A STUDY ON ATTITUDE CONTROL SCHEMES BASED ON NEURAL NETWORK STRUCTURES

V.S. KODOGIANNIS

10-99

Preprint no. 10-99/1999

Department of Computer Science
University of Ioannina
451 10 Ioannina, Greece

A STUDY ON ATTITUDE CONTROL SCHEMES BASED ON NEURAL NETWORK STRUCTURES

V.S. Kodogiannis

Department of Computer Science, University of Ioannina, Ioannina, GR-45110, Greece, Tel: +30-651-97308, Fax: +30-651-48131, Email: vassilis@cs.uoi.gr

ABSTRACT: The attitude control problem of a rigid body, such as a spacecraft, in three-dimensional space is addressed, by introducing two new control strategies developed in hypercomplex algebra. The proposed approaches are based on two parallel controllers both derived in Quaternion Algebra. The former is a feedback controller of PD type, while the latter is a feed-forward controller, which is implemented by means of either a Hypercomplex Multilayer Perceptron Network or a Hypercomplex Radial Basis Functions Network. The performances of both hybrid approaches are evaluated through simulations. These results are also compared with a classical PD controller and an adaptive controller, showing the improvements due to the use of the neural networks as feed-forward controllers, especially when an external disturbance acts in the rigid body. In particular, the hybrid approach fitted with a Hypercomplex Multilayer Perceptron network, exhibits better results when considering trajectories not presented during the learning phase.

Key words: Feedback Control, Neural Networks, Radial Basis Functions.

1. INTRODUCTION

The attitude control of a rigid body in 3-D space consists in the computation of a control law that allows leading the object to a desired orientation and angular speed. Most of the applications arise from the control of robotics spacecrafts and satellites, but also the orientation of a rigid object held by robots, can be considered as an attitude control problem (Weng, *et al.* 1991), (Joshi, *et al.* 1995).

Among the methods for defining the rotation of a body between two different orientations, the rotation matrix is one of most commonly adopted. However this representation, that uses a 3×3 direction cosine matrix, requires a redundant number of parameters. In order to overcome this problem, without using the Euler angle representations that can be affected from singularities, a four-parameter description of the orientation, known as *quaternions*, can be used (Hamilton, 1969). In particular the kinematics and dynamic equations of a rigid body and the corresponding control equations can efficiently be written by using quaternion algebra (Chou, 1992), which could be considered as a generalisation of complex algebra. Moreover, recently, quaternion algebra applied to neural networks has allowed efficient learning algorithms to be developed (Arena, *et al.* 1997), (Nitta, 1993). To deal with the attitude control problem various quaternion-based control laws have been proposed (Weng, *et al.* 1991), (Fenton, *et al.* 1994).

In this work, a feed-forward controller, implemented by using an hypercomplex neural network, coupled with a quaternionic PD controller is considered. Two different kinds of hypercomplex neural networks have been used. The first one is a generalisation of the classical Multilayer Perceptron network (Rumelhart, *et al.* 1986), (HMLP: Hypercomplex MLP), while the other is an extension of the Radial Basis Functions network (Chen, *et al.* 1991) (HRBF: Hypercomplex RBF). Simulation results evaluate the performances of the proposed approaches. Moreover a comparison among the two proposed approaches and other control strategies as proposed in (Weng, *et al.* 1991) is displayed, in particular with:

- PD + HMLP
- PD + HRBF
- PD

- Adaptive controller

Simulations also evaluate the performance of the proposed hybrid approaches when an external disturbance acts on the system. This paper presents some fundamentals definitions and operators of quaternion algebra. In particular the use of quaternion algebra for representing rotations and writing efficiently the kinematics and dynamic equations is outlined. The controller structures proposed in (Weng, *et al.* 1991), particularly model-independent (PD) and model-dependent and adaptive control, that will be adopted for a comparison with the neural strategies, are summarised, while details are also given for the implementation of the proposed HMLP and HRBF networks. The remainder of this paper contains a comparative study of various control schemes.

2. QUATERNIONS AND ROBOTICS

2.1 *Quaternion Algebra*

W. Hamilton invented quaternion algebra (**H**) in 1843 in order to extend the properties of complex numbers in 3-D space (Hamilton, 1969). A *quaternion* $\mathbf{q}\in\mathbf{H}$ is defined as a generalised complex number, obtained by the direct sum of a real number and a vector, as represented in the following:

$$\mathbf{q} = q_0 + q_1\mathbf{i} + q_2\mathbf{j} + q_3\mathbf{k} \qquad (1)$$

where q_0, q_1, q_2, q_3 are real parameters and **i, j, k** are three unit vectors of an orthogonal frame in the space. The unit vectors **(i, j, k)** of quaternion algebra have the same role of the imaginary unit i in the complex algebra, that is:

$$\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = -1 \qquad (2)$$

Moreover, the following properties are satisfied:

$$\begin{aligned} \mathbf{jk} &= -\mathbf{kj} = \mathbf{i} \\ \mathbf{ki} &= -\mathbf{ik} = \mathbf{j} \\ \mathbf{ij} &= -\mathbf{ji} = \mathbf{k} \end{aligned} \qquad (3)$$

While complex algebra can represent rotations in the plane, quaternion algebra can represent rotations in 3-D space. The definition of Eq. (1) could also be written as follows:

$$\mathbf{q} = q_0 + \vec{q} \qquad (4)$$

where q_0 and $\vec{q}$ are called *scalar quaternion* and *vector quaternion* respectively.

