ESANN'2003 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 23-25 April 2003, d-side publi., ISBN 2-930307-03-X, pp. 413-418

Post-Failure Analisys of an Adaptive Predictor-Corrector Neural Controller on a Flight Simulator

M.Battipede[+], P.Gili[*] , M. Lando[*], L. Massotti[*], M.R. Napolitano[#], G. Campa[#], M.G. Perhinschi[#]

+ Department of Mechanics , Politecnico di Torino, Torino 10129, Italy
* Department of Aeronautical and Space Engineering, Politecnico di Torino, Torino 10129, Italy
# Department of Mechanical and Aerospace Engineering, West Virginia University, Morgantown, WV 26506/6106, USA

Abstract. This paper is concerned with the comparison between a classical robust control system and a neural network controller based on the *predictor-corrector* control scheme featuring different neural network architectures and on-line training algorithms. Both the controllers have been applied to an adaptive flight control system for the F-15 WVU flight simulator and the results are given in terms of performance comparison and control activity evaluation.

1. Introduction

In the last decades artificial intelligence techniques, especially Neural Networks (NN's), have been successfully used for the design of adaptive control systems. In particular, aeronautical and space control applications are taking advantages of these intelligent flight controllers that provide consistent handling qualities without requiring the computational efforts of the gain-scheduling activity, typical of the classic control theory.

The aim of this paper is to show the enhancement of performance resulting from a neural control approach with respect to a sophisticated classical controller, based on the Stochastic Optimal Feedforward and Feedback Technique [1] (SOFFT), which belongs to the class of the State Feedback Linear Quadratic Optimal Control.

Both the neural and the non neural controller have been realized within the *NASA Intelligent Flight Control System* (IFCS) program [2], with the purpose of developing and investigating innovative flight control schemes, able to recover from primary control surface failures. The simulation environment consists of a 6 DOF nonlinear model of the F-15 high performance military aircraft implemented in Matlab/Simulink by the West Virginia University (WVU) researchers [3]. A primary control surface failure modeling is added to simulate failure scenarios.

The neural controller dealt with in this paper belongs to the adaptive *predictor-corrector* control strategy, based on the system identification theory. In particular it features two adaptive neural entities (*emulator* and *controller*) which identify the forward and the inverse F15 model [4]. The identification of the forward dynamics of the plant is accomplished to estimate on-line the plant Jacobian, which is used in the inverse model adaptation process to implement the back propagation through the

ESANN'2003 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 23-25 April 2003, d-side publi., ISBN 2-930307-03-X, pp. 413-418

model. In previous efforts this control architecture has been successfully adopted by the first two authors to address a Single-Input-Single-Output (SISO) neural adaptive normal acceleration limiter for a nonlinear F-16 combat aircraft model [4] and a Multi-Input-Multi-Output (MIMO) neural adaptive rate damping autopilot for a nonlinear H-106 helicopter model [5].

In this study the controller is used as a Control Augmentation System (CAS), with the reference model tracking task. The inverse neural entity of the *predictor-corrector* controller has been implemented through different NN architectures comparing various NARX (*Neural Auto Regressive with eXternal inputs*) systems and on-line training algorithms, such as *Recursive Pseudolinear Regression* (RPLR) and *Extended Back Propagation* (EBPA). The controller performance are evaluated in terms of trajectory tracking error and pilot workload in pre/post-failure conditions.

2. Simulation Environment and Control Strategy

2.1 Aircraft Model

Simulations are accomplished through a 6 DOF nonlinear mathematical model of the F-15 high performance military aircraft, that is derived from a Fortran code distributed by NASA to academic institutions within the 1990 AIAA GNC Design Challenge. This model is based on 42 look-up tables which are functions of flight variables and controls (Mach number, altitude, angle of attack, sideslip angle, stabilator and rudder deflection) and provide aerodynamic and thrust characteristics. The pilot inputs are given through 4 channels that act directly on stabilators, ailerons, rudders and throttle. A failure modeling strategy has been developed and applied for primary control surface blockage and/or partial destruction.

The F-15 flight simulator is developed in *Matlab/Simulink* environment and interfaced with the *Aviator Visual Design Simulator* (AVDS) simulation package for graphic display and pilot interaction. Particularly, the aircraft dynamic model is flown through a joystick device or on the basis of pre-recorded command histories.

2.2 Intelligent Flight Control System

The control activity is handled by two adaptive neural entities which identify the forward and the inverse F-15 models and are connected according to the *predictor-corrector* scheme as shown in Figure 1.

