AAS 98-345

SATELLITE ATTITUDE CONTROL USING MULTILAYER PERCEPTRON NEURAL NETWORKS

Valdemir Carrara[+]
Sebastião Eduardo Corsatto Varotto[++]
Atair Rios Neto[+++]

This work simulates and tests the use of artificial neural networks for satellite attitude dynamics identification and control. In order to exemplify this application, a satellite with a rigid main body, three reaction wheels and three flexible solar panels was chosen (lay-out similar to Brazilian Remote Sensing Satellite). The main objective is to test the neural control and analyze its interaction with the elastic motion and variable geometry of the satellite. Two control schemes are used, the Internal Model Control (IMC) and a modified version of the Fedback Learning Control (FLC). The identification of neural nets parameters is performed by a Kalman filtering algorithm with a local parallel processing version in the IMC scheme and by the steepest descent method in the FLC scheme.

INTRODUCTION

In recent years the neural computing has evolved significantly. Main reason for the coming back of neural nets is, besides the increasing processing power of the new generation of computers, the development of new neural net architectures and training algorithms. The number of applications has also increased: vehicle guidance, financial analysis, printed circuit layout, voice synthesis and recognition, pattern classification, optical character recognition, exchange rate forecast, manufacturing process control and robotics among others (Ref. 1). Aeronautics also has found use for neural nets, mainly in failure analysis and detection, and automatic guidance and control. Although space applications are still limited, there are several possibilities: subsystem failure detection, isolation and identification, autonomous orbit propagation and control (Ref. 2), attitude determination and control, intelligent task managing, etc.

Attitude control of satellites normally is based on linearization of the dynamical equations of motion and application of an optimization method in order to guarantee the stability and controllability under the environmental conditions. Neural nets can overcome the non-linearities of the attitude behavior. Beyond the non-linearities inherent of the attitude dynamics, the effect of non-rigidity can also be present in the problem, due to flexibility of some structure component and to

[+] *Instituto Nacional de Pesquisas Espaciais – INPE/MCT*
CEP 12201-970 CP 515 São José dos Campos, SP. E-mail: val@dem.inpe.br

[++] *Instituto Nacional de Pesquisas Espaciais – INPE/MCT*
CEP 12201-970 CP 515 São José dos Campos, SP. E-mail: varotto@dem.inpe.br

[+++] *Instituto de Pesquisa e Desenvolvimento, Universidade do Vale do Paraíba.*
CEP 12245-720 São José dos Campos, SP. E-mail: atair@univap.br

geometry variation (due to module accretion, mass migration or appendage motion, for instance).

In what follows, two neural control methods are tested for attitude control, by using simulated data of a satellite attitude behavior where either flexibles appendages or variable geometry is present. Section 2 presents the general perceptron neural net as well as the training procedures. The equations of motion are presented in Section 3. Simulation, test results and conclusions follow the preceding sections.

NEURAL NETWORKS

A neural network is a computational structure composed of several basic units called artificial neurons. Each neuron can be understood as an operator that process with a nonlinear activation function f the weighted sum of its inputs to produce an output signal. The signal processing performed by the neuron establishes its functionality. The connections between the artificial neurons, on the other hand, define the behavior of the net, identify its applicability and training methods. In a multilayer perceptron network the neurons are grouped in one or more layers, with the output of each layer being the input to the next one.

The training process consists in adjusting the neuron weights based on the expected output and some optimization rules. In a supervised scheme the weights are adjusted interactively, by comparing the output of the network with the desired value at each step. This means that the training process teaches the net what should be its output for a given input.

A feedforward multilayer perceptron network can be seen as a mapping function with n_0 input elements and n_L outputs. This neural network is composed by L layers with n_l (l = 1, 2, ..., L) neurons in layer l. If x_i^l is the output of the i^{th} neuron of layer l, w_{ij}^l is the weight of the j^{th} input (coming from the j^{th} neuron of the preceding layer) and f^l is the activation function, then:

$$x_i^l = f^l\left(\bar{x}_i^l + b_i^l\right) = f^l\left(\sum_{j=1}^{n_{l-1}} w_{ij}^l\, x_j^{l-1} + b_i^l\right) \qquad (1)$$

where x_j^{l-1} is the output of the j^{th} neuron in previous layer l-1, and b_i^l is the bias, introduced to allow the neuron to present a non-null output even in presence of a null input.

The determination process of the neuron bias can be transferred to the determination of the neuron weights if one admits the presence of a new constant input. Eq. (1) can be expressed in vector-matrix form, and if W^l is the weight matrix, then

$$x^l = f^l\left(\bar{x}^l\right) = f^l\left(W^l x^{l-1}\right) \qquad (2)$$

where W^l includes the neuron bias in the last column; and the dimensions of the output vector x^l and the weight matrix W^l are now n_l+1 e $n_l \times n_{l-1}$+1, respectively.

A simple feedforward neural net with linear activation function in the output layer and the sigmoid activation function, Eq. (3), in the hidden layer can be used to represent dynamical systems and limited continuous functions (Ref. 3).

$$f(x) = \frac{1 - e^{-x}}{1 + e^{x}} \tag{3}$$

The increasing number of hidden layers normally makes the neural net to better represent the dynamical system and to reduce the output error (Ref. 4 and Ref. 5) even when the same number of neurons are taken. Nevertheless, the capacity of generalization, i. e. the ability to interpolate between points where the neural net was not trained, is more accentuated on nets with fewer or even only one hidden layer (Ref. 6). On the other hand, nets with high number of neurons or layers have small output errors at the trained points. Thus if the dynamics of the system is not complex a neural net with one hidden sigmoid and linear output layers is sufficient for a large number of applications. The number of neurons in the hidden layers is important for the approximation degree: few neurons tend to decrease the stability and result in a bad approximation, too many neurons cause oscillation on the output between the trained points (Ref. 7).