In quaternion algebra, the fundamental operations can be easily defined. In particular the *quaternion product* is defined as:

$$\mathbf{q}\otimes\mathbf{p} = q_0 p_0 - \vec{p}\cdot\vec{q} + q_0\vec{p} + p_0\vec{q} + \vec{q}\times\vec{p} \qquad (5)$$

where '×' is the classical vector product operator and '•' is the classical scalar product operator and the inverse of a quaternion is defined as:

$$\mathbf{q}^{-1} = \frac{\mathbf{q}^*}{\mathbf{q}^*\otimes\mathbf{q}}$$

where $\mathbf{q}^* = q_0 - \vec{q}$ is the conjugate operator. Quaternion algebra is the only real, associative, divisional, but non-commutative algebra of dimension greater than two.

2.2 *Representation of Rotations*

Using quaternions in robotics, the rotations and the dynamic equations of a rigid body in the space are allowed to be represented efficiently (Chou, 1992). The quaternion **q**:

$$\mathbf{q} = q_0 + \vec{q} = \cos(\frac{\theta}{2}) + \mathbf{u}\sin(\frac{\theta}{2}) \qquad (6)$$

where $\mathbf{u}^T\mathbf{u} = 1$, represents a rotation of an angle θ around an axis **u** of a moving 3-D frame with ~~respect to~~ an inertial one. It has been proven that this ~~representation of rotations in the 3-D space~~ contains the minimum number of parameters that allows avoiding the singularities of attitude representation (Weng, *et al.* 1991).

Considering a quaternion α, the following operation:

$$\alpha' = \mathbf{q} \otimes \alpha \otimes \mathbf{q}^* \tag{7}$$

represents a transformation of α to α', which does not modify the scalar part of the quaternion but rotates its vectorial part of an angle θ around a rotation axis **u.** If α is a vector quaternion, *i.e.* a quaternion with zero real part then Eq.(7) expresses Euler's theorem, which states that:

*A generic rotation of a vector in the space can be described by a single rotation of an angle θ around an axis **u**.*

It is possible to determine also the inverse rotation, that is:

$$\alpha = \mathbf{q}^* \otimes \alpha' \otimes \mathbf{q} \tag{8}$$

A relationship exists between the quaternionic representation and the corresponding rotation matrix:

$$R = \begin{bmatrix} n_x & o_x & a_x \\ n_y & o_y & a_y \\ n_z & o_z & a_z \end{bmatrix} \tag{9}$$

where:

$n_x = q_0^2 + q_1^2 - q_2^2 - q_3^2$

$n_y = 2(q_1 q_2 - q_0 q_3)$

$n_z = 2(q_1 q_3 + q_0 q_2)$

$o_x = 2(q_1 q_2 + q_0 q_3)$

$o_y = q_0^2 - q_1^2 + q_2^2 - q_3^2$

$o_z = 2(q_2 q_3 - q_0 q_1)$

$a_x = 2(q_1 q_3 - q_0 q_2)$

$a_y = 2(q_2 q_3 + q_0 q_1)$

$a_z = q_0^2 - q_1^2 - q_2^2 + q_3^2$

These relations can be also expressed in a more compact form by the Rodriguez formula:

$$R = (q_0^2 - \bar{q}^T \bar{q})I + 2(\bar{q}\bar{q}^T + q_0 \tilde{q}) \tag{10}$$

where $\tilde{q}$ is the matrix:

$$\tilde{q} = \begin{bmatrix} 0 & -q_3 & q_2 \\ q_3 & 0 & -q_1 \\ -q_2 & q_1 & 0 \end{bmatrix} \tag{11}$$

This matrix is used to express the vector product as a product between the matrix $\tilde{q}$ and a vector, *i.e.*:

$$\bar{q} \times \bar{p} = \tilde{q}\bar{p} = -\tilde{p}\bar{q} \tag{12}$$

2.3 Kinematic and Dynamic Equations

The kinematic and dynamic equations representing the attitude of a rigid body in quaternion algebra are the following (Chou, 1992):

$$\begin{aligned} \dot{q}_0 &= -\frac{1}{2}\omega^T \bar{q} \\ \dot{\bar{q}} &= \frac{1}{2} q_0 \omega + \bar{q} \times \omega \end{aligned} \tag{13}$$

$$I\dot{\omega} = \tau - \omega \times I\omega \tag{14}$$

where $\mathbf{q} = q_0 + \vec{q}$ is the quaternion representing the orientation of the body with respect to an inertial frame, I is the matrix of principal moments of inertia of the body, τ and ω are vectors quaternions, representing the externally applied torque and the angular speed, respectively.

