 5 present the tracking error and the ICA for *Man #2* with right stabilator failure while Table 6 and Table 7 concern *Man #3,* with right canard failure. The best results among the three methods are pointed out by grey-colored cells in the tables mentioned above. Figures 4, 6 and 8 show the comparison between Method #1 and Method #3 in terms of tracking error for the three channels (roll, pitch and yaw); whereas Figures 5, 7 and 9 represent the reference and the actual signal of angular rates for Method #2. Figures from 10 to 13 show the deflections of the control surfaces exposed to failure (stabilators and canards) for the three methods in non failure (*Man #1*) and failure condition (*Man #2* and *Man #3*).

Man #1 has been accomplished with the same pre-recorded time history of command inputs for all the three methods: this maneuver presents a substantial change of the operative flight condition and consequently the state variables vary within a wide portion of the flight envelope. Table 2 and Table 3 show that both Method #1 and Method #3 provide valuable and comparable performance in terms of tracking error (Figure 4), adaptation and control activity (Figures 10 and 12) but Method #3 requires more computational efforts with higher simulation time (about up to 14 times). On the other hand, Method #2 achieves poor results especially in terms of tracking error on the roll channel as it can be seen in Table 2 and Figure 5. *Man #2* and *Man #3* have been performed with the pilot-in-the-loop and show up the combined effects of the failure and of the operative condition change (in terms of altitude and ground speed). The relative results presented in Table 4 to Table 7 highlight that tracking errors slightly increase, in particular for the pitch channel in *Man #2* and for the roll channel in *Man #3*, as failures affect the aerodynamic surfaces that contribute, respectively, to the pitch and roll angular rate. In addition the control activity increases for all the control surfaces, apart from the surface directly affected by the failure: this is due to the asymmetry of the failure which may introduce dynamic coupling effects and alteration of both force and moment setting up.

In failure conditions, it can be noticed that Method #1 is more adaptive and achieves better results than Method #2 both in terms of tracking errors (Figures 7 and 9) and control activity (Figures 11 and 13). Furthermore, Method #1 provides even better performance than Method #3 in terms of tracking errors, except for the roll channel (Figures 6 and 8), and above all in terms of control activity (Figures 11 and 13). In fact, Method #3 with the RPLR algorithm requires a large computational effort because of the covariance matrices $\mathbf{P}$, whose dimensions grow with the square of the dimension of each NN. Obviously, higher simulation time leads the pilot to increase remarkably his/her control action through the joystick device with the drawback of an intensive control activity during the maneuvers performed with the pilot-in-the-loop. However, as shown in Figure 14 and Figure 15, Method #1 presents some oscillations on the control surface deflections that may damage the structure of the primary control surfaces.

5. Conclusions

Performance analisys of three adaptive neural control schemes has been carried out in different flight conditions and control surface failures. Results show the capability of the neural controllers of adapting to operative flight condition changes and of accomodating various damages with a very short transient. Both Method #1 and Method #3 performs remarkable results in all the considered maneuvers, while Method #2 achieves poor results. However, Method #3 requires a large amount of CPU and the simulation time is rather

high, therefore when the maneuvers are performed with the pilot-in-the-loop (*Man #2* and *Man #*3), the control activity increases significantly with respect to the other methods.

Acknowledgment

Support for this research effort has been provided through a grant from the Institute for Software Research (ISR).