Backpropagation Algorithm

Training a neural net generally consists in applying methods in order to adjust or estimate the neuron weights. The training process normally minimizes the neural net output error through the application of an optimization method. All methods need to know how the net output varies with respect to the variation of a given neuron weight. This can be achieved with the back propagation algorithm (Ref. 8), which obtains the partial derivative of the output elements in a recursive way. In matrix form the back propagation algorithm gives the derivative of the output vector with respect to the j^{th} weight of the i^{th} neuron of the l^{th} layer:

$$\frac{\partial x^{L}}{\partial w_{ij}^{l}} = \Delta^{l}\left[0 \ldots \quad x_{j}^{l-1} \ldots \quad 0\right]^{T}, \tag{4}$$

where Δ^l is the back propagation matrix, obtained from:

$$\Delta^{l} = \Delta^{l+1} W^{l+1} F^{l} \tag{5}$$

with initial condition at output layer l given by $\Delta^L = F^L$, where F^l is a diagonal matrix with the derivatives of the activation function f^l:

$$F^{l} = \operatorname{diag}\left[\frac{df^{l}(\bar{x}^{l})}{d\bar{x}_{i}^{l}} : i = 1,2,\ldots,n_{l}\right] \tag{6}$$

It should be noted that, due to the inclusion of the neuron bias on the weight matrix, F^l should be a $n_{l-1}+1 \times n_{l-1}+1$ matrix, with the last diagonal element equal to zero. In order to reduce the computational effort both F^l and W^l can be resized with elimination of the last row when performing matrix products.

Steepest Descent Method

The steepest descent method, combined with the backpropagation, exhibits a high degree of paralelism and simplicity. The weights are corrected based on the minimization of the neural net output error. Weight updating starts at the net output layer and then the error is backpropagated to the preceeding layer in order to compute

its weight corrections. The minimization criterion uses the network output quadratic error as the performance index:

$$J(t) = \frac{1}{2}\varepsilon(t)^{\mathrm{T}}\ \varepsilon(t), \quad \varepsilon(t) = y^{d}(t) - y(t) \tag{7}$$

where $y^{d}(t)$ and $y(t)$ are expected and actual network output; and $\varepsilon(t)$ is the network output error at time t.

Weight updatings of layer k are performed using:

$$W^{l}(t+1) = W^{l}(t) - \lambda\, \nabla J^{l}\ , \ \nabla J^{l} = -\Delta^{l^{\mathrm{T}}} \varepsilon x^{l-1^{\mathrm{T}}} \tag{8}$$

where the gradient of the square backpropagated error ∇J^{l} comes from the backpropagation matrix.

Convergence of the weights depends on the adjusting of the learning rate coefficient λ, ranging from 0 to 1.

Stochastic Optimal Parameter Estimation Neural Nets Training

The supervised training of a neural net to learn a nonlinear continuous mapping:

$$f(x): x \in D \subset R^{nI} \rightarrow y \in R^{nO} \tag{9}$$

can naturally be treated as a problem of estimating the connection weight parameters w in the network correspondent mapping:

$$f^{e}(x,w): x \in D \subset R^{nI} \rightarrow y^{e} \in R^{nO} \tag{10}$$

such as to have $f^{e}(x, w)$ as close as possible to $f(x)$ for $x \in D$. A set of pairs $(x(t),y(t))$, t=1,2,...N, given by the mapping in Eq.(9) is selected in order to get this approximation, and the parameters are usually determined under the condition of minimizing

$$J(w) = \frac{1}{2}\left[[w-\overline{w}]^{T} \overline{P}^{-1} [w-\overline{w}] + \sum_{t=1}^{N} [y(t) - y^{e}(t)]^{T} R^{-1}(t) [y(t) - y^{e}(t)] \right] \tag{11}$$

where $\overline{w}$ is given a priori value of w; $y^{e}(t) = f^{e}(x(t),w)$; $\overline{P}^{-1}$ and $\overline{R}^{-1}(t)$ are weight matrices.

To solve the problem given by Eq. (11) an iterative scheme based on linear perturbation can to be used (Ref. 9). In a k^{th} typical iteration, one usually takes:

$$J(w(k)) = \frac{1}{2}\Big[[w(k) - \overline{w}]^{T} \overline{P}^{-1} [w(k) - \overline{w}] +$$

$$\sum_{t=1}^{N} \left[\alpha^{k} [y(t) - \overline{y}(k,t)] - f_{w}^{e}(x(t), \overline{w}(k)) [w(k) - \overline{w}(k)] \right]^{T}$$

$$R^{-1}(t) \left[\alpha^{k} [y(t) - \overline{y}(k,t)] - f_{w}^{e}(x(t), \overline{w}(k)) [w(k) - \overline{w}(k)] \right] \tag{12}$$

where k=1,2,. . ., k_c; $\overline{w}^l = \overline{w}$, $\bar{y}(k,t) = f^e(x(t).\overline{w}(k))$, $f_w^e(x(t),\overline{w}(k))$ is the matrix of first partial derivatives with respect to w; $0< \alpha^k \leq 1$ is an adjusting parameter to guarantee the hypothesis of linear perturbation. The solution of Eq.(12) is formally equivalent to the following stochastic parameter estimation problem

$$\overline{w} = w(k) + \bar{\varepsilon} \tag{13}$$

$$\alpha^k\left[y(t) - \bar{y}(k,t)\right] = f_w^e(x(t),\overline{w}(k))\left[w(k) - \overline{w}(k)\right] + \nu(t) \tag{14}$$

where, $E\left[\bar{\varepsilon}\right] = 0$, $E\left[\bar{\varepsilon}\ \bar{\varepsilon}^T\right] = \overline{P}$, $E\left[\nu(t)\right] = 0$, $E\left[\nu(t)\ \nu^T(t)\right] = R(t)$, usually diagonal; $E[.]$ is the expectation value operator; $\bar{\varepsilon}$ e $\nu(t)$ are assumed to be gaussian distributed and not correlated; and $\nu(t)$ is also assumed not correlated along t=1,2,. . .,N.