2.4 External Disturbance acting on the Spacecraft.

In practical situations a spacecraft is subject to environmental disturbances. The actions of these disturbances determine a deterioration of the control performances, and should not be neglected during the design phase of the control system.

Let consider a generic disturbance acting on the spacecraft due to an external source placed in the point **A** of the space, as shown in Fig. 1. The disturbance torque that will act on the controlled body, can be expressed by the following law:

$$\tau_D = k \frac{1}{[d(\mathbf{p}, \mathbf{A})]^2} (\mathbf{F} \times \mathbf{p}) \tag{15}$$

where k is a constant suitable chosen, **p** is a point of the spacecraft on which an external force $\mathbf{F}/[d(\mathbf{p}, \mathbf{A})]^2$ acts, where $d(\mathbf{p}, \mathbf{A})$ denotes the distance between the points **p** and **A**.

3. ATTITUDE CONTROL IN QUATERNION ALGEBRA

The attitude control of a spacecraft in 3-D space consists in finding a control law which allows to lead the body to the desired orientation ($\mathbf{q}_d$) and angular speed (ω_d) starting from any initial condition. To deal with this problem several control laws have been proposed (Weng, *et al.* 1991), (Joshi, *et al.* 1995). Three different control laws, reported in (Weng, *et al.* 1991), share the common structure of a proportional-derivative feedback controller plus some feed-forward action, which can be:

- *zero,*
- *a Coriolis torque compensation,*
- *an adaptive compensation*

In the first case the control law, independent by any model information, is given by:

$$\tau_{PD} = k_p \vec{q} - k_v \Delta\omega \tag{16}$$

where k_p and k_v are positive scalar gains, while $\vec{q}$ and $\Delta\omega$ represent respectively, the error between the actual and desired attitude and the error between the actual and desired angular speed. In particular $\vec{q}$ is the vector quaternion denoting the orientation of the desired attitude ($\mathbf{q}_d$) with respect to the fixed body reference frame ($\mathbf{q}_s$) and is computed as:

$$\mathbf{q} = \mathbf{q}_d \otimes \mathbf{q}_s^{-1} \tag{17}$$

while the speed error is defined as:

$$\Delta\omega = \omega_s - \omega_d \tag{18}$$

The proof of the global stability of this PD controller for a class of desired trajectories has been described in (Weng, *et al.* 1991).

Without any loss of generality it is assumed that the desired attitude corresponds to the zero attitude, therefore the body frame will tend to be aligned with the inertial frame, with zero final angular speed. One class of desired trajectories, for which this control law stabilises the rigid body, is given by:

$$\theta_d(t) = \theta_i \exp(-\alpha t^2) \tag{19}$$

$$\omega_d(t) = -2\alpha\theta_i \exp(-\alpha t^2) \cdot \hat{k} \tag{20}$$

Where θ_i is the initial rotation around the axis $\hat{k}$ and α is a positive gain (Weng, *et al.* 1991).

Due to the lack of model information, there is a significant nonzero transient tracking error in the classical PD controller. Better performance can be obtained including some model information in the control law. The modified control law as proposed in (Weng, *et al.* 1991) is:

$$\tau = k_p \bar{q} - k_v \Delta\omega + I \frac{d\omega_d}{dt} + z_1 \times I z_2 \tag{21}$$

where z_1 and z_2 can be ω_d , ω_s, and k_p and k_v are positive constants satisfying some stability constraints. In the control law of Eq. (21), the inertia matrix is assumed to be known, however this information is not always known or given. In this case the inertia matrix of the model can be replaced by an estimate that is updated by means of an adaptive scheme (Weng, *et al.* 1991). Hence, before defining the adaptive control law, some definitions are needed.
Given a symmetric matrix W, the following vector is defined:

$$v(W) = \begin{bmatrix} W_{11} & W_{12} & W_{13} & W_{22} & W_{23} & W_{33} \end{bmatrix}^T \tag{22}$$

Considering two arbitrary vectors, a and b, the function h can be defined as:

$$a^T W b = h^T(a,b) v(W) \tag{23}$$

Taking into account Eqs. (22), (23), the adaptive control law can be expressed as follows:

$$\frac{dv(\Delta I_s)}{dt} = -K_I \left[h\left(\Delta\omega_s - c \cdot q_s, \left(\frac{d\omega_d}{dt} \right) \right) - h\left(\omega_d \times (\Delta\omega_s - c \cdot q_s), \omega_d \right) \right] \tag{24}$$

where

$$\Delta I_s = I_s - I_d$$

I_d denotes the inertia matrix of the system while I_s denotes the estimated inertia matrix. K_I is a positive gain and c is a positive constant that needs to be sufficiently small to guarantee the stability of the system (Weng, *et al.* 1991). In the control law of Eq. (21), the inertia matrix I should be replaced with the estimate, determined by means of the law of Eq. (24).