References

1. Calise A. J., Lee S., Sharma M., "Direct Adaptive Reconfigurable Control of a Tailless Fighter Aircraft", *Proc. of the 1998 AIAA GNC Conference*, Boston, MA, August 1998, AIAA 98-4108.
2. Rysdyk R.T., Calise A. J., "Fault Tolerant Flight Control via Adaptive Neural Network Augmentation", *Proc. of the 1998 AIAA GNC Conference*, Boston, MA, August 1998.
3. Calise A. J., Sharma M., "Adaptive Autopilot Design for Guided Munitions", *AIAA Journal of Guidance, Control and Dynamics*, vol. 23, no 5*,* 2000, pp. 837-843.
4. Napolitano M.R., Younghawn A., Seanor B., "A Fault Tolerant Flight Control System for Sensor and Actuator Failures using Neural Networks", *Aircraft Design*, Elsevier Science Ltd, vol.3, no 2, 2000.
5. Kaneshige J., Gundy-Burlet K., "Integrated Neural Flight and Propulsion Control System", *Proc. of the 2001 AIAA GNC Conference*, Montreal, Canada, August 2001, AIAA Paper 01-4386.
6. Napolitano M. R., Song Y., Seanor B., "On-Line Parameter Estimation for Restructurable Flight Control Systems", *Aircraft Design*, Elsevier Science Ltd, no 4, 2001, pp. 19-50.
7. Song Y., Campa G., Napolitano M. R., Seanor B. Perhinschi M. G., " Online Parameter Estimation Techniques Comparison Within a Fault Tolerant Flight Control System", *AIAA Journal of Guidance, Control and Dynamics*, vol. 25, no 3, 2002, pp. 528-537.
8. Annon., "Intelligent Flight Control: Advanced Concept Program – Final Report", The Boeing Company, BOEING-STL 99P0040, May 1999.
9. Perhinschi M.G., Campa G., Napolitano M.R., Lando M., Massotti L., Fravolini M.L., "A Simulation Tool for On-line Real Time Parameter Identification", *Proc. of the 2002 AIAA MST Conference*, Monterey, CA, August 2002.
10. Brumbaugh, R.W. "An Aircraft Model for the AIAA Controls Design Challenge", NASA CR 186019, Dec. 1991.
11. Perhinschi M.G., Campa G., Napolitano M.R., Lando M., Massotti L., Fravolini M.L., "Modeling and Simulation of Failures for Primary Control Surfaces", *Proc. of the 2002 AIAA MST Conference*, Monterey, CA, August 2002.
12. Rauw M.O., "*FDC 1.2: A Simulink Toolbox for Flight Dynamics and Control Analysis*", Delft University of Technology, The Netherlands, 1998.
13. "*Aviator Visual Design Simulator (AVDS) – User Manual*", Rassmussen Simulation Techn., Ltd., Oct. 2000.
14. Kaneshige J., Bull J., Totah J.J., "Generic Neural Flight Control and Autopilot System", *Proc. of the 2000 AIAA GNC Conference*, Denver, CO, August 2000, AIAA Paper 00-4281.
15. Shin Y., Ghosh J., "The Pi-Sigma Network: An Efficient Higher-Order Neural Network for Pattern Classification and Function Approximation", *1991 IJCNN Seattle International Joint Conference on Neural Networks,* July 1991, IEEE 7803-0164.
16. Stengel R.F., *"Optimal Control and Estimation"* Dover Publication Inc. New York, 1994.
17. Halyo N., Direskeneli H., Taylor B., *"A Stochastic Optimal Feedforward and Feedback Control Methodology for Superagility"*, NASA CR 4471, November 1992.
18. Ostroff A.J., Proffitt M.S., *"Design and Evaluation of a Stochastic Optimal Feedforward and Feedback Technology (SOFFT) Flight Control Architecture"*, NASA TP 3419, June 1994.
19. Isidori A., *"Nonlinear Control Systems"*, Springer-Verlag, London, Third Edition, 1995.
20. Campa G., Sharma M., Calise A.J., Innocenti M., "Neural Network Augmentation of Linear Controllers With Application to Underwater Vehicles", *Proc. of the 2000 American Control Conference*, San Diego, CA, June 2000.
21. Campa G., Fravolini M.L., Napolitano M.R., "Extended Minimal Resource Allocating Neural Networks For Aircraft SFDIA", *Proc. of the 2001 ANNIE Smart Engineering System Design Conference*, Rolla, MO, November 2001.
22. Gili P., Battipede M., "An Adaptive Neurocontroller for a Nonlinear Combat Aircraft Model", *Journal of Guidance, Control, and Dynamics*, vol. 24, no. 5, 2001, pp. 910-917.
23. Gili P., Battipede M., "A MIMO Neural Adaptive Autopilot for a Nonlinear Helicopter Model", *Proc. of the 1999 AIAA GNC Conference*, Portland, OR, August 1999, AIAA 99-4219.
24. Norgaard M., "Neural Network Based System Identification Toolbox", *DTU-TR-97-E-851*, Department of Automation, Technical University of Denmark, Lyngby, June 1997, chap. 2, pp. 22-23.