The problem of estimating the vector of weights w_i^l of neuron i of layer l , can be solved in a local way, through an estimator of Gauss Markov, in the Kalman form (Ref. 9); with the assumptions that:

(i) for weight parameters $w_a(k)$ of layers after the l[th] layer there are already available $\hat{w}_a(k)$ and $P_a(k)$ the estimated values of parameters and of the covariance matrix of the associated erros e_a(k);

(ii) for weight parameters $w_s(k)$ of ther neurons in the same layer l, there are a priori estimates $\bar{w}_s(k)$ and $\overline{P}_s(k)$;

(iii) for weight parameters $w_e(k)$ of the earlier layers there are a priori estimates $\overline{w}_e(k)$ and $\overline{P}_e(k)$.

In a typical iteration k=1,2,..., kc, thus results the local estimation:

$$\hat{w}_i^l(k) = \overline{w}_i^l + \overline{K}_i^l(k)\left[\overline{z}_i^l(k) - H_i^l(k)\overline{w}_i^l(k)\right] \tag{15}$$

$$P_i^l(k) = \left[I - \overline{K}_i^l(k)H_i^l(k)\right]\overline{P}_i^l \tag{16}$$

$$\overline{K}_i^l(k) = \overline{P}_i^l H_i^l(k)^T\left[H_i^l(k)\overline{P}_i^l H_i^l(k)^T + \overline{R}_i^l(k)\right]^{-1} \tag{17}$$

where $\overline{R}_i^l(k) = E\left[\overline{\nu}_i^l(k)\overline{\nu}_i^l(k)^T\right.$ is the covariance matrix of observation errors and can be evaluate as:

$$\begin{aligned}\overline{R}_i^l(k) = {} & f_{w_a}(x(t),\overline{w}(k))P_a(k)f_{w_a}(x(t),\overline{w}(k))^T + \\ & f_{w_s}(x(t),\overline{w}(k))\overline{P}_s(k)f_{w_s}(x(t),\overline{w}(k))^T + \\ & f_{w_e}(x(t),\overline{w}(k))\overline{P}_e(k)f_{w_e}(x(t),\overline{w}(k))^T + R(t)\end{aligned} \tag{18}$$

ATTITUDE DYNAMICS

The variation rate of the angular momentum, expressed in body coordinates x^o, y^o and z^o, is (Ref. 11) and (Ref. 12):

$$\dot{L}^o = I_o \dot{\omega}_o^o - \Omega(\omega_o^o) I_o \omega_o^o = N_{cont} + N_{pert} \tag{19}$$

where I_o is the satellite inertia matrix, ω_o^o is its angular velocity; $\Omega(\cdot)$ is the vector product matrix, defined by:

$$\Omega(\omega) = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \tag{20}$$

and external torque is separated in environmental or disturbance torque, N_{pert} and attitude control torque, N_{cont}. If the satellite is composed of articulated appendages, or if some appendages like the solar arrays are flexible, the above equations shall be modified in order to reflect the effects caused by the non-rigidity.

Articulated Appendages

An articulated satellite has a variable geometry, due to the relative motion of the appendages. Consider, for instance, a spacecraft pointed to Earth with solar arrays tracking the Sun, or the process of unfolding the solar arrays after orbit injection, or a robot space arm or even the docking of a new module in a space station. In all these examples, both the inertia and center of mass position vary in time. Suppose that a rigid main body with n articulated and also rigid appendages composes the satellite. In order to avoid increasing the system degrees of freedom, the angular velocities and accelerations of each articulation is supposed known. This is true for a large number of satellites, as for example the Earth pointing satellite which drives the solar arrays to the Sun.

The angular momentum rate of the satellite can now be expressed as a sum of the individual momenta:

$$\dot{L}^o = \int_{V_o} \Omega(r_o^o)\, \ddot{r}_o^o \, dm_o + \sum_{k=1}^{n} \int_{V_k} \Omega(r_k^o)\, \ddot{r}_k^o \, dm_k \,. \tag{21}$$

where r_o and r_k are, respectively, the position of the mass elements dm_o e dm_k, belonging to the main body and the appendage k ($k = 1, \ldots, n$). The momentum rate with respect to the satellite center of mass and the position vectors are expressed in main body coordinates. V_o and V_k are the volumes of the main body and appendage k. The above integral yields:

$$\dot{L}^o = (I_o + J_n)\dot{\omega}_o^o + \Omega(\omega_o^o) I_o \omega_o^o + H_n \,. \tag{22}$$

Except for J_n and H_n, the angular momentum rate is similar to the Eq. (14). J_n and H_n represent the appendage and center of mass motion effects. They are defined by:

$$J_n = \sum_{k=1}^{n} \left(A_{k,o}\, I_k\, A_{k,o}^T - m_k\, \Omega(a_{ok}^o - a_{ko}^o)\, \Omega(a_{ok}^o - a_{ko}^o) \right) +$$

$$+ \sum_{k=1}^{n} \left(m_k\, \Omega(a_{ok}^o - a_{ko}^o) \right) \sum_{k=1}^{n} \left(\mu_k\, \Omega(a_{ok}^o - a_{ko}^o) \right) \tag{23}$$

and

$$H_{,,} = \sum_{k=1}^{n}\left[\Omega(\omega_o^o + \omega_k^o)A_{k,o}I_k A_{k,o}^T(\omega_o^o + \omega_k^o) - A_{k,o}I_k A_{k,o}^T(\dot{\omega}_k^o + \Omega(\omega_o^o)\omega_k^o)\right] +$$

$$+\sum_{k=1}^{n} m_k \Omega(a_{ok}^o - a_{ko}^o)\beta_k - \left(\sum_{k=1}^{n} m_k \Omega(a_{ok}^o - a_{ko}^o)\beta_k\right)\sum_{k=1}^{n}\mu_k\beta_k , \qquad (24)$$

where I_k is the inertia matrix of appendage k expressed in the appendage coordinate system. $A_{k,o}$ is the rotation matrix between the appendage k and the main body coordinate systems and m_k is the appendage mass. The position of a fixed point in the articulation k defines the vector a_{ok}, with respect to the origin of the main body and a_{ko}, with respect to the origin of the appendage frames. The mass proportion μ_k is defined by:

$$\mu_k = \frac{m_k}{m_o + \sum_{k=1}^{n} m_k} \qquad (25)$$

where m_o is the main body mass. The angular acceleration β_k is given by:

$$\beta_k = \Omega(\omega_o^o)\Omega(\omega_o^o)(a_{ok}^o - a_{ko}^o) - \Omega(\omega_o^o)\Omega(\omega_k^o)a_{ko}^o -$$
$$-\Omega(\omega_k^o)\Omega(\omega_o^o + \omega_k^o)a_{ko}^o + \Omega(a_{ko}^o)\left(\Omega(\omega_o^o)\omega_k^o + \dot{\omega}_k^o\right) \qquad (26)$$