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 dynamics response of angular rates $\boldsymbol{y} = [p\ q\ r]^T$ to pilot input commands (δ_{lat} , δ_{lon} , δ_{dir}) and viceversa. Moreover, desired handling qualities are achieved through a *reference model* that provides filtered angular rate $\boldsymbol{y}_{ref} = [p_{ref}\ q_{ref}\ r_{ref}]^T$ and angular acceleration commands ($\dot{p}_{ref}$, $\dot{q}_{ref}$, $\dot{r}_{ref}$) receiving in input the pilot commands.The reference signals are processed by the neural controller, which computes the required control signals, reallocates them and feeds the actuators. In particular, the pitch control signal (q) moves the collective stabilators, the roll channel (p) commands the differential

ailerons and stabilators, the yaw control channel (*r*) acts on the collective rudders and the differential canards, whereas collective canard deflections are scheduled as a function of Mach number and angle-of-attack.

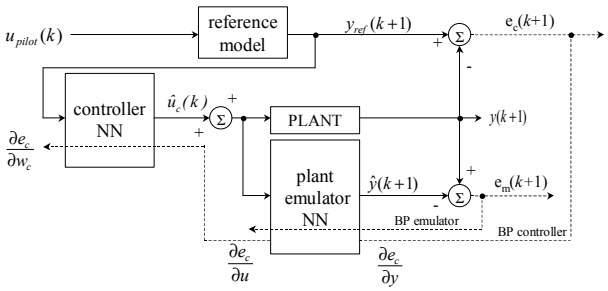

Figure 1 Predictor-Corrector scheme

3. NN Architecture

The forward model is based on three Multi-Input-Single-Output (MISO) networks connected in a parallel structure, whereas the inverse model features a SISO network for the longitudinal channel (*q*) and two MISO networks for the lateral and directional channels (*p, r*). This architecture has been selected as a result of a tradeoff between the requirement of suppling exhaustive information to each network and the necessity of minimizing the neural system. Each NN features a Multi Layer Perceptron (MLP) with a single hidden layer. The forward and the inverse models are initially trained off-line using the back-propagation technique featuring the Levenberg-Marquardt method [6].

3.1 NARX Structure

The identification through the NARX model is performed by each NN according to the following scheme:

$$\begin{aligned} \mathbf{y} &= f_{NNfwr}(\phi_{fwr}) \\ \mathbf{u} &= f_{NNinv}(\phi_{inv}) \end{aligned} \tag{2}$$

where ϕ is the regressor vector whose structure is shown in Table 1. The first column refers to the forward model regressor vector whereas the other columns describe both the full and the "reduced" inverse model regressor vector. *n* is the number of discrete time delays and represents the order of the NARX model. The blocks outlined by the dashed lines are repeated in sequence depending on the dimension of the input vector **u**, for the forward model, and of the output vector **y**, for the inverse model.

The input vector of the inverse NN is independent from the calculated output, meaning that there is no direct feedback of the NN output, to reduce the risk of oscillations during transient phases. According to Table 1, signals *u* (*k-1*), ... , *u* (*k-n+1*) that should be provided by the direct feedback are supplied by a linear inverse model, where aerodynamic and stability derivatives are updated by a Pre-Trained

ESANN'2003 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 23-25 April 2003, d-side publi., ISBN 2-930307-03-X, pp. 413-418

Neural Network (PTNN) within the whole flight envelope. Successively the NN inverse model filters the signals and compensates for nonlinearities, modeling errors, model uncertainties and changes in dynamics due to failures and/or non-nominal flight conditions.

ϕ_{fwr}	ϕ_{inv}	$\phi_{inv\ RED.}$
$y(k)$	$y_{i\ ref}(k+1)$	$y_{i\ ref}(k+1)$
$\vdots$	$y_i(k)$	$y_i(k)$
	$\vdots$	$\vdots$
$y(k-n+1)$	$y_i(k-n+1)$	$y_i(k-n+1)$
$u_i(k-1)$	$u(k-1)$	$u(k-1)$
$\vdots$	$\vdots$	$\vdots$
$u_i(k-n)$	$u(k-n+1)$	$u(k-n+1)$

Table 1. Regressor vector structure for each NN, with k = time step and n = descrete time delays.