25. Ljung L., "System Identification: Theory for the User", *Prentice-Hall Inc*., Englewood Cliffs, New Jersey, 1987, pp. 71-72.

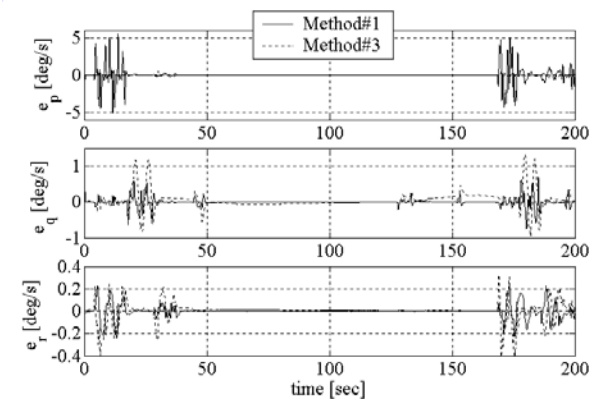

Figure 4. Tracking errors of Method #1 and Method #3 (*Man #1* = no failure)

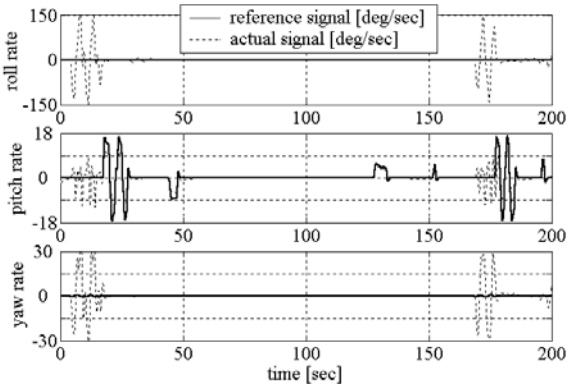

Figure 5. Reference and actual signals of Method #2 (*Man #1* = no failure)

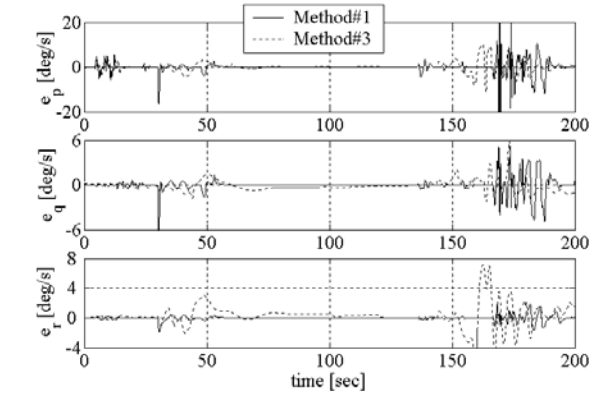

Figure 6. Tracking errors of Method #1 and Method #3 (*Man #2* = stabilator failure)