Note that the appendage angular velocity ω_k^o and acceleration $\dot{\omega}_k^o$ vectors define both the momentum and the direction of the articulation joint. Equations of motion can now be integrated in order to simulate the attitude of a satellite with variable geometry.

Flexible Dynamics

In this case the equations of motion are obtained by the Lagrangian approach for quasi-coordinates (rotational motion) and for generalized coordinates (elastic motion) . The development is addressed to a peculiar class of satellites constituted of a rigid central body also containing rigid rotors, and rectangular solar panels which are considered flexible after deployment.

The Lagrangian formulation for quasi-coordinates and for generalized coordinates (Ref. 13) has been used to derive the equations of motion as well as Meirovitch notation. A flexible spacecraft represents a distributed-parameter system which in theory has an infinite number of degrees of freedom. In practice, the system must be discretized, to avoid partial differential equations in the formulation. It can be done by the finite element technique, the lumped parameter method or the assumed modes method. In this work the last one was used and thus, the elastic displacement vector can be written as a linear combination of space-dependent admissible vector functions ϕ multiplied by time-dependent generalized coordinates (Ref. 13) q in thes form:

$$\{v\} = [\phi]\{q\} \qquad (27)$$

where $[\phi]$ is a rectangular matrix of space-dependent admissible functions and $\{q\}$ is time-dependent vector of generalized coordinates.

Taking into account this discrretization procedure, the kinetic energy can be written as:

$$T = \frac{1}{2}\{\omega\}^T[J]\{\omega\} + \frac{1}{2}\{\Omega\}^T[I]\{\Omega\} + \frac{1}{2}\{\dot{q}\}^T[M]\{\dot{q}\} + \{\omega\}^T[M]\{\Omega\} + \{\omega\}^T[H]\{\dot{\epsilon}\} \quad (28)$$

where [J] and [I] are the inertia matrices of the satellite in deformed state and of the rotors, respectively; $\{\omega\}$ e $\{\Omega\}$ are the angular velocity vectors of the satellite (absolute) and of the rotor (relative to the satellite), respectively; $\{\dot{q}\}$ is the rate of change in time of the generalized elastic displacement vector, and finally [M] e [H] are matrices involving integrals of space-dependent admissible functions.

The elastic potential energy can be written as:

$$V = \frac{1}{2}\{q\}^T[K]\{q\} \quad (29)$$

where [K] is a symmetric matrix involving spatial derivatives of the admissible functions.

The modified dynamics Euler's Equations were them derived by the Lagrangian Formulation for quasi-coordinates, resulting:

$$[J]\{\dot{\omega}\} + [\dot{J}]\{\omega\} + [\tilde{\omega}][J]\{\omega\} + [\tilde{\omega}][I]\{\Omega\} + [\tilde{\omega}][H]\{\dot{q}\} + [H]\{\ddot{q}\} = \{T_P\} - \{T_C\} \quad (30)$$

where $[\tilde{\omega}]$ is the same as $\Omega[\omega]$ in Eq. 20.

The elastic dynamic equations have been derived by the Lagrangian formulation for generalized coordinates and are given by:

$$[M]\{\ddot{q}\} + [H]^T\{\dot{\omega}\} + [K]\{q\} - \{F\} = \{Q_q\} \quad (31)$$

where $\{F\}$ involves partial derivatives of [J] relative to generalized elastic coordinates.

The Kinematics Equations were written using the Euler Parameter :

$$\{\dot{q}^*\} = \frac{1}{2}[\tilde{\Omega}^*]\{q^*\} \quad (32)$$

where $[\tilde{\Omega}^*]$ is a matrix composed by components of the satellite angular velocity and $\{q^*\}$ is the quaternion of satellite attitude.

In this study only the Gravity Gradient torque as external perturbation (Ref.14) and the first out-of-plane bending mode for each solar arrays were considered. The in-plane and torsional modes were not considered. This could be assumed because the solar arrays are short and somewhat rigid in the satellite studied.

SIMULATION RESULTS

Satellite with Variable Geometry

The neural network control (NNC) was implemented and simulated using MECB (Brazilian Complete Space Missions) satellite characteristics. They are small

satellites designed to test low Earth orbit communications and to perform Earth observation.
Immediately after orbit injection, the spacecraft shall perform a rate reduction, in order to stop the tumbling and rotation motion imposed by the launcher's last stage and separation torque. The satellite then opens 3 solar panels and enters in attitude acquisition in order to point the panels to Sun. During the deployment, the mass motion of the solar arrays changes the satellite inertia and center of mass position. It was supposed that a neural net controls the attitude of the satellite in this phase. For attitude data acquisition, the satellite uses a magnetometer and an analog sun sensor. Attitude is controlled with hydrazine thrusters, on 3 axes, with a torque generation of 0.19 Nm maximum.

