Military Technical College,
Kobry El-Kobbah,
Cairo, Egypt

9th International Conference
On Aerospace Sciences &
Aviation Technology

NEURAL PREDICTIVE SCHEME FOR AIRCRAFT LONGITUDINAL CONTROL

BAYOUMY*, A. M., BORDENEUVE-GUIBE‡, J.

ABSTRACT

Model Based Predictive Control (MBPC) is a well-known control technique especially in chemical industries and recently in aeronautics. Generalized Predictive Control (GPC) is an algorithm of MBPC family. Although existence of adaptive version of GPC, problems such as need for accurate modelling, linearization and online adaptation to system variations are still questions.

In this paper a new control method is proposed named Linear Neural Generalized Predictive Control (LNGPC). It uses a combination of Linear Neural Networks (LNN) and GPC. LNN is used as parametric identifier that makes both linearization and parameter extraction. An Algorithm is proposed for online learning of system Input/ Output bounds and making corresponding weight scaling. This ensures the stability of learning process even with variable system bounds. Online batch learning is used in LNN identification. An interpretation to the LNN weights is proposed to get system parameters as a discrete Transfer Function (TF). This given known order TF is then passed to a standard GPC controller.

Automatic Flight Control Systems (AFCS) often faces many problems such as unmodeled dynamics and fast parametric variations. LNGPC is tested with a realistic longitudinal rotorcraft model in a terrain-following application. The simulations show good results in terms of stability and adaptation comparing to other control schemes.

KEYWORDS

Predictive Control, Neural Networks, Aircraft, Flight Control

* Ph.D. student, Dpt. of Avionics &Systems, ENSICA, France, Email:amgad733@afmic.com
‡ Doctor, Dpt. of Avionics &Systems, ENSICA, France. Email: joel.bordeneuve@ensica.fr
Ecole Nationale Supérieur d'Ingénieur de construction aéronautique ENSICA, 1, pl. Emile Blouin, F-31056, Toulouse Cedex, France
http://www.ensica.fr

NOMENCLATURE

Abbreviations

ADALINE	ADaptive LINear Elements
AFCS	Aircraft Flight Control System
BP	Backpropagation Algorithm
GPC	Generalized Predictive Control
I/O	Input / Output
LNGPC	Linear Neural Generalized Predictive Control
LNN	Linear Neural Network
MBPC	Model Based Predictive Control
NN	Neural Network
TF	Transfer Function

Symbols

A, B	Discrete transfer function denominator and numerator as polynomial in q^{-1}
T	NN Target vector
G	System step response coefficients
I	NN input vector
N_1, N_2, N_u	Minimum, maximum and control cost horizons
N_B	NN Batch size (training-set length)
N_i, N_u, N_y	Length of NN input vector and system input and output delay line
O	NN output vector
q^{-1}, z^{-1}	Backward time shift operator
t	Time variable
u	System input, linear horizontal velocity
U_{max}	Maximum absolute input signal
W	NN Weight matrix
W_u	NN Weight system input sub-matrix
W_y	NN Weight system output sub-matrix
y	System output
Y/U	Maximum output maximum input ratio
Y_{max}	Maximum absolute output signal
na, nb	Degree of polynomial A, B
Δ	Finite difference operator
η	Learning rate
λ	Control energy weight factor
τ_e	Set-point filter time constant
ξ	Disturbance signal, white noise

Subscripts

j	Value after j time steps in the future
m	Model value

Superscripts

$\hat{}$	Predicted value
$\bar{}$	Normalised value

1. INTRODUCTION

Model Based Predictive Control (MBPC), or simply Predictive Control, is a family of algorithms with common strategy. MBPC appears in the decade of 1970s and have got a good reputation in the chemical industries and process control. [1]
The main strategy of the MBPC is as follows. A model of the controlled system is used to predict its behaviour in the future. A known required reference trajectory is given for some time steps in the future. Then an optimisation algorithm is used to find the optimum control sequence for certain number of steps in the future that minimise a certain cost function which includes future predicted errors and control increments. A receding horizon technique is then applied where only the first control signal of the optimum future control sequence is applied to the controlled system.

A milestone of MBPC was done by Clark *et. Al.* on 1987 with an algorithm called Generalized Predictive Control (GPC)[2,3]. GPC is known to control inverse unstable systems, open-loop unstable systems, and variable dead time. It is also robust with respect to modelling errors, over and under-parameterisation and sensor noise [4]. GPC has many variations, which are capable to deal with system constraints [5], multivariable systems [6,7] and to improve its stability [8-13].

Recently in aeronautics, GPC has been applied to terrain-following flight for a rotorcraft [14,15], in aircraft guidance [25], for active flutter suppression [16] and recently in control of an aeroelastic tiltrotor aircraft [17]

Artificial Neural Networks or simply Neural Nets (NN) are mathematical models of brain-like elements claiming to model it in order to benefit from its computational power. Linear NN are first proposed by Widrow and Hoff in 1960s with the name ADALINE [20,21] and have been used as the typical NN architecture till the creation of the Backpropagation algorithm (BP) which provides a tool for training nonlinear multi-layer precptron neural nets (MLP) in the late 1980s [23,24].