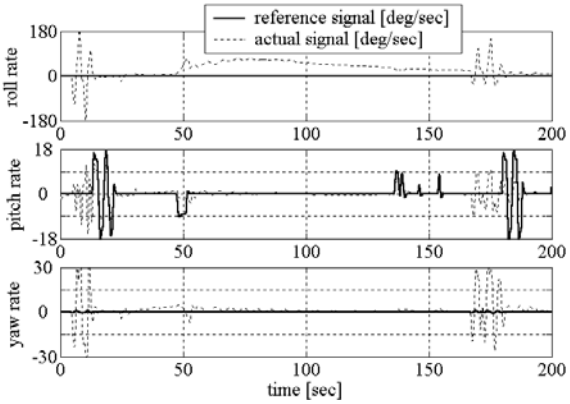

Figure 7. Reference and actual signals of Method #2 (*Man #2* = stabilator failure)

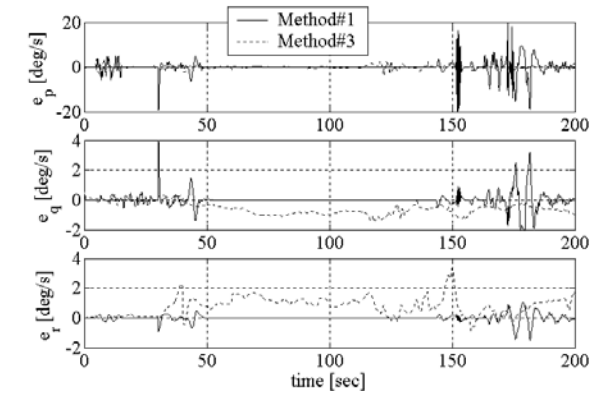

Figure 8. Tracking errors of Method #1 and Method #3 (*Man #3* = canard failure)

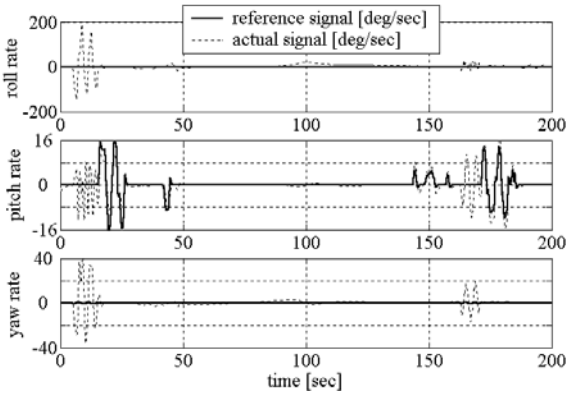

Figure 9. Reference and actual signals of Method #2 (*Man #3* = canard failure)

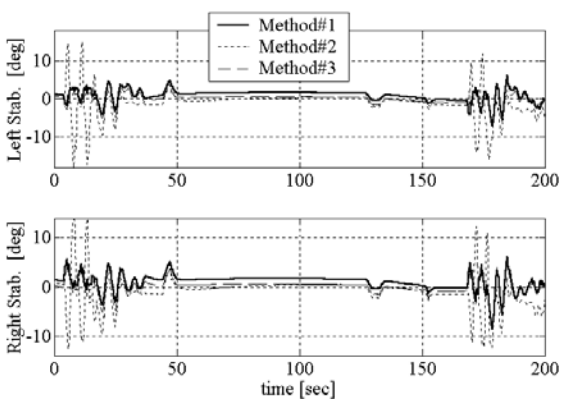

Figure 10. Left/Right Stabilator deflections (*Man #1* = no failure)

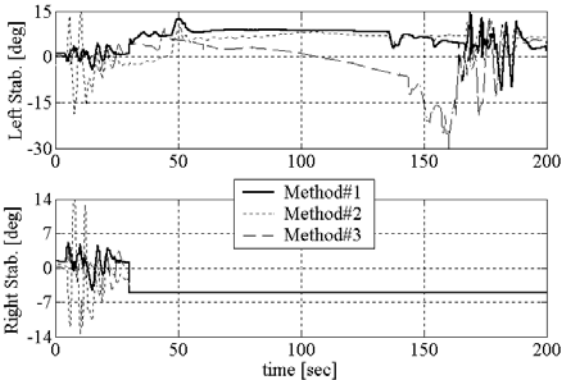

Figure 11. Left/Right Stabilator deflections (*Man #2* = stabilator failure)