The network training process uses the attitude response to the torque control in order to update the neural weights. A feedback learning control (FLC) algorithm (Ref.15) was initially employed to train the network. However, FLC showed a strong competition between the neural and the PID controls. If the neural signal u^c was opposite to the PID output u^d, then the satellite remained uncontrolled, and the feedback error kept the process as if it as, in a steady state. Another important drawback of the FLC was the absence of a feedback dynamical signal at the neural network input. If the network is driven only by a reference trajectory, then it can't generate torque when the trajectory reaches the final point and the residual attitude errors is not corrected. A different approach was adopted, as shown in Figure 1. The neural network receives inputs from the trajectory error and the output torque. The learning signal, as in the FLC, comes from the PID controller, but instead of combining both PID and NNC, only the network output torque controls the attitude. The learning process obtains the weights that minimize the PID signal. Due to the delay in the feedback error, some torque oscillations may occur, and the control becomes unstable. In order to avoid this behavior, the network output torque was also added to the learning signal, as shown also in Figure 1. This procedure not only guarantees the control stability, but also tends to minimize the control output and therefore the hydrazine consumption.

Unfortunately, the process of adjusting the PID gains and the network feedback torque gain K_c was very difficult, as the stability of NNC teaching was assured only within a reduced gain margin. The learning rate coefficient λ had to be small, in order to compensate the deviations of the learning signal from the unknown teaching control u^d.

In the attitude simulations were carried out to teach the neural control, propagation time was 1000 s of duration, with time step of 1 sec. The solar arrays were opened at t = 500 s. Random initial conditions were selected uniformly distributed between ± 45° attitude angles and ± 0,5 rd/s angular velocity. Reference trajectory y^d was fixed with null angular rate.

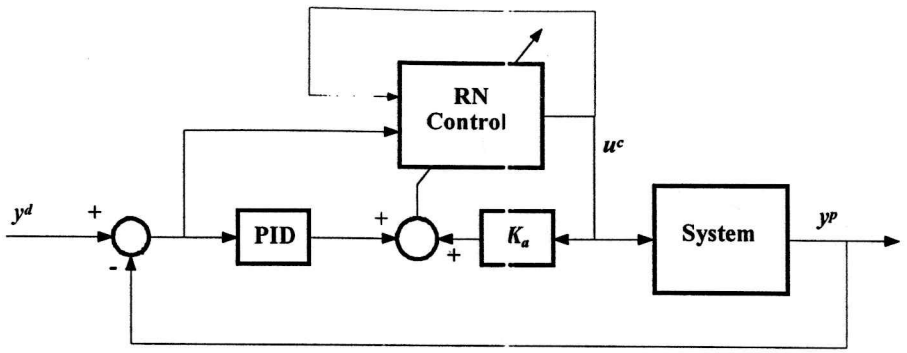

Fig. 1 – Feedback error learning control without PID supervision.

Neural network inputs were composed by the attitude angles φ^p, η^p and ψ^p, (from a *XYZ* rotation), the components of the satellite angular velocity, ω_x, ω_y and ω_z and the solar array deployment sensor angle at time *t*. In order to provide information about the attitude dynamics, these values at instant *t*, *t*-1 ... *t*-3 were also given. The input vector contains also the components of the output torque τ_x, τ_y and τ_z at times *t*-1 ... *t*-4. The network was composed of 40 neurons in the hidden layer (with sigmoid activation function) and 3 output neurons for torque generation with hard limited linear activation function. The learning rate coefficient, λ, was adopted as 0.001 after several trials with different values. The PID controller gains was 0.08, 0.05 and 20, respectively. These values were obtained by trials, based on learning convergence and stability and do not reflect any optimization criteria.

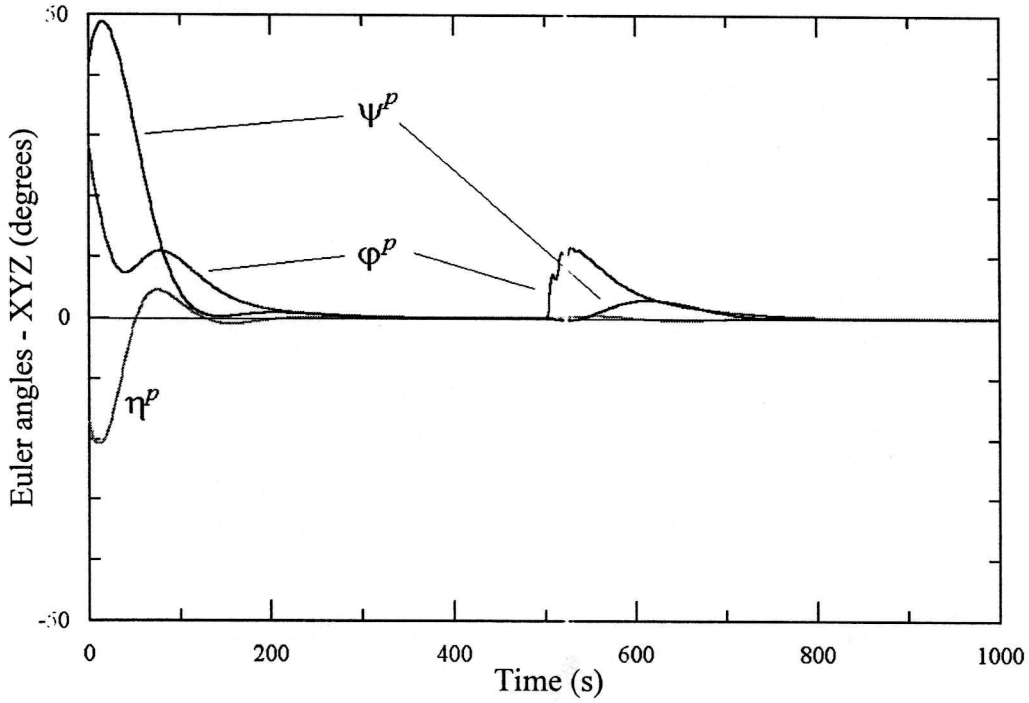

Fig. 2 – Satellite attitude during solar array deployment with a FLC without PID supervising.