Although linear nets have been invented before nonlinear ones, they still have some attractive features, such as their ability to find best-fit linear representation of nonlinear mappings and its relatively fast linear computations. Multiple output version of Linear NN (MADALINE) exists also. [22]

Three possibilities exist for the utilisation of NN in a MBPC scheme:

- One is to perform identification using a nonlinear NN, like MLP trained with BP. Linearization of the identified NN model is to be done every time step to use it with a linear MBPC scheme, (e.g. GPC).
- Second is to use a nonlinear NN model with a nonlinear MBPC scheme using a nonlinear optimisation technique (e.g. [19])
- The third alternative that will be addressed in this paper is to use a Linear NN model, like ADALINE, with a linear optimisation MBPC scheme.

This scheme, although simple, seems to be a good alternative because of its fewer requirements in computations.

In this paper LNN are employed to perform batch discrete parametric identification and linearization. After this a linear GPC controller is used to control the nonlinear system using discrete TF found online by the LNN. Since the NN is normally working in the range [-1,1], another interesting feature proposed and used in this paper is the *AutoScaling* of the input/ target/ output signals by online learning of the system I/O bounds and corresponding scaling of NN weights.

This paper is divided into six sections with the introduction being the first one. The second section reviews the main idea of MBPC and the derivation of GPC control law. Linear neural nets and their application in online system identification are described in section three. The LNGPC control scheme is described in section four. An application of the proposed control scheme is performed on a rotorcraft in a terrain-following longitudinal flight and compared to another published work. Finally, the last section, section six, concludes the paper.

2. GENERALIZED PREDICTIVE CONTROL (GPC) ALGORITHM

The complete derivation of GPC is given in [1] but here is a brief review
Starting from CARIMA model

$$\Delta A\left(q^{-1}\right)y(t)=B\left(q^{-1}\right)\Delta u(t-1)+\xi(t) \quad \text{.....(1.)}$$

Where:

$$\begin{aligned} A\left(q^{-1}\right)&=1+a_1q^{-1}+\ldots+a_{n_a}q^{-na}\\ B\left(q^{-1}\right)&=b_0+b_1q^{-1}+\ldots+b_{n_b}q^{-nb} \quad \text{.....(2.)}\\ \Delta&=1-q^{-1} \end{aligned}$$

Using polynomial division (also called Diophantine equation) to divide 1 by ΔA
Note: ΔA is a polynomial with negative power
We get:

$$\frac{1}{\Delta A}=E_j+q^{-j}\frac{F_j}{\Delta A} \quad \text{.....(3.)}$$

Where E_j and F_j are polynomials calculated recursively
Substituting in (Eqn.1) and taking the prediction at time $(t+j)$

$$\hat{y}(t+j\,|\,t)=E_jB\Delta u(t+j-1)+F_jy(t) \quad \text{.....(4.)}$$

Note that the prediction of ξ is zero
As seen here, the benefit of the use of (Eqn.3) is to separate the future predicted output from the actual measured output till the current time t.

Performing the same on E_jB by 1,

$$E_jB=G_j+q^{-j}\Gamma_j \quad \text{.....(5.)}$$

Substitute we get,

$$\begin{aligned}\hat{y}(t+j\,|\,t) &= G_j\Delta u(t+j-1)+\Gamma_j\Delta u(t-1)+F_j y(t)\\ &= G_j\Delta u(t+j-1)+y_o(t+j)\end{aligned} \quad \text{.....(6.)}$$

Where y_o is the free response of the system (response at $\Delta u = \mathbf{0}$)

$$y_o = \Gamma_j\Delta u(t-1)+F_j y(t) \quad \text{.....(7.)}$$

Now defining the cost function J:

$$J = \sum_{j=N_1}^{N_2}\left[\hat{y}(t+j)-w(t+j)\right]^2+\lambda\sum_{j=1}^{N_u}\Delta u(t+j-1)^2 \quad \text{.....(8.)}$$

Subject to: $\Delta u(t+j-1)=0$ for $N_u < j < N_2$ where N_u is the control horizon.
Note: This is equivalent to put infinite weights on control increments after the time N_u

Then the prediction equation becomes in matrix form

$$\hat{\mathbf{y}} = \mathbf{G}_1\tilde{\mathbf{u}}+\mathbf{y}_o \quad \text{.....(9.)}$$

Where:

$$\mathbf{G}_1 = \begin{bmatrix} g_{N_1} & g_{N_1-1} & \cdots & g_0 & 0 & \cdots & 0\\ g_{N_1+1} & \ddots & & & \ddots & \ddots & \vdots\\ \vdots & & \ddots & & & g_0 & 0\\ \vdots & & & \ddots & & & g_0\\ g_{N_2} & g_{N_2-1} & & & \ddots & & g_{N_2-N_u+1}\end{bmatrix}_{[N_2-N_1+1\times N_u]} \quad \text{.....(10.)}$$

$$\begin{aligned}\hat{\mathbf{y}} &= \left[\hat{y}(t+N_1)\quad \hat{y}(t+2)\quad \hat{y}(t+3)\quad \cdots\quad \hat{y}(t+N_2)\right]^T\\ \tilde{\mathbf{u}} &= \left[\Delta u(t)\quad \Delta u(t+1)\quad \cdots\quad \Delta u(t+N_u-1)\right]^T\\ \mathbf{y}_o &= \left[y_o(t+N_1)\quad \cdots\quad y_o(t+N_2)\right]^T\\ \mathbf{w} &= \left[w(t+N_1)\quad \cdots\quad w(t+N_2)\right]^T\end{aligned}$$