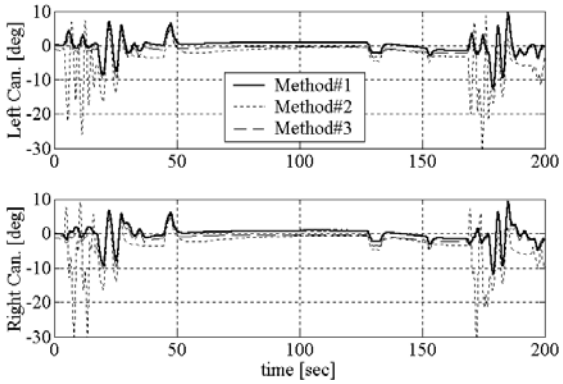

Figure 12. Left/Right Canard deflections (*Man #1* = no failure)

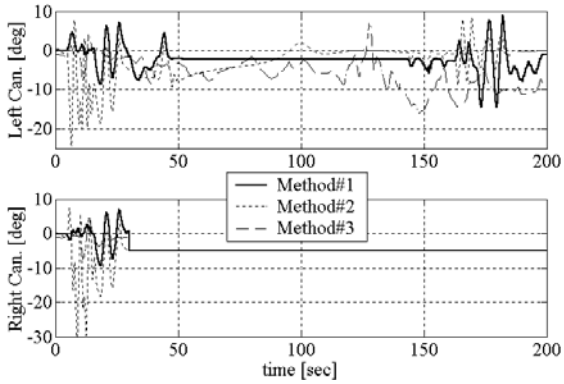

Figure 13. Left/Right Canard deflections (*Man #3* = canard failure)

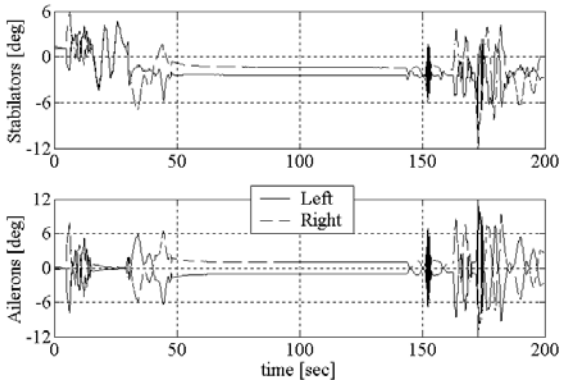

Figure 14. Method #1: stabilators and ailerons deflection (*Man #3* = canard failure)

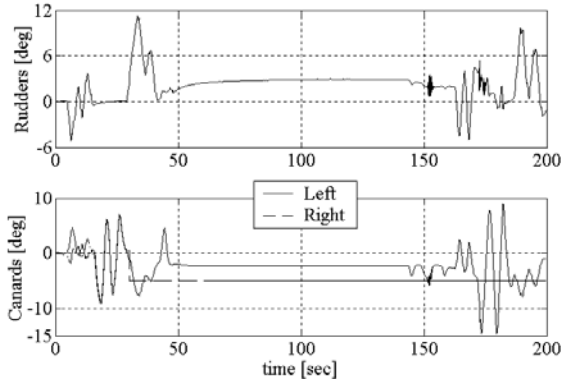

Figure 15. Method #1: rudders and canards deflection (*Man #3* = canard failure)