4. THE QUATERNIONIC NEURAL CONTROLLER

A generalisation of the MLP (Rumelhart, *et al.* 1986), which uses quaternion algebra, is the Hypercomplex Multilayer Perceptron (HMLP) neural network has been defined in developed (Arena, *et al.* 1997), (Nitta, 1993). The HMLP is a MLP in which input/output values, weights and biases are quaternions instead of real numbers. The sigmoidal activation function of the hidden neurons is again a quaternion function defined as:

$$\Sigma(q) = \sigma(q_0) + \mathbf{i}\sigma(q_1) + \mathbf{j}\sigma(q_2) + \mathbf{k}\sigma(q_3) \tag{25a}$$

where $\sigma(\cdot)$ is the real sigmoidal function:

$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{25b}$$

As a generalisation of a classical results obtained for the MLP (Cybenko, 1989), it has been proven that an HMLP with the proposed activation function is able to approximate with the desired accuracy any nonlinear continuous quaternionic function. Moreover, it has been observed that the HMLP requires a lower number of real parameters, with respect to the corresponding MLP, thus improving the obtained performance (Arena, *et al.* 1997).

The first neural control strategy in this work is based on the PD controller, expressed by Eq. (10), coupled with a feedforward controller implemented by means of an HMLP network, as shown in Fig. 2.
The HMLP neural network used for control has three input neurons ($\mathbf{q}_s$, ω_s, $\dot{\omega}_s$), one hidden layer and one output neuron (τ_{ff}) and can be shown in Fig. 3.
The control law implemented by this network has the following form:

$$\tau_{ff} = f\left(q_s, \omega_s, \frac{d\omega_s}{dt}\right) \tag{26}$$

Hence, the whole control law is then:

$$\tau = \tau_{PD} + \tau_{ff} \tag{27}$$

The network has been trained using as error for the back propagation phase, the torque $\tau_{PD.}$ In particular the HMLP network has been trained according to the following steps:

1. *The HMLP weights and biases are randomly initialised,*
2. *A set of trajectories has been generated by means of the Eqs. (19), (20), starting with different initial conditions,*
3. *According to Fig. 2, the points of such reference trajectories are used a inputs for the neural network,*
4. *For each trajectory the control law of Eq. (27) is applied to the system,*
5. *At fixed time steps an earning cycle is performed, using τ_{PD} as error for the backpropagation phase of the network,*
6. *The steps 4 and 5 are repeated until a good tracking error is obtained on the whole set of trajectories.*

5. HRBF NEURAL NETWORK

HMLP neural networks with one hidden layer have increasingly been employed to realise complex nonlinear decision functions or to approximate nonlinear maps. However, these networks exhibit a drawback, since they are highly nonlinear in the parameters and the learning must be based on nonlinear optimisation techniques, such as the gradient descent algorithm. As a consequence, the training of such neural network requires a great number of learning cycles to achieve a good approximation of the presented data and obtain satisfactory results also for trajectories not presented to the network.

In many signal-processing applications the Radial Basis Function (RBF) neural networks, described in (Chen *et al.*, 1991), represent an alternative to the MLP neural networks. In this work, the RBF network is implemented using quaternion algebra, hence it can be called HRBF (Hypercomplex RBF) neural network. The HRBF network has been used as a feed-forward term for the control of the rigid body attitude, therefore the block diagram of the whole control scheme is the same as that one reported in Fig. 2, replacing the HMLP block with the HRBF one.

An RBF is a function that has an in-built distance criterion with respect to a centre. A typical RBF neural network consists of three layers (input, hidden, output). The activation of a hidden neuron is determined in two steps: the first involves computing the distance (usually by using the Euclidean norm) between the input vector and a center c_i which represents the i^{th} hidden neuron. Second, a function h (which is usually bell-shaped) is applied, using the previously distance to get the final activation of the hidden neuron. In this case, the well-known Gaussian function

$$G(x) = \exp\left(-\frac{x^2}{\sigma^2}\right) \tag{28}$$

was used. The parameter σ is called the "unit width", and is determined using the "*global first nearest-neighbour*", heuristic rule (Moody and Darken, 1989). It uses a uniform average width for all units, using the Euclidean distance in the input space between each unit m and its nearest neighbour n. All the widths in the network are fixed to the same value σ, and this results in a simpler training strategy.

The activation of a neuron in the output layer is determined by a linear combination of the fixed nonlinear basis functions, *i.e.*

$$F^*(x) = \sum_{i=1}^{M} w_i \phi_i(x) \tag{29}$$

where $\phi_i(x) = G(\|x - c_i\|)$ and w_i are the adjustable weights that link the output nodes with the appropriate hidden neurons. These weights in the output layer can then be learnt using the least-squares method.