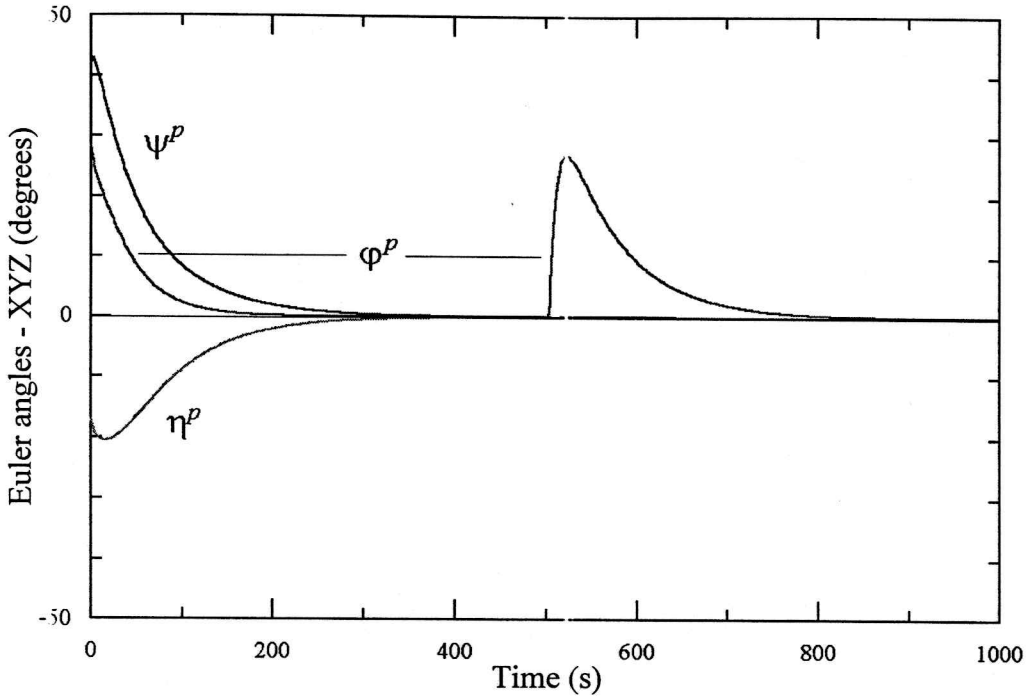

Fig. 3 – Satellite attitude during solar array deployment with PID control.

The same is true for the K_c gain, adjusted in 0.02. After the training process (6000 interactions), the neural net was used to control the satellite starting with a different attitude, shown in Figure 2. As can be seen, NNC can provide an effective attitude control even without the presence of the PID supervision.

The attitude motion was then compared with that of an exclusive PID controller, with the same gains used to train the neural network. As shown in Figure 3, the PID exerts a control on the satellite similar to that of the NNC when no geometry variation occurs. The main difference, as expected, happens when the solar arrays are opened. In such a situation the NCC performance is better than the PID, mainly due to the adaptation caused by the deployment information.

Satellite with Flexible Appendages

The control structure used in this implementation is known as Internal Model Control (IMC) (Ref. 16). In this structure an Artificial Neural Network (ANN) is trained to behave as the dynamic system (direct model). Soon after, a second ANN, the control network is trained according to the inverse model, using in the training the retro-propagation of error in the direct model disturbances. The difference between the real trajectory of the plant and the trajectory supplied by the direct model is used then in the form of feedback to correct the state and to compensate the effects of the disturbances . Due to the fact that the nets are not fed with information about the disturbances " d " that affect the behavior of the system, they don't get to eliminate the nonlinear errors in the trajectory due to the effects of these disturbances (Ref.17).

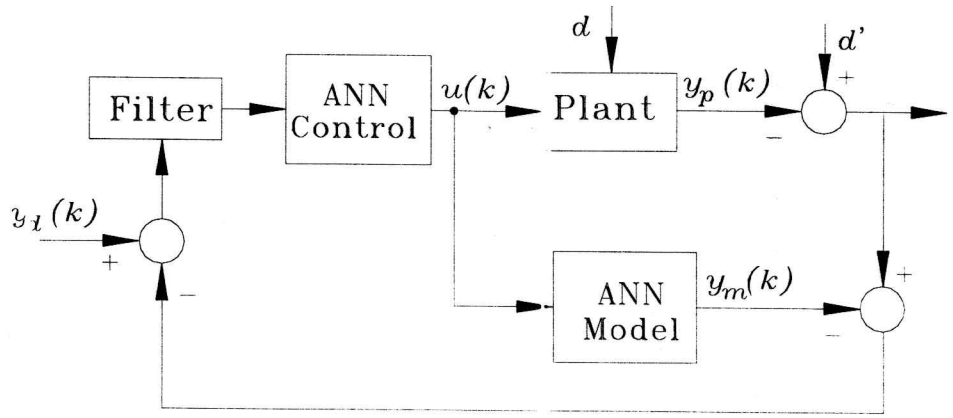

Fig.4 - Internal Model Control (IMC)

The neural network control (IMC) was implemented and simulated using a satellite with configuration similar to MECB remote sensing satellite characteristics. During the phase of fine pointing, the satellite will have a horizon infra-red and fine solar sensor, positioned in an appropriate way on the main body of the satellite. In this phase of mission the satellite will have three actuators of the type Reaction Wheel with a maximum torque generation of 0,2 Nm, to supply the torque demanded by the control system.

The first step for the implementation of the neural control was to make the identification of an ANN for the direct model, which had as inputs the control torque, the displacements and the angular velocity at instants *t*, *t-1* and *t-2*. After some tests varying the number of neurons in each layer and verifying the error at the end of the net training, it was adopted a configuration composed by 22 neurons in the input layer (21 elements and one more due to the " bias "), 30 neurons in the first hidden layer, 10 neurons in the second hidden layer and 6 neurons in the exit layer. The hyperbolic tangent activation function was adopted for all the neurons.

The second step was the identification of the inverse model. This training was executed in an off-line way using the specialized inverse model (Ref. 3), with an input vector similar to that used previously.The topology of the control network was established taking as a basis the general lines delineated for the identification of the direct model net; tests led to a configuration composed by 25 neurons in the input layer, 30 neurons in the first hidden layer hide, 10 neurons in the second hidden layer and 3 neurons in the exit layer. The hyperbolic tangent activation function was adopted for all the neurons.