Then J could be written as (in matrix form)

$$J = (\hat{\mathbf{y}}-\mathbf{w})^T\cdot(\hat{\mathbf{y}}-\mathbf{w})+\lambda\tilde{\mathbf{u}}^{\mathrm{T}}\cdot\tilde{\mathbf{u}} \quad \text{.....(11.)}$$

Minimise J to get optimum $\tilde{\mathbf{u}}$ we get:

$$\tilde{\mathbf{u}} = \left(\mathbf{G}_1^T\cdot\mathbf{G}_1+\lambda\mathbf{I}\right)^{-1}\cdot\mathbf{G}_1^T\cdot(\mathbf{w}-\mathbf{y}_o) \quad \text{.....(12.)}$$

Taking the first element of the control sequence (as described in the beginning of this Section.1)

$$\Delta u(t) = \mathbf{k}^T(\mathbf{w}-\mathbf{y}_o) \quad \text{.....(13.)}$$

Where: $\mathbf{k} = \left[1\quad 0\quad 0\quad \cdots\quad 0\right]^T\left(\mathbf{G}_1^T\cdot\mathbf{G}_1+\lambda\mathbf{I}\right)^{-1}\cdot\mathbf{G}_1^T$

The incremental controller ensures zero offsets even with nonzero constant disturbance. The choices of parameters (N_1, N_2, N_u and λ) determine the stability and performance of the GPC controller. Some guidelines for selecting them exist in [3,4].
Note: An interesting theorem of GPC stability is:
A system of n-states is stable under GPC control if:

1- The system is stablizable and detectable
2- If $N_u = N_1 \geq n, N_2 - N_1 \geq n-1, \lambda = \varepsilon \to 0$

Note: In order to ensure smooth transition between output and the set point, a first order filter with time constant (τ_e) is often used.

3. LINEAR NEURAL NETWORKS (LNN)

In this paper a simple type of the neural networks is used. It is called ADALINE or Linear Neural Networks (LNN). It consists of a single linear layer with N_I inputs and one output unit (Fig.1)

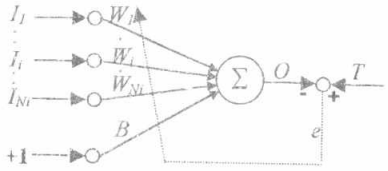

Fig.1: The Linear Neural Network (LNN) structure

Its mathematical model is written as:

$$O = \sum_{i=1}^{N_I} W_i \cdot I_i + B \cdot 1 \quad \text{.....(14.)}$$

Or in matrix form,

$$O = W \cdot I + B \quad \text{.....(15.)}$$

Where W, I and B have dimensions $[1 \times N_I]$, $[N_I \times 1]$ and $[1 \times 1]$ respectively.
Linear neural nets (LNN) are generally simpler to be trained than nonlinear ones. The Widrow-Hoff learning rule is often used for weight update [20]

$$\Delta W = \eta e I \quad \text{.....(16.)}$$

Where:

e is the local error between target and actual output ($T - O$)
η Learning rate, a small positive real number

Two ways of learning exist: *"Batch Learning"* and *"Incremental Learning"*. In *"Incremental Learning"* the weight update is calculated and applied with every training pair (i.e.input/target), where *"Batch Learning"* the weight update is averaged over a training set with reasonable batch size N_B.

The second type is used in this paper but the online version of it because of its accuracy and rapidity. The input and target signals are accumulated. Every N_B samples the training algorithm is applied to update the weights of LNN.

LNN is used in this paper to perform instantaneous discrete linear parametric identification of the controlled system, which is generally nonlinear.

A teaching signal with sufficient rich dynamics is applied to the to excite the system dynamics.

$$y_k = f\left(\underbrace{y_{k-1}, y_{k-2}, \cdots, y_{k-N_y}}_{N_y}, \underbrace{u_k, u_{k-1}, \ldots, u_{k-Nu+1}}_{N_u}\right) \quad \text{.....(17.)}$$

Where $f(.)$ is assumed to be a linear function.
To make the identification the current input (u_k) and previous system output (y_{k-1}) are given with sufficient delays (N_u and N_y) to the LNN as shown in Fig.2

Because the NN can only identify Bounded Input Bounded Output (BIBO) stable system the identification capability increases if the input and output signal are multiplied with $\Delta(=1\text{-}z^{-1})$.

$$\Delta y_k = f\left(\underbrace{\Delta y_{k-1}, \ldots}_{N_y}, \underbrace{\Delta u_k, \ldots}_{N_u}\right) \quad \text{.....(18.)}$$
$$\Delta y_k = f\left(Y_{k-1}, U_k\right)$$

Where U_k and Y_{k-1} are the inputs and output portion of the Neural Network input vector respectively.