Tracking error	Mean error [rad/sec]			Std error [rad/sec]			Max error [rad/sec]		
	roll	*pitch*	*yaw*	*roll*	*pitch*	*yaw*	*roll*	*pitch*	*yaw*
Method #1	-3.218E-5	4.751E-5	-1.081E-5	1.429E-2	1.954E-3	8.971E-4	9.667E-2	1.320E-2	4.257E-3
Method #2	2.477E-3	1.248E-3	-7.739E-3	4.843E-1	4.116E-2	1.153E-1	3.195E+0	2.590E-1	6.733E-1
Method #3	1.496E-5	8.364E-4	1.570E-5	1.319E-3	4.235E-3	1.320E-3	9.349E-3	2.340E-2	8.067E-3

Table 2. Tracking performance (*Man #1* = no failure)

Integral Control Activity (ICA)	Stabilators		Ailerons	Rudders	Canards	
	collective	*differential*	*differential*	*collective*	*collective*	*differential*
Method #1	2.371E+00	1.209E+00	2.412E+00	1.117E+00	2.314E+00	1.211E+00
Method #2	4.857E+00	5.596E+00	6.791E+00	5.930E+00	1.224E+01	5.828E+00
Method #3	2.763E+00	1.904E+00	3.808E+00	2.123E+00	5.382E+00	1.806E+00

Table 3. NN Control Activity (*Man #1* = no failure)

Tracking error	Mean error [rad/sec]			Std error [rad/sec]			Max error [rad/sec]		
	roll	*pitch*	*yaw*	*roll*	*pitch*	*yaw*	*roll*	*pitch*	*yaw*
Method #1	-4.718E-4	-1.603E-4	-1.696E-4	4.560E-2	1.601E-2	5.615E-3	4.505E-1	1.131E-1	3.294E-2
Method #2	-5.204E-1	-1.328E-3	-2.255E-2	5.633E-1	5.717E-2	1.156E-1	3.210E+0	2.838E-1	6.518E-1
Method #3	1.551E-3	-1.784E-3	5.945E-3	3.912E-2	1.231E-2	2.559E-2	1.984E-1	1.061E-1	1.248E-1

Table 4. Tracking performance (*Man #2* = stabilator failure)

Integral Control Activity (ICA)	Stabilators		Ailerons	Rudders	Canards	
	collective	*differential*	*differential*	*collective*	*collective*	*differential*
Method #1	2.656E+00	1.814E+01	1.799E+01	3.742E+00	2.290E+00	3.397E+00
Method #2	5.284E+00	3.449E+01	2.636E+01	7.084E+00	9.056E+00	6.323E+00
Method #3	9.946E+00	2.235E+01	3.259E+01	1.445E+01	1.859E+01	1.540E+01

Table 5. NN Control Activity (*Man #2* = stabilator failure)

Tracking error	Mean error [rad/sec]			Std error [rad/sec]			Max error [rad/sec]		
	roll	*pitch*	*yaw*	*roll*	*pitch*	*yaw*	*roll*	*pitch*	*yaw*
Method #1	-1.928E-4	1.010E-4	-3.335E-5	4.867E-2	8.037E-3	3.980E-3	4.013E-1	6.986E-2	2.589E-2
Method #2	-7.961E-2	9.922E-4	-7.312E-3	3.811E-1	4.111E-2	1.024E-1	3.355E+0	2.194E-1	6.859E-1
Method #3	1.551E-3	-1.784E-3	5.945E-3	3.912E-2	1.231E-2	2.559E-2	1.984E-1	1.061E-1	1.248E-1

Table 6. Tracking performance (*Man #3* = canard failure)

Integral Control Activity (ICA)	Stabilators		Ailerons	Rudders	Canards	
	collective	*differential*	*differential*	*collective*	*collective*	*differential*
Method #1	3.466E+00	2.771E+00	5.534E+00	4.372E+00	6.267E+00	4.527E+00
Method #2	8.213E+00	1.050E+01	1.275E+01	8.005E+00	1.433E+01	1.327E+01
Method #3	9.946E+00	2.235E+01	3.259E+01	1.445E+01	1.859E+01	1.540E+01

Table 7. NN Control Activity (*Man #3* = canard failure)