3.2 Neural Network Training Algorithms

The forward and the inverse model 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:

$$\boldsymbol{E}_m = \tfrac{1}{2}\left(\boldsymbol{y}-\hat{\boldsymbol{y}}\right)^T \boldsymbol{K}_{p_m}\left(\boldsymbol{y}-\hat{\boldsymbol{y}}\right) \tag{3}$$

$$\boldsymbol{E}_c = \tfrac{1}{2}\left(\boldsymbol{y}_{ref}-\boldsymbol{y}\right)^T \boldsymbol{K}_{p_c}\left(\boldsymbol{y}_{ref}-\boldsymbol{y}\right) \tag{4}$$

where $\boldsymbol{K}_{p_m}$ and $\boldsymbol{K}_{p_c}$ are gain matrices and $\hat{\boldsymbol{y}} = [\ \hat{p}\ \hat{q}\ \hat{r}\ ]^T$ is the output vector of the forward model.

Two on-line training algorithms are used in this comparitive study. The first belongs to the Recursive Identification methods category and is basically an extension of the *Recursive Pseudolinear Regression* (RPLR) algorithm [7]. This technique is based on the step by step updating of the Θ_{fw} and Θ_{in} vectors, which group in vector shape respectively the couples of weight matrices $\boldsymbol{W}1_{fw}$, $\boldsymbol{W}2_{fw}$ and $\boldsymbol{W}1_{in}$, $\boldsymbol{W}2_{in}$. The RPLR algorithm is thoroughly described in [8], where the algorithm stability proof for MIMO systems is also provided.

The second method is a modified version of the *Back-Propagation* (BP) algorithm where the capability of each neuron of the hidden layer is enhanced by manipulating the standard sigmoid activation function (used for the RPLR) and adding further independent variables:

$$f(net,U,L,T) = \frac{U-L}{1+e^{-net/T}} + L \tag{5}$$

where *net* is the input to the activation function and U, L and T are independent variables. In the EBPA [9] the back-propagation algorithm it used not only to update the weights of the input and output matrices $\boldsymbol{W}1$and $\boldsymbol{W}2$, but also to update the parameters $U,L,T,$ that define the shape of each neuron. This expedient allows to

ESANN'2003 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 23-25 April 2003, d-side publi., ISBN 2-930307-03-X, pp. 413-418

avoid local minima which is one of the most critical problems of the standard BP algorithm. Moreover, solution is reached in fewer iterations thus the training process is sped up.

4. Results and Conclusions

Performance of the classical and neural controllers have been evaluated in pre and post failure conditions, in order to evaluate the NN controller capability of adapting at a sudden and critical dynamic modification. The test maneuver shown in Figure 2 has been accomplished through two sets of doublets on each control surface with a failure occurring at the left stabilator (jammed at –5 deg.) at 48 sec. The results presented in Table 2 feature the tracking performance, in terms of mean values, maximum values and standard deviations of the errors between the reference model and the aircraft angular rates. The results listed in Table 3 show the NN activity, in terms of the difference between the command signals (δ_{lat}, δ_{lon}, δ_{dir}) calculated by the linear inverse model and those actually provided by the NN controller.

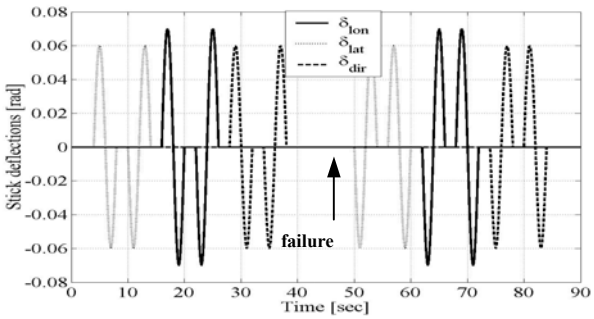

Figure 2. Test maneuver.

The comparison in post-failure conditions is between the SOFFT linear controller (Case #1) and four neural architectures: Case #2 (2^{nd} order for the forward and the inverse model with RPLR), Case #3 (2^{nd} order for the forward model and 4^{th} reduced order for the inverse model with RPLR), Case #4 (2^{nd} order for the forward model and 4^{th} full order for the inverse model with RPLR) and Case #5 (4^{nd} order for the forward model and 6^{th} order for the inverse model with EBPA). It can be noticed that performance are noteworthily improved by using neural networks because of their capability to handle nonlinear dynamic systems with the drawback of higher simulation times. Among the neural architectures, the 4^{th} order with RPLR (Case #4) achieves better results with respect to the other cases, however the simulation time raises remarkably up to 4 times in comparison with Case #2 and up to 8 times for Case #5. Using the EBPA algorithm for the on-line NN training, the NN activity can be compared with Case #4 but the tracking error values are definitely worse (both for low and high values of the learning rates). The reason of the poor EBPA performance can be attributed to a wrong application of the algorithm. In fact, the EBPA feature of fast convergence makes it very effective in static mapping or signal recognition functions, where high local precision is required. On the contrary, in dynamic sytem

identification the on-line training should converge trying to preserve a 'memory' of the previous solution, that is assuring global properties, in order not to diverge during transitory phases.