Another feature proposed and used in this paper is to augment the NN with two additional parameters (U_{max} and Y_{max}) to represent the input and output range. The values of these parameters are determined online after each batch (N_B samples) and the whole training set is normalised with respect to them as follows:

$$\overline{U}_k = \frac{1}{U_{max}} \cdot U_k$$
$$\overline{Y}_{k-1} = \frac{1}{Y_{max}} \cdot Y_{k-1} \quad \text{.....(19.)}$$

Then the Input and Target pair to the LNN is formed as:

$$\bar{I}_i = \begin{bmatrix} \overline{U}_k \\ \overline{Y}_{k-1} \end{bmatrix} \qquad \left[N_u + N_y \times 1\right]$$
$$\overline{T}_i = \frac{\Delta y_k}{Y_{max}} \qquad \forall i \in \left[1, N_B\right]$$

The batch learning is performed on the normalised training set $\{\bar{I}_i, \bar{T}_i\}$ $i = 1,2,3,..,N_B$

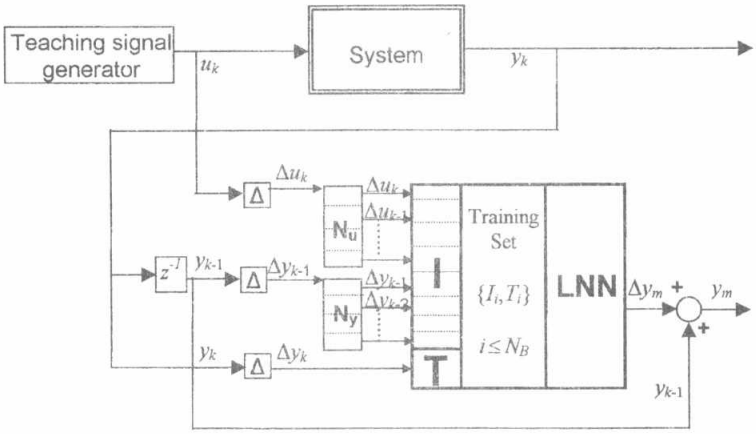

Fig.2: Online LNN Batch Identification

The network simulation equation of the scaled LNN is then:

$$\bar{O} = W \cdot \bar{I} + B \quad \text{.....(20.)}$$

In this case the bias is equal to zero (no disturbance case)

$$\bar{O} = W \cdot \bar{I}$$

$$\Delta\bar{y}_k = \left[W_u \vdots W_y\right] \cdot \begin{bmatrix} \bar{U}_k \\ \bar{Y}_{k-1} \end{bmatrix} \quad \text{.....(21.)}$$

$$\Delta\bar{y}_k = W_u \cdot \bar{U}_k + W_y \cdot \bar{Y}_{k-1} \quad \text{.....(22.)}$$

Where dimensions of W_u and W_y are [1×*Nu*] and [1×*Ny*]

$$\Delta y \in \left[-Y_{max}, Y_{max}\right],\ \Delta u \in \left[-U_{max}, U_{max}\right] \quad \text{.....(23.)}$$

Replace the scaled signals with its equivalent

$$\Delta y_k / Y_{max} = \left(1/U_{max}\right) \cdot W_u \cdot U_k + \left(1/Y_{max}\right) \cdot W_y \cdot Y_{k-1} \quad \text{.....(24.)}$$

Multiply by $Y_{\max}$ we get:

$$\Delta y_k = W_u \cdot U_k \times \left(Y_{max}/U_{max}\right) + W_y \cdot Y_{k-1} \quad \text{.....(25.)}$$

The parameter $\left(Y_{max}/U_{max}\right)$ is the most important parameter in the scaling of the LNN and for simplicity we will note it as $\left(Y/U\right) = \dfrac{Y_{max}}{U_{max}}$

When the I/O Bounds changes, and in order to keep the learned information the network have to generate the same output for the same input

$$O_1 = O_2 \qquad \text{for } I_1 = I_2$$

Where $(_1)$ denotes before scaling and $(_2)$ after scaling

$$(\Delta y_k)_1 = (\Delta y_k)_2 \quad \text{.....(26.)}$$

For the same $(U_k)_1 = (U_k)_2 = U_k$ and $(Y_{k-1})_1 = (Y_{k-1})_2 = Y_{k-1}$

$$(Wu)_1 \cdot U_k (Y/U)_1 + (Wy)_1 Y_{k-1} = (Wu)_2 \cdot U_k (Y/U)_2 + (Wy)_2 Y_{k-1} \quad \text{.....(27.)}$$

Then,

$$(W_y)_1 = (W_y)_2 \quad \text{.....(28.)}$$

$$(W_u)_2 = \frac{(Y/U)_1}{(Y/U)_2}(W_u)_1 \quad \text{.....(29.)}$$

Then in order to maintain the attained system identification information the input weight matrix (W_u)must be scaled with the factor $(Y/U)_{old}/(Y/U)_{new}$

Comparing the network simulation equation with CARIMA model without disturbance

$$A(q^{-1})\Delta y(k) = B(q^{-1})\Delta u(k) \quad \text{.....(30.)}$$

$$\Delta y_k = (Y/U)W_u \cdot \begin{bmatrix} \Delta u_k \\ \vdots \\ \Delta u_{k-N_u+1} \end{bmatrix} + W_y \cdot \begin{bmatrix} \Delta y_{k-1} \\ \vdots \\ \Delta y_{k-N_y} \end{bmatrix} \quad \text{.....(31.)}$$

Then,

$$\begin{bmatrix} 1 & \vdots & -W_y \end{bmatrix} \cdot \begin{bmatrix} \Delta y_k \\ \vdots \\ \Delta y_{k-N_y} \end{bmatrix} = (Y/U) \cdot W_u \cdot \begin{bmatrix} \Delta u_k \\ \vdots \\ \Delta u_{k-N_u+1} \end{bmatrix} \quad \text{.....(32.)}$$