The present study adopts a systematic approach to the problem of centre selection. Because a fixed centre corresponds to a given regressor in a linear regression model, the selection of RBF centres can be regarded as a problem of subset selection. The orthogonal least-squares (OLS) method can be employed as a forward selection procedure that constructs RBF networks in a rational way. The algorithm chooses appropriate RBF centres, one by one, from training data points until a satisfactory network is obtained. Each selected centre minimises the increment to the explained variance of the desired output, and the ill-conditioning problems that frequently occur in a random selection of centres can thereby automatically be avoided. In contrast to most learning algorithms, which can only work if a fixed network structure has first been specified, the OLS algorithm is a structural identification technique where the centres and estimates of the corresponding weights can be simultaneously determined in a very efficient manner during learning. The OLS learning procedure generally produces an RBF network that is smaller than that produced by a randomly selected RBF network (Chen *et al.*, 1991). Due to its linear computational procedure at the output layer, the RBF has a faster training time, compared to its backpropagation counterpart.

A major drawback of this method is associated with dimensionality of the input space. For large numbers of inputs units, the number of radial basis functions required, can become excessive. If too many centres are used, the large number of parameters available in the regression procedure will cause the network to be over-sensitive to the details of the particular training set, and will result in poor generalisation performance (overfit). To avoid this problem, a regularisation method has been proposed by Orr (Orr, 1993). Both the forward selection and zero-order regularisation techniques were proposed to construct parsimonious RBF networks with improved generalisation properties. Using the regularised forward selection algorithm (RFS), the error criterion for minimisation is given by the equation

$$J = e^T e + \lambda w^T w \tag{30}$$

instead of the standard $J = e^T e$, where λ is the regularisation parameter. However, the proposed algorithm does not utilise an orthogonalisation scheme; it is therefore computationally more expensive than the standard OLS algorithm. An efficient combination of the zero-order regularisation and the OLS algorithm has been proposed by Chen and Chng (1996). Similarly, the new error criterion for minimisation in the ROLS algorithm is

$$J = e^T e + \lambda\, g^T g \tag{31}$$

where g is the orthogonal weight vector which satisfies the triangular system

$$g = AW \tag{32}$$

and A is an upper triangular matrix which is computed directly from the OLS regression procedure (Chen *et al.*, 1991).

6. DISCUSSION OF RESULTS

In order to show the performance of the HMLP neural network, simulation results are shown. Following the example reported in (Weng, *et al.* 1991), the adopted inertia matrix is:

$$I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0.63 & 0 \\ 0 & 0 & 0.85 \end{bmatrix} \tag{33}$$

The PD controller gains are selected as:

$$k_p = 4$$

$$k_v = 8$$

The gain introduced in the Eqs. (19),(20) is α=0.04. The control action has been applied for T=10 *sec* for each trajectory and the learning cycles are performed every $\Delta T = 0.1s$.
An HMLP neural network with two hidden neurons has been trained on a set of trajectories, shaped as Eqs.(19) and (20), that differ for the initial angle and for the direction of the rotation axis. Several learning cycles have been performed on these trajectories. Moreover, as previously outlined, the system is subject to the external disturbances, shown by Eq. (15), which tend to move the spacecraft along a direction generally different from the desired one.

The simulations shown in this paper have been obtained with an HMLP trained on a set of 9 different trajectories. The initial angular speed was zero for each trajectory while the final desired attitude was:

$$q = (1,0,0,0)^T ; \omega = (0,0,0)^T$$

A comparison has been performed between the simple PD and the PD+HMLP controllers on different trajectories, presented and not during the learning phase.

Each plot reports, for a single trajectory, a comparison between the scalar parts of the error quaternion when a single PD controller and the proposed PD+HMLP controller are connected to the system. Figs. 4 and 5 show the first component of the quaternion representing the error between the actual and the desired trajectories when no disturbance acts on the spacecraft. The respective initial angle and axis are indicated on the figures.
For the above mentioned trajectories, an external disturbance acting on the system, expressed by the law of Eq. (15), has also been considered, and the obtained simulation results are reported in Fig. 6-7. The achieved improvement by the HMLP neural feed-forward term is remarkable.
Similar results have been obtained for several other trajectories, presented and not during the learning phase also when a disturbance acts on the system. The gain of the external disturbance was selected to be k=100000.

It can be observed that in the presence of disturbance the simple PD controller does not show good performance since the system structure has not been taken into consideration. On the other hand, the simulations emphasise the fact that the HMLP neural network approximates the desired trajectory better than the classical PD, whether the trajectory belongs to the training set or it has never presented to the network.

Moreover the use of the HMLP introduces a great computational efficiency with respect to the classical MLP, allowing reducing considerably both the number of parameters involved in the neural structure and the number of products needed during the forward phase. These results show that the HMLP network has learned well the map presented to it and is then capable to approximate also the trajectories not used during the learning phase (generalisation).

In order to compare the performance between the HMLP and the HRBF controllers the same simulations have been performed with the HRBF network.
Similarly, to the previous case, the actual orientation, angular speed and acceleration compose the input vector of the HRBF network. The proposed "*global first nearest-neighbour*" method for width definition was found in practice to be inadequate for the problem. A rather heuristic method, taking half the maximum Euclidean distance, was finally adopted for the simulations. Although an elegant approach to the selection of the regularisation parameter λ is to adopt Bayesian techniques, in this work this parameter was set by trial and error to small positive values, which satisfy the optimal problem solution. The λ parameter was set equal to 0.0018. For each trajectory of the training set, 14 centres have been selected using the ROLS algorithm.