Simulations were made involving attitude maneuvers where perturbations during orbit corrections were considered, with several initial conditions to evaluate the performance of the proposed scheme. A typical maneuver is shown with the objective of illustrating the performance of the control scheme. The Figures 5 and 6 show respectively the response of the attitude angle and angular velocity in relation to the reference signal. Figure 7 shows the torque demanded to the actuator for the maneuver in the pitch axis. It is observed from these results, that at the end of the pointing maneuver, the attitude angle as well as the angular speed of the satellite are inside the acceptable accuracy. It is also noticed that the torque applied to the rotor stayed limited to the compatible values.

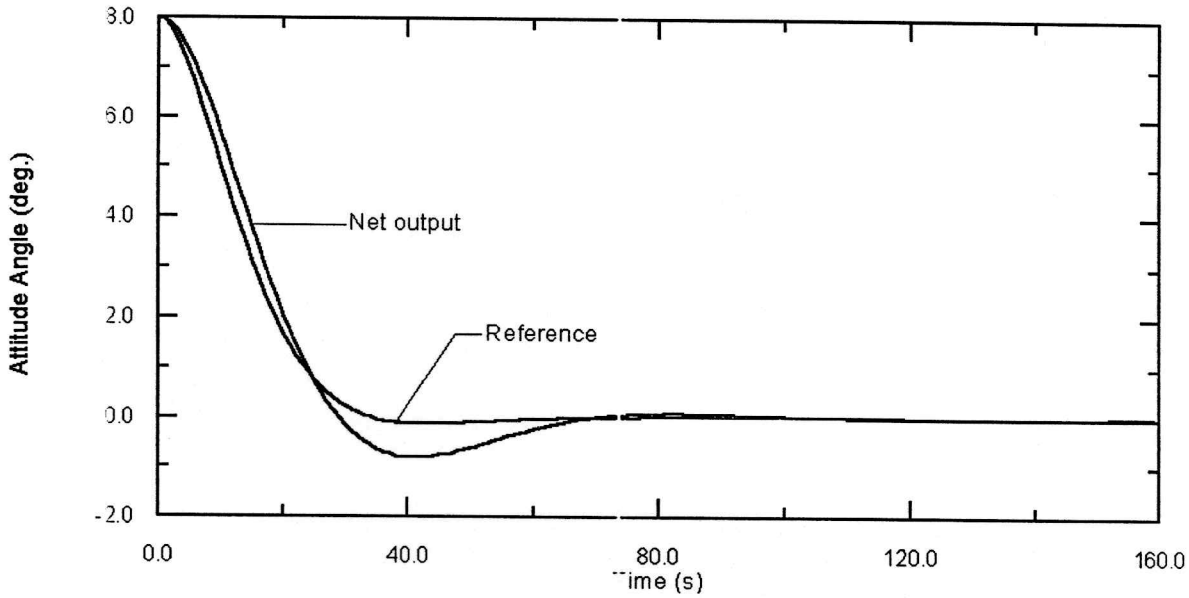

Fig. 5 - Attitude angle.

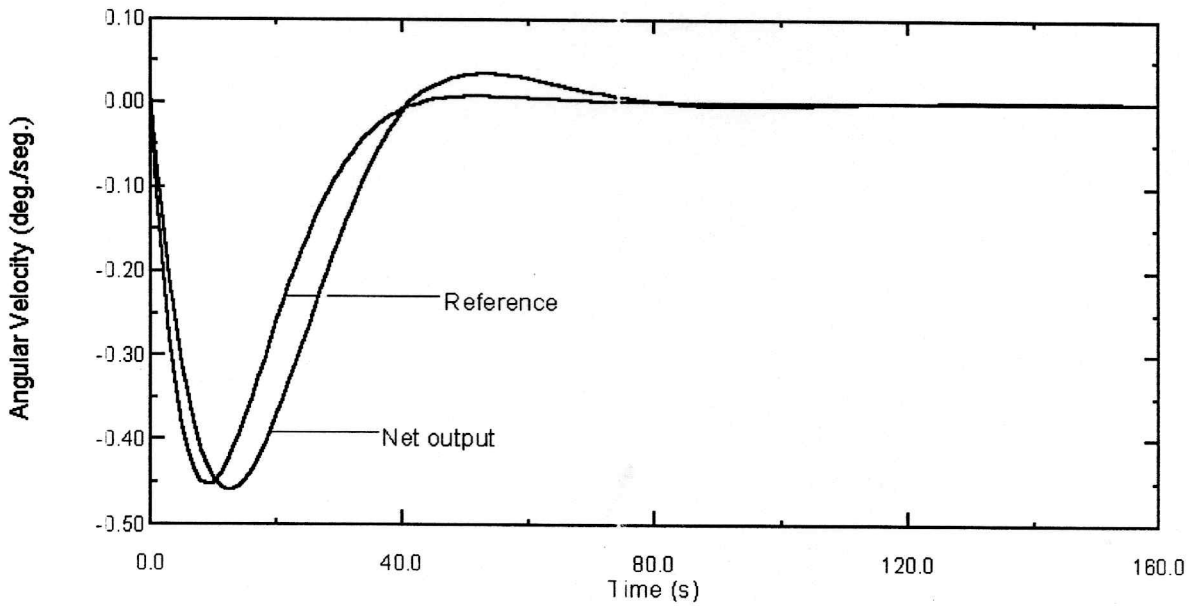

Fig. 6 - Angular velocity.