Comparing we get:

$$\begin{aligned} A &= \begin{bmatrix} 1 & \vdots & -W_y \end{bmatrix} \\ B &= (Y/U) \cdot W_u \\ na &= N_y \\ nb &= N_u - 1 \end{aligned} \quad \text{.....(33.)}$$

A, B with dimensions $[N_y + 1 \times 1]$, $[N_u \times 1]$ respectively

4. LINEAR NEURAL GENERALIZED PREDICTIVE CONTROL (LNGPC)

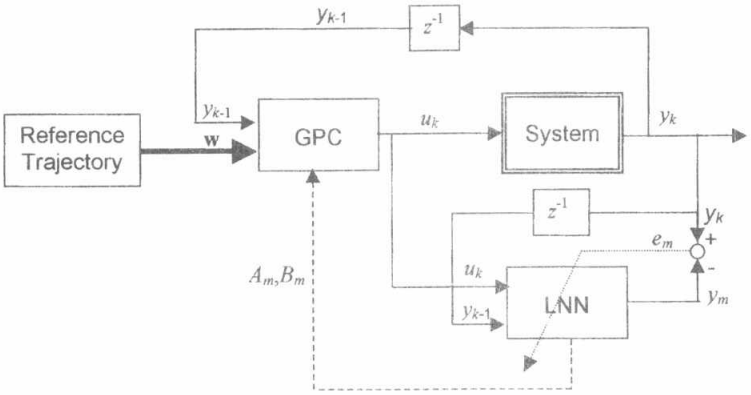

Fig.3: LNGPC control scheme

The operation of the LNGPC scheme (Fig.3) is as GPC system described in Section.2. A SISO controlled system is assumed in this scheme. GPC should start with a simplified reduced order model to be able to control the system.
After the first batch of samples, and every batch of samples, LNN extract the linear discrete transfer function as described in Section.3 and deliver it to the GPC controller.

An encouraging feature of the GPC controller is its robustness to modelling error and over and under parameterisation (error in estimate of system order). Another interesting feature of the present scheme is that the LNN makes both the linearization and identification at the same time with reasonably fast linear computations. So, this scheme is valid also for non-linear systems control.

The present control system is very suited to trajectory control applications. Its performance, as any MBPC scheme, is improved when the reference trajectory is given as a sequence of future desired values, not just as a single desired value, every time step.

In the next Section the LNGPC scheme is used successfully to control a two-channel multivariable weak coupled system.

5. APPLICATION

The adaptive capability of the proposed LNGPC- scheme is demonstrated using a realistic rotorcraft terrain-following application Fig.4. The results of the simulation are compared to the results of an adaptive GPC scheme with RLS identification [14]. The

simulation uses the same vehicle and controller parameters. As described in [14] the bare-airframe rotorcraft dynamics to be used are given by the following set of linear longitudinal state equations:

$$\begin{bmatrix}\dot{u}\\ \dot{w}\\ \dot{q}\\ \dot{\theta}\end{bmatrix}=\begin{bmatrix}-0.01 & 0 & 0 & -32.2\\ 0 & -1 & 101 & 0\\ 0 & 0 & -5.6 & -6.25\\ 0 & 0 & 1 & 0\end{bmatrix}\begin{bmatrix}u\\ w\\ q\\ \theta\end{bmatrix}+\begin{bmatrix}0 & 0\\ 0 & 1.5\\ 0.133 & 0\\ 0 & 0\end{bmatrix}\begin{bmatrix}\delta_B\\ \delta_C\end{bmatrix} \quad \text{.....(34.)}$$

Where:

u = Longitudinal velocity perturbation in x-body axis, [ft/sec]
w = Vertical velocity perturbation in z-body axis, [ft/sec]
q = Pitch rate, [rad/sec]
θ = Pitch attitude, [rad]
U_0 = Trim airspeed, 101 [ft/sec] (60 kts)
δ_b = Longitudinal cyclic control, % of full deflection
δ_c = Collective control, % of full deflection

Taking into consideration the realistic aspects, the authors of Ref. [14] use a stability augmented vehicle (Fig.5) with Stability Augmentation System (SAS) Transfer Functions: G_{az} and G_θ defined as,

$$G_\theta=\frac{909\left[(s/1.2)+1\right]}{\left[(s/1.2)+1\right]}$$
$$G_{az}=\frac{1.39(s+1)}{s^2} \quad \text{.....(35.)}$$

The commanded vertical flight path and horizontal speed are given by

$$h_c(t)=20\left(1-e^{-0.05t}\right)\cdot\left[sin(0.05(2\pi t))+sin(0.06(2\pi t))+sin(0.08(2\pi t))\right] \text{ [ft]}$$
$$u_c(t)=20\left(1-e^{-0.05t}\right)\cdot\left[sin(0.05(2\pi t))\right]+U_0 \quad \text{[ft/s]} \quad \text{.....(36.)}$$

The term $\left[1-exp(-.05t)\right]$ is added to relax the trajectory at the start of simulation (t=0) and to prevent unrealistic transitions
The atmospheric turbulence is not included in the simulation

Finally, the following non-linear kinematics equations are employed to describe the flight path:

$$\dot{h}=(U_0+u)sin\theta-w\,cos\theta$$
$$A_z=\dot{w}-(U_0+u)q \quad \text{.....(37.)}$$