After the selection of the RBF centres, the parameters of the network and then the torque τ_{rbf} has been computed. The RBF torque, added to the PD one, allows controlling the attitude of the spacecraft. The HRBF neural network has been trained on different training trajectories, the same previously used for the learning of the HMLP network, and also the test trajectories are the same adopted when using the PD+HMLP controller.
In the same sense, also for the HRBF network an external disturbance, expressed by the law of Eq.(15), with a gain k=100000 that can act on the spacecraft, has been considered.

In the Fig.8-11, a comparison between the scalar parts of error quaternions when the proposed PD+HRBF controller and the single PD controller are connected to the system is shown. In the Figs. 8-9 the disturbance is not considered, while in the Figs.10-11 it acts on the system.
Both for the PD+HMLP and PD+HRBF neural networks, results similar to those shown in the figures above reported have been obtained for many other trajectories, presented and not during the learning phase an when a disturbance acts or not on the spacecraft.
As can be observed from Figs. 8-11, the HRBF network has good capabilities of interpolation for the whole set of trajectories presented during the learning, improving the results which are obtained controlling the system with the single PD. However, in the contrary of the HMLP network, it has not good extrapolation capabilities when using a trajectory not presented previously.

The mean squares error obtained for the different trajectories, presented and not during the learning phase of the neural networks (HMLP and HRBF) have been compared for the external disturbance acting or not on the spacecraft. The results are reported in Tables I and II. Table I contain the results obtained for 9 trajectories belonging to the training set (1-9) and for 2 trajectories not presented previously to the neural networks (10-11). The results obtained for the same trajectories when the spacecraft is subject to a disturbance are reported in Table II.
In total, four different control strategies have been addressed for this issue:

- *PD+HMLP*
- *PD+HRBF*
- *PD*
- *Adaptive controller*

In the trajectories presented, the single PD controller achieves the worst performance, while the adaptive controller gives better results than those of the PD do.
This is due to the fact that such controller determines an estimate of the system inertia matrix. Better results are obtained using the HRBF and HMLP neural networks. In particular, as it can be observed in this case, the HRBF networks gives the best performance.
For the two trajectories not used for the learning of the HMLP and HRBF networks, it can be seen that the HMLP assures good performance, similar to those obtained when the desired trajectory belongs to the training set. On the other hand, the HRBF network gives the worst results also with respect to the PD and the adaptive controllers.

In table II the results for the case of external disturbance acting on the spacecraft are shown. It can be observed that PD+HMLP and PD+HRBF controllers give also in this case the best performance, despite the presence of the disturbance. It results that the HRBF network achieves the best performance. As it can be observed (from those trajectories don't belong to the training set), the HRBF network fails to approximate the desired trajectory in a satisfactory way, while the HMLP network assures good performance as in the previous case. The presence of the disturbance affects considerably the results of the PD and the adaptive controllers.

7. CONCLUSIONS

In this study, the attitude control problem of a rigid body, such as a spacecraft, using quaternion algebra to represent control and dynamic equations, has been considered. This algebra allowed the computational complexity of the equations to be reduced and has simplified the attitude control in 3-D space. To improve the control of the rigid body with respect to the simple PD controller, a new control strategy has been implemented. The control strategy proposed in this paper uses two parallel controllers:

- A feedback PD controller,
- A feed-forward controller implemented by means of an HMLP neural network.

The feed-forward controller improves the performances of the simple PD controller allowing to better compensating the nonlinearities of the model without compromising the closed loop stability.

The HMLP neural network has been trained on different sets of trajectories and an external disturbance that can act on the spacecraft has been considered.

The second proposed strategy use the HRBF neural network as feed-forward term in alternative to the HMLP network. In the same way, this neural network has been trained on various sets of trajectories, the same considered for the HMLP network, both in the cases when a disturbance acts or not on the rigid body.

The simulation results obtained using the two neural networks have been compared with those obtained when a classical PD controller or an adaptive controller is used. Simulations have shown that the HRBF neural networks assure good performance for trajectories presented during the learning phase but not for the others, concluding that they have inferior extrapolation capabilities. These results have been obtained both in presence or absence of external disturbance acting on the system. On the other side, the HMLP based controller assures good results for each trajectory, presented or not during the learning of the network, even if a smaller degree of accuracy than that obtained with the HRBF networks is reached, for the trajectories used for the learning. In both cases the obtained results show an improvement with respect to the simple PD and the adaptive controllers.