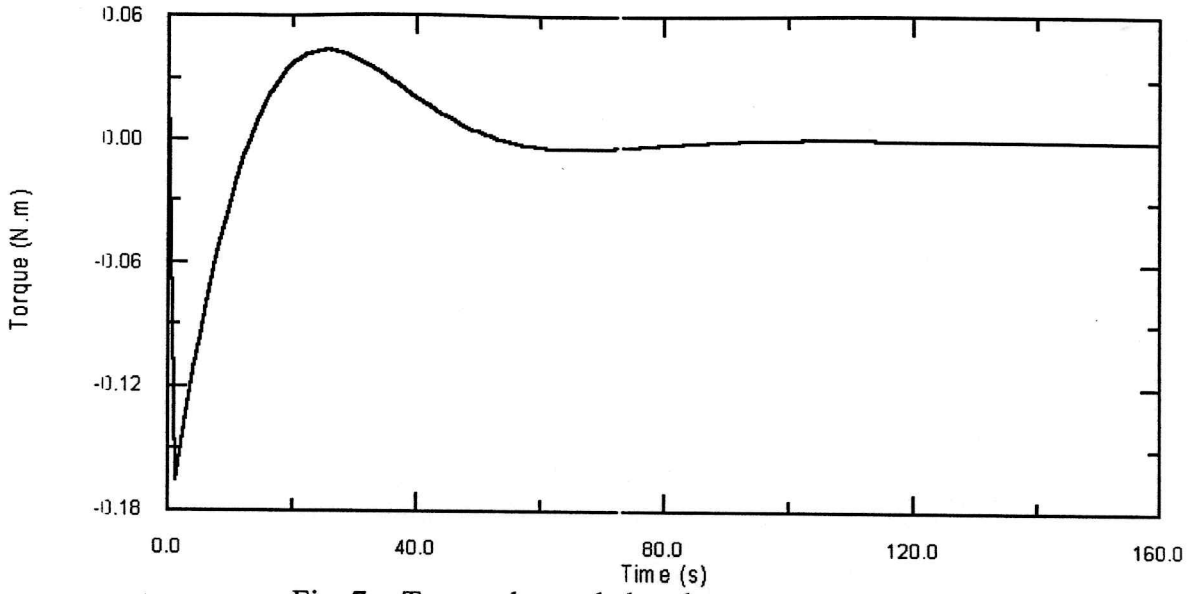

Fig. 7 - Torque demanded to the actuator.

In the tested situations, the neural control procedure was able to execute the satellite pointing maneuver. The free oscillations in the extremity of the panels (not shown in the previous graphs) stayed quite small with values of the order of 3x10-5 mm, not introducing any type of sensitive disturbance in the attitude of the vehicle.

CONCLUSIONS

Two attitude control schemes using multilayer perceptron neural networks were developed and tested under simulated conditions of use. The first one was an attitude controller for a satellite with variable geometry derived from the feedback error learning algorithm, without the PID control supervision. The results indicated that the performance of the NNC can, under certain conditions, be better than that of a conventional PID controller. The second one was an attitude controler for a satellite with flexible appendages using the IMC control procedure. Results obtained with this scheme are very encouraging. It could be verified that the strong point of ANNs is really their capacity of non linear mapping, mainly in the identification of the System Direct Model. In the Inverse Model identification, special care should be taken concerning the choice of the variables to represent the dynamic system, since they play a fundamental role in obtaining the correct inverse mapping of the plant.

The control schemes with " off-line " training of the ANNs facilitate a more immediate application, however its reliability and robustness are limited, because such controllers possess a restricted operation and are not capable to compensate eventual disturbances or spurious interactions between the environment and the plant to be controlled. Further studies shall address adaptive schemes using special computational structures and training algorithms for "on-line" retraining of the ANNs.

BIBLIOGRAPHY

1. Demuth, H; Beale, M. *Neural network toolbox user's guide.* Natick, MA. Math Works, 1992.

2. Rios Neto, A.; Rao, K. R. *A study on the on board artificial satellite orbit propagations using artificial neural networks.* Proceedings of the 11th International Astrondynamics Symposium, Gifu, Japan,1996.

3. Hunt, K. J.; Sbarbaro, D.; Zbikowski, R.; Gawthrop, P. J. Neural networks for control systems - a survey. *Automatica.* 28 (6):1083-1112, 1992.

4. Nguyen, D. H.; Widrow, B. Neural networks for self-learning control systems. *IEEE Control Systems Magazine*, 10 (3), Apr. 1990.

5. Chen, S.; Billings, S. A. Neural networks for nonlinear dynamic system modelling and identification. *International Journal of Control*, 56 (2):319-346, 1992.

6. Baffes, P. T.; Shelton, R. O.; Phillips, T. A. *NETS, a neural network development tool.* Lyndon B. Johnson Space Center, JSC-23366, 1991.

7. Billings, S. A.; Jamaluddin, H. B.; Chen, S. Properties of neural networks with applications to modelling non-linear dynamical systems. *International Journal of Control.* 55 (1):193-224, 1992

8. Zurada, J. M. *Introduction to Artificial Neural Systems*, West Publishing Co. 1992.

9. Rios Neto, A. Stochasttic Optimal Linear Parameters Estimation and Neural Nets Training in Systems Modeling. *Journal of the Brazilian Soc. Mechanical Sciences*, Vol. XIX, nº 2, 138-146, 1997.

10. Gelb, A. *Applied Optimal Estimation.* The M.I.T. Press, MA, 1974.

11. Crandall, S. H.; Kornopp, D. C.; Kurtz Jr, E. F.; Pridmore-Brown, D. C.; *Dynamics of mechanical and electromechanical systems.* Mc Graw-Hill, NY, 1968.

12. Wertz, J. R. *Spacecraft attitude determination and control*, London, D. Reidel, 1978 (Astrophysics and Space Science Library).

13. Meirovitch, L. *Method of Analytical Dytiamics*, New York, McGraw-Hill, 1970.

14. Kaplan, M. H. *Modern Spacecraft Dynamics.* USA, Jhon Wiley & Sons Inc., 1973.

15. Chen, S.; Billings, S. A.; Grant, P. M. Non-linear system identification using neural networks. *International Journal of Control*, 51 (6):1191-1214, June 1990.

16. Narendra, K. S.; Parthasarathy, K. Identification and control for dynamic systems using neural networks. *IEEE Transactions on Neural Networks*, 1, 1990.

17. Garcia, C. E. ; Morari, M. Internal model control - . A Unifying Review ans Some New Results, *Ind. Eng. Chem. Process Des. Dev.*, vol. 21, pp. 308-323, 1982.