An internal model of the vehicle is represented by the following two discrete transfer functions

$$u/\theta_c=\left[a_1z^{-1}+a_2z^{-2}+a_3z^{-3}+a_4z^{-4}\right]/\left[1-b_1z^{-1}-b_2z^{-2}-b_3z^{-3}-b_4z^{-4}\right] \quad \text{.....(38.)}$$

$$h/A_{zc} = \left[c_1 z^{-1} + c_2 z^{-2} + c_3 z^{-3} + c_4 z^{-4}\right] / \left[1 - d_1 z^{-1} - d_2 z^{-2} - d_3 z^{-3} - d_4 z^{-4}\right] \quad \text{.....(39.)}$$

The coefficients a_i, b_i, c_i and d_i are found by the LNN as described in Section.3

The simulation of the rotorcraft is done as following:

(1) The GPC activity begins with a control law developed from a nominal model of the vehicle. This nominal model employed reduced-order system transfer functions (table-1). Although the GPC is calculated initially using the reduced-order models, the vehicle simulation always uses the complete state space model of (Eq.34) with the stability augmentation system of (Eq.35) and (Fig.2). In this example the following GPC parameter values were selected as Ref. [14]:

N_1 = 1 (0.1 secs)
N_2 = 20 (2 secs)
N_u = 10 (1 sec)
τ_e = 0.5 secs
$\lambda_{a_{zc}}$ = 10, λ_{θ_c} = 7.10^4

(2) LNN is employed to identify the TF coefficients. (Eq.38,39) with a batch size of 100 samples (10 sec).
(3) Every 10 sec the internal model of GPC is updated with the new values extracted from the LNN
(4) Forty seconds into the run, the gain on the transfer function G_{az} in (Eq.35) was halved, simulating a "soft" failure in the stability augmentation system responsible for vertical acceleration control. After 10 sec the LNN extracts the new coefficients to be used by the GPC controller.

The results of simulation of the adaptive LNGPC system are shown in Fig.7. It should be noted that without the LNN online identification, the GPC with its initial model could not control the system and produces unstable results Fig.8. Finally, the results of Ref. [14] (Fig.9 in [14]) are compared with the present results in Fig.6.

The following remarks can be noticed:

- The actual and required outputs of Ref. [14] are apparently the same as the present simulations.
- In the present work the initial tuning period is the half (10sec instead of 20sec)
- The soft SAS failure produces more sever oscillations in [14] than the present simulations

IMPLEMENTATION

The simulations are done using MATLAB 5.3 / SIMULINK 3.0 environment. Neural Networks Toolbox 3.0 has been used as a base for developing the neural network model. Some features are added to the basic linear networks to be able to deal with online I/O bounds scaling and Online Batch Learning.

To implement the system of multivariable with separate channels, an array of NN objects is used. For the GPC controller, similar work was done. All the parameters of the controllers were stored in an object with the internal model then array of these objects is used also. Every GPC object is associated with a NN object and the system parameters are transferred in between every batch.

6. CONCLUSION AND FUTURE WORK

The present study concludes that:

- LNGPC can deal adaptively with time varying system.
- The proposed algorithm can work with even nonlinear systems.
- A physical interpretation was done to the LNN weights by terms of system parameters in system identification applications.
- LNGPC can successfully control a rotorcraft in a terrain-following application even with SAS failures.

And suggests as a future work:

- Study of LNGPC with multivariable more coupled systems.
- Study the real-time application of the proposed scheme.
- Work with system constraints.
- Work with disturbances.
- Work with a complete airplane model.

REFERENCES

[1] Garcia C.E. , Prett D.M. , and M. Morari," Model predictive control: theory and practice- a survey ," Automatica, 25(3): 335-348, (1989)

[2] Clarke D.W., C. Mohtadi and P.S. Tuffs," Generalized Predictive Control. Part1: the basic algorithm. ," Automatica, vol.23, Nº2, pp. 137-148, (1987)

[3] Clarke D.W., C. Mohtadi and P.S. Tuffs," Generalized Predictive Control. Part2: Extensions and interpretations," Automatica, vol.23, Nº2, pp. 149-160, (1987)

[4] Clarke D.W. and C. Mohtadi ," Properties of generalized predictive control," Proceedings of the 10th triennial world congress of IFAC, Munich, FRG, pp. 65-76, (1987)

[5] Clarke D.W. and R. Scattolini," Constrained receding-horizon predictive control," IEEE proceedings-D, vol.138, Nº4, pp.347-354, (1991)

[6] Kouvaritakis B. and Rossiter J. A. ," Multivariable stable generalized predictive control," IEE proc. D, 140(5): 364 - 372, (1993)

[7] Kinnaert M. ," Adaptive generalized predictive controller for MIMO systems," Int. J. Control, 50(1): 161- 172, (1989)

[8] Gossner J. R. , Kouvaritakis B. , and Rossiter J. A. ," Stable generalized predictive control with constraints and bounded disturbances," Automatica, 33(4):551-568, (1997)

[9] Kouvaritakis B. , Rossiter J. A., and Chang A. O. T. ," Stable generalized predictive control: an algorithm with guaranteed stability," IEE proceedings-D, 139 4, 349-362, (1992)