REFERENCES

1. Arena, P., Fortuna, L., Re, R., Xibilia, M., Multilayer Perceptron to Approximate Quaternion values Functions. *Neural Networks*, Vol. 10, No. 2, 1997, pp. 335-342.
2. Chen, S., Cowan, C.F., and Grant, P., Orthogonal least-squares learning algorithm for radial basis function networks. *IEEE Trans. on Neural Networks*, 1991, **2**, 302-309.
3. Chen, S. and Chng, E., Regularised orthogonal least squares algorithm for constructing radial basis function networks. *Int. J. Control,* 1996, **64**(5), 829-837.
4. Chou, J., Quaternion kinematic and dynamic differential equations. IEEE Trans on Robotics and Automation, Vol. 8, No.1, 1992, pp.53-64.
5. Cybenko, G., Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 1989, **2**, 303-314.
6. Fenton, R., Xi, F., Computational analysis of robot kinematics, dynamics and control using the algebra of rotation. *IEEE Trans on SMC*, Vol. 24, No. 6, 1994, pp.942-956.
7. Hamilton, W., *Elements of Quaternions*. Chelsea, 1969.
8. Joshi, S., Kelkar, A., Robust Attitude Stabilization of Spacecraft using nonlinear quaternion Feedback. *IEEE Trans on Automatic Control*, Vol. 40, No. 10, 1995.
9. Moody, J. and Darken, C., Fast learning in networks of locally tuned processing units. *Neural Computation*, 1989, **1**, 281-294.
10. Nitta, T., A backpropagation algorithm for neural networks based on 3D vector product. *Proc. of Int. Joint Conference on neural networks*, 1993, pp. 589-592.
11. Orr, M., *Regularised centre recruitment in radial basis function networks*. Research report No 59, Centre for Cognitive Science, University of Edinburgh, UK, 1993.
12. Rumelhart, D. and McClelland, T. L., (eds), *Parallel Distributed processing: Explorations in the Microstructure of Cognition. Vol. 1: Foundations*, MIT Press, 1986.
13. Weng, J, Kreutz-Delgado, K., The attitude control problem. *IEEE Trans on Automatic Control*, Vol. 36, No. 10, 1991, pp.1148-1162.

No. Traj.	Init. angle (rad)	PD	HMLP+PD	HRBF+PD	Adapt.
1	1.4508	0.0030	6.3689e-5	8.0055e-6	0.0021
2	0.6126	4.3865e-4	1.4665e-4	1.0378e-6	4.2843e-4
3	1.6920	0.0030	1.0035e-4	6.1151e-6	0.0024
4	0.8308	9.2385e-4	2.6081e-5	2.5259e-6	8.1786e-4
5	0.3567	1.4098e-4	4.1878e-5	3.0993e-7	1.4082e-4
6	0.0661	5.6160e-6	2.1688e-7	1.4134e-8	5.6144e-6
7	1.4966	0.0026	8.3023e-5	5.9120e-6	0.0021
8	1.6841	0.0036	1.4101e-4	8.7385e-6	0.0027
9	0.2609	5.5186e-5	4.0487e-6	9.6114e-8	5.5057e-5
10	0.1337	1.9618e-5	3.4629e-7	4.9161e-5	1.9597e-5
11	2.4648	0.0091	7.0711e-4	0.0091	0.0043

Table I

No. Traj.	Init. angle (rad)	PD	HMLP+PD	HRBF+PD	Adapt.
1	1.4508	0.5583	2.9890e-5	3.8781e-5	0.5715
2	0.6126	0.7449	0.0130	3.0143e-5	0.7439
3	1.6920	0.6972	0.0033	2.7620e-5	0.7054
4	0.8308	0.6876	0.0031	4.2560e-5	0.6877
5	0.3567	0.7141	0.0037	3.0012e-5	0.7139
6	0.0661	0.6573	1.8969e-4	3.6097e-5	0.6573
7	1.4966	0.8501	7.0685e-4	3.7872e-5	0.8568
8	1.6841	0.9393	0.0027	3.2474e-5	0.9171
9	0.2609	0.5995	9.4895e-4	4.0733e-5	0.5996
10	0.1337	0.6555	2.9197e-4	2.2412e-5	0.6555
11	2.4648	1.1388	0.0507	0.7791	1.1125

Table II

CAPTIONS LIST

1. Spacecraft, represented by the cube, is subject to an external disturbance whose source is places in the point $\boldsymbol{A}$.
2. Block diagram of the proposed control scheme.
3. HMLP neural network with only one hidden layer.
4. Comparison of the scalar parts of the error quaternions obtained using the PD controller and the PD+HMLP controller for a trajectory presented during the learning phase. No disturbance acts on the spacecraft.
5. Comparison of the scalar parts of the error quaternions obtained using the PD controller and the PD+HMLP controller for a trajectory not presented during the learning phase. No disturbance acts on the spacecraft.
6. Comparison of the scalar parts of the error quaternions obtained using the PD controller and the PD+HMLP controller for a trajectory presented during the learning phase. A disturbance acts on the spacecraft.
7. Comparison of the scalar parts of the error quaternions obtained using the PD controller and the PD+HMLP controller for a trajectory not presented during the learning phase. A disturbance acts on the spacecraft.
8. Comparison of the scalar parts of the error quaternions obtained using the PD controller and the PD+HRBF controller for a trajectory presented during the learning phase. No disturbance acts on the spacecraft.
9. Comparison of the scalar parts of the error quaternions obtained using the PD controller and the PD+HRBF controller for a trajectory not presented during the learning phase. No disturbance acts on the spacecraft.
10. Comparison of the scalar parts of the error quaternions obtained using the PD controller and the PD+HRBF controller for a trajectory presented during the learning phase. A disturbance acts on the spacecraft.
11. Comparison of the scalar parts of the error quaternions obtained using the PD controller and the PD+HRBF controller for a trajectory not presented during the learning phase. A disturbance acts on the spacecraft.