Tracking error	**Mean error [rad/sec]**			**Std error [rad/sec]**			**Max error [rad/sec]**		
	roll	*pitch*	*yaw*	*roll*	*pitch*	*yaw*	*roll*	*pitch*	*yaw*
Case #1	2.775E-1	-5.376E-2	1.429E-2	6.098E-1	1.506E-1	9.763E-2	3.295E+0	1.144E+0	6.591E-1
Case #2	1.098E-2	-2.073E-2	3.851E-3	3.623E-2	2.752E-2	1.460E-2	3.029E-1	8.523E-2	9.299E-2
Case #3	4.439E-3	-2.809E-3	5.442E-3	2.595E-2	8.109E-3	2.678E-2	2.523E-1	6.299E-2	2.790E-1
Case #4	5.120E-4	-3.263E-3	5.671E-3	7.017E-3	4.224E-3	1.066E-2	1.429E-1	5.252E-2	1.182E-1
Case #5	1.207E-1	-4.465E-2	6.847E-3	5.851E-1	8.233E-2	2.388E-2	2.842E+0	3.234E-1	1.268E-1

Table 2. Tracking performance.

NN Activity	**Mean value [rad]**			**Std value [rad]**			**Max value [rad]**		
	$\Delta\delta_{lon}$	$\Delta\delta_{lat}$	$\Delta\delta_{dir}$	$\Delta\delta_{lon}$	$\Delta\delta_{lat}$	$\Delta\delta_{dir}$	$\Delta\delta_{lon}$	$\Delta\delta_{lat}$	$\Delta\delta_{dir}$
Case #1	-	-	-	-	-	-	-	-	-
Case #2	3.416E-2	5.234E-2	-3.213E-2	5.033E-2	7.637E-2	4.493E-2	1.920E-1	2.661E-1	1.786E-1
Case #3	9.991E-3	1.477E-2	2.025E-3	5.725E-2	1.042E-1	6.437E-3	4.958E-1	9.364E-1	6.163E-2
Case #4	-2.036E-3	-3.802E-3	1.670E-3	6.169E-3	2.743E-2	3.978E-3	3.783E-2	5.462E-1	5.736E-2
Case #5	3.770E-3	-1.343E-2	4.006E-3	8.118E-3	4.493E-2	1.669E-2	2.876E-2	3.810E-1	8.475E-2

Table 3. NN activity.

References

[1] G.Campa, M.L.Fravolini, M.R.Napolitano, M.G.Perhinschi, M.Battipede, "A Stochastically Optimal Feedforward and Feedback Technique for Flight Control Systems of High Performance Aircrafts", *Accepted for presentation at the ACC 2003*, Denver, Co, USA, 4-6 June 2003.

[2] Annon., "Intelligent Flight Control: Advanced Concept Program – Final Report", The Boeing Company, BOEING-STL 99P0040, May 1999.

[3] M.G.Perhinschi, G.Campa, M.R.Napolitano, M.Lando, L.Massotti, M.L.Fravolini, "A Simulation Tool for On-line Real Time Parameter Identification", *Proc. of the AIAA MST Conference*, Monterey, Ca, August 2002.

[4] P.Gili, M.Battipede, "An Adaptive Neurocontroller for a Nonlinear Combat Aircraft Model", *Journal of GCD*, vol. 24, no. 5, Sept.-Oct. 2001, pp. 910-917.

[5] P.Gili, M.Battipede, "A MIMO Neural Adaptive Autopilot for a Nonlinear Helicopter Model", *Proc. of the AIAA GNC Conference*, Portland, OR, August 1999.

[6] M. Norgaard, "Neural Network Based System Identification Toolbox", *DTU-TR-97-E-851*, Dept. of Automation, Technical University of Denmark, Lyngby, 1997.

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

[8] M.Battipede, P.Gili, M.R.Napolitano, M.G.Perhinschi, L.Massotti, M.Lando, "Implementation of an Adaptive Predictor-Corrector Neural Controller within the NASA IFCS F-15 WVU Simulator", *Accepted for presentation at the ACC 2003*, Denver, Co, USA, 4-6 June 2003.

[9] M.R. Napolitano, C.I. Cheng, S. Naylor, "Aircraft Failure Detection and Identification Using Neural Networks", *Journal of GCD*, vol. 16, no. 6, Nov.-Dec. 1993, pp. 999-1009.