[10] Rawlings J. B. and Muske K. R. ," The stability of constrained receding horizon

control," IEEE Trans. Automatic control, 38(10):1512 – 1516, (1993)
[11] Rossiter J. A. , and Kouvaritakis B.," Constrained stable generalized predictive control," IEE Proc. D, 140(4): 243 – 254, (1993)
[12] Rossiter J. A. , Kouvaritakis B. and Gossner J. R. ," Feasibility and stability results for constrained stable predictive control," Automatica, 31(3):863 – 877, (1995)
[13] Scokert P. O. M. and Clarke D. W. ," Stabilizing properties of constrained predictive control," IEE Proc.-Control Theory Appl., 141(5):295 – 304, (1994)
[14] Hess R. A. , and Jung Y. C. ," Self-tuning generalized predictive control applied to terrain-following flight," AIAA paper No. 89-3450 AIAA guidance, navigation and control conference, boston, MA, (Aug.14-16 1989)
[15] Jung Y. C. and Hess R. A. ," Precise Flight-Path Control Using a Predictive Algorithm," Journal of Guidance, Control and Dynamics vol. 14, No. 5, pp. 936 – 942, (Sept.-Oct. 1991)
[16] Scott R. C. ," Active Control of Wind-Tunnel Model Aerolastic Response Using Neural Networks ," SPIE Paper 3991-30, (March 2000)
[17] Kvaternik R. G. , Juang J-N, and Bennett R. L. ," Exploratory Studies in Generalized Predictive Control for Active Aeroelastic Control of Tiltrotor Aircraft," NASA Report: NASA/TM-2000-210552, (October 2000)
[18] Clarke D.W.," Adaptive predictive control, Adaptive Systems in Control and Signal Processing," Proceedings of the 5th ACASP Symposium, Budapest, Hungary, pp.43-54., (1995)
[19] Soloway D. and Haley P. ," Neural Generalized Predictive control: A Newton-Raphson Implementation," IEEE CCA/ISIC/CACSD, IEEE Paper No. ISIAC-TA5.2, (Sept. 15-18, 1996)
[20] Widrow B. and . Hoff M. E. Jr. ," Adaptive Switching Circuits.," IRE WESCON Convention Record, Part 4, pp. 96-104. Reprinted in Anderson and Rosenfeld (1988), pp. 126-134., (1960)
[21] Widrow B., Mantey P. E. , Griffiths L. J. , and Goode B. B.," Adaptive Antenna Systems," Proceedings of the IEEE, 55: 2143-2159, (1967)
[22] Widrow B., and Lehr M. A. ," 30 Years of Adaptive Neural Networks: Perceptron, Madaline, and Backpropagation," Proceedings of the IEEE, 78(9): 1415-1442, (1990)
[23] Rumelhart D. E., Hinton G. E., and Williams R. J. ," Learning Internal Representations by Error propagation ," In D. E. Rumelhart and J. L. McClelland, eds., Parallel Distributed Processing, vol. 1 chapter 8. reprinted in Anderson and Rosenfeld (1988), pp. 675-695., (1986)
[24] Rumelhart, D. E., Hinton G. E. , and R. J. Williams. ," Learning Representations by Back-Propagation Error," Nature, 323: 533-536. Reprinted in Anderson and Rosenfeld (1988), pp. 696-699, (1986)
[25] Aymes J. M. and Bordeneuve-Guibe J. ," Advanced Predictive Control for Aircraft Guidance," Proc. of IFAC World Congress, Vol. P pp 249-254, (1996)

Table 1: System Transfer Functions

System transfer function (with SAS)

$$\frac{u}{\theta_c} = \frac{-356.8476s^2 - 356.8476s}{s^6 + 6.71s^5 + 12.577s^4 + 18.6423s^3 + 11.8924s^2 + 0.1171s} \qquad \frac{h}{a_{zc}} = \frac{-2.085s^4 - 13.782s^3 - 24.845s^2 - 13.278s - 0.13}{s^7 + 8.695s^6 + 25.698s^4 + 13.341s^3 + 0.13s^2}$$

Reduced-order transfer functions

$$\frac{u}{\theta_c} = \frac{-356.8476}{s^4 + 5.71s^3 + 6.867s^2 + 11.775s + 0.1171} \qquad \frac{h}{a_{zc}} = \frac{-2.085}{s^2(s + 2.0847)}$$

z-transfer functions of reduced-order system

$$\frac{u}{\theta_c} = \frac{-0.0013z^{-1} - 0.0131z^{-2} - 0.0117z^{-3} - 0.0009z^{-4}}{1 - 3.5081z^{-1} + 4.59z^{-2} - 2.6469z^{-3} + 0.565z^{-4}} \qquad \frac{h}{a_{zc}} = \frac{-0.0003z^{-1} - 0.0013z^{-2} - 0.0003z^{-3}}{1 - 2.8118z^{-1} + 2.6236z^{-2} - 0.8118^{-3}}$$

(Sampling time = 0.1 sec) (Sampling time = 0.1 sec)

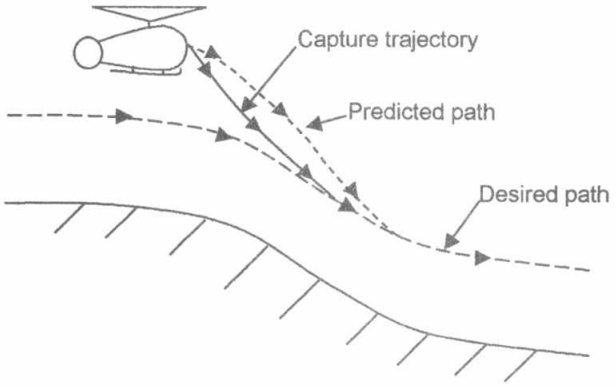

Fig.4: The Terrain-following task

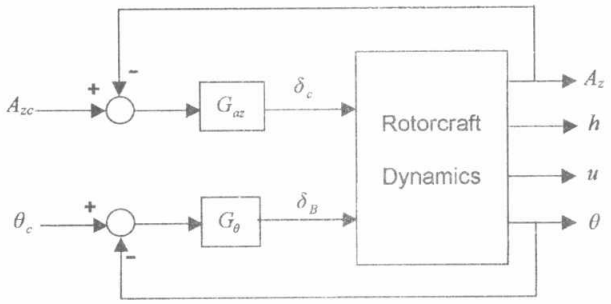

Fig.5: The Stability Augmentation System (SAS)

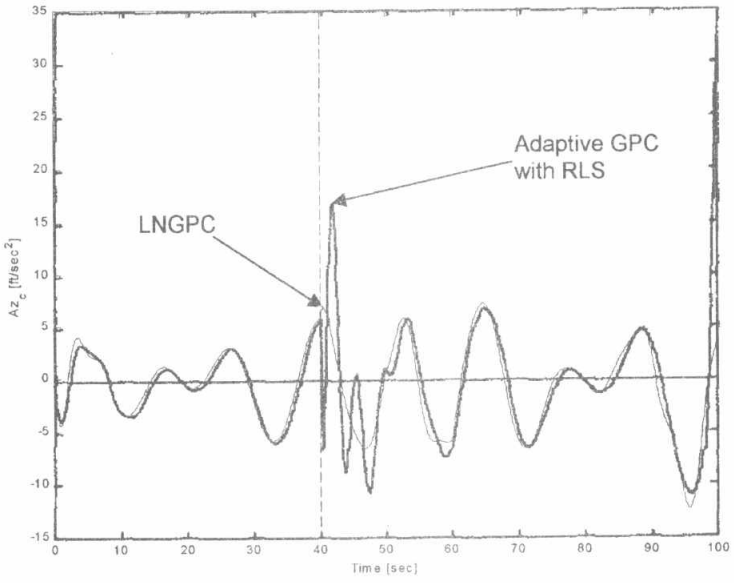

Fig.6: Comparing between Adaptive GPC and NLGPC

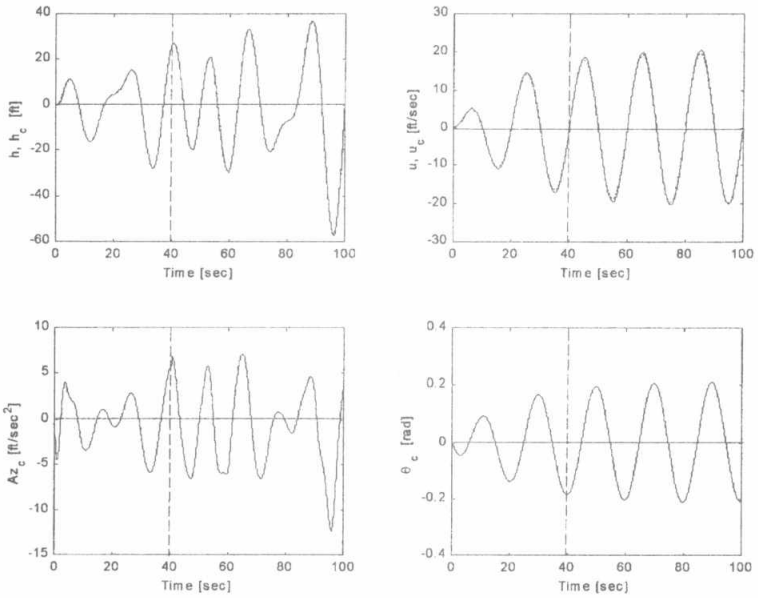

Fig.7: LNGPC, Rotorcraft, Terrain-following Application with NN identification [adaptation ON]

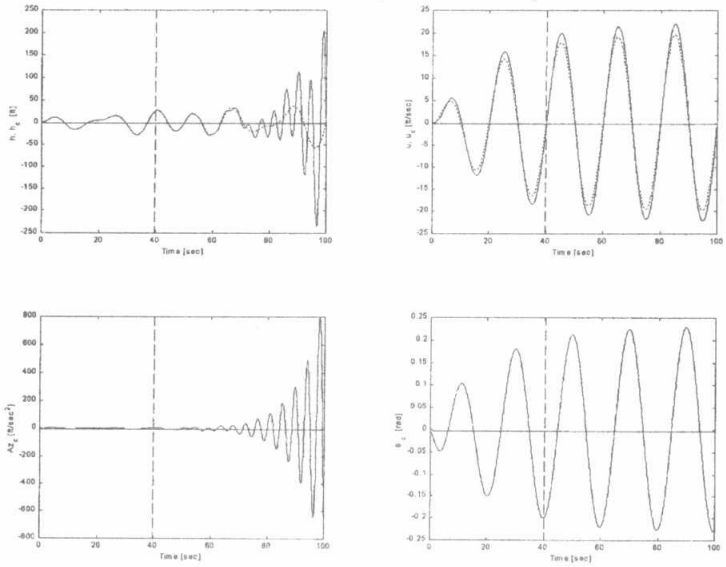

Fig.8: GPC, Rotorcraft, Terrain-following application without NN identification [adaptation OFF]