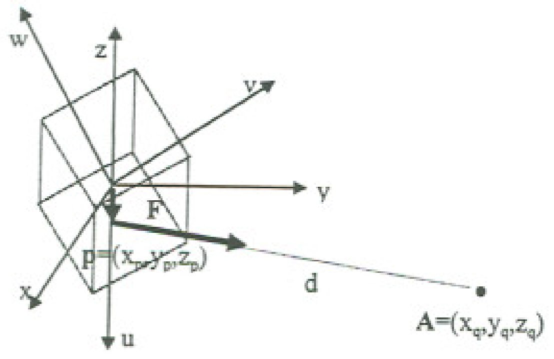

Figure 1.

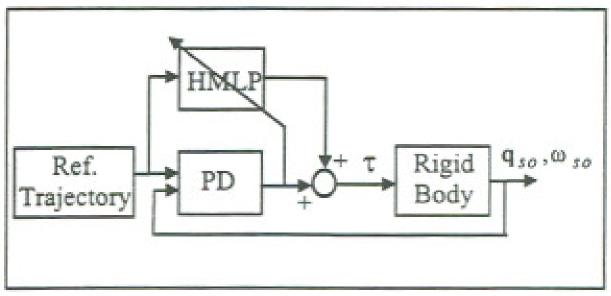

Figure 2.

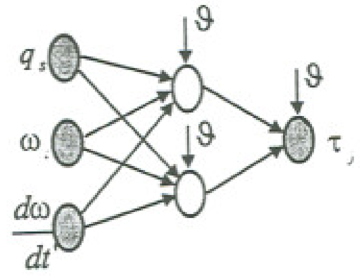

Figure 3.

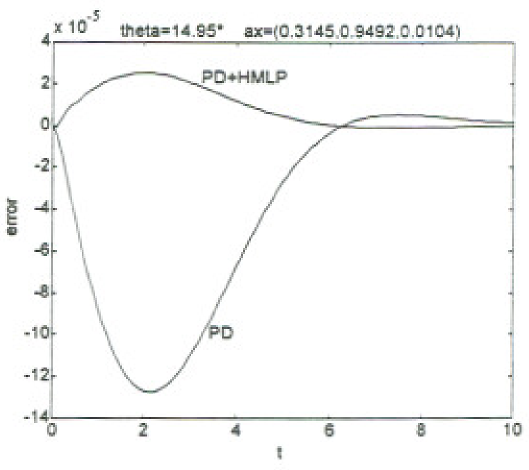

Figure 4.

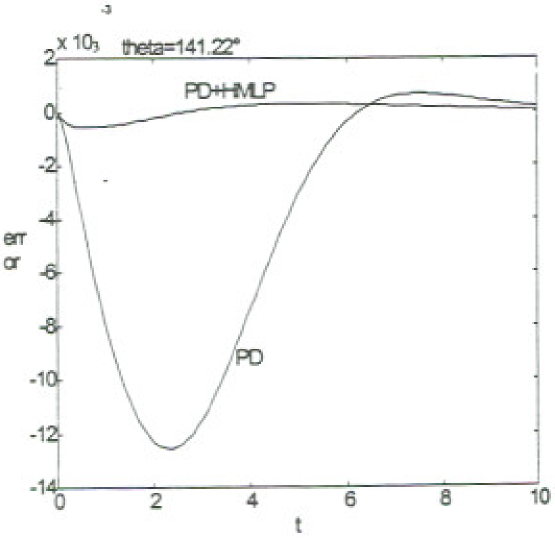

Figure 5.

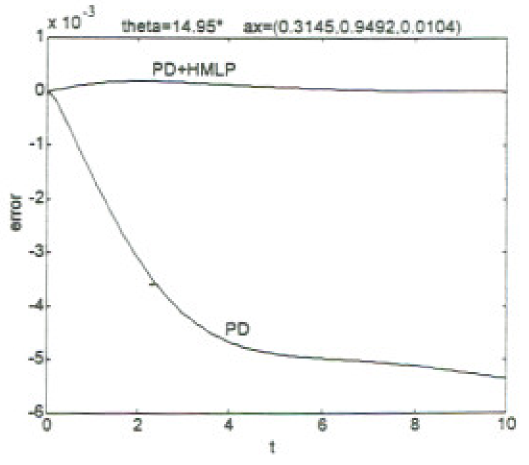

Figure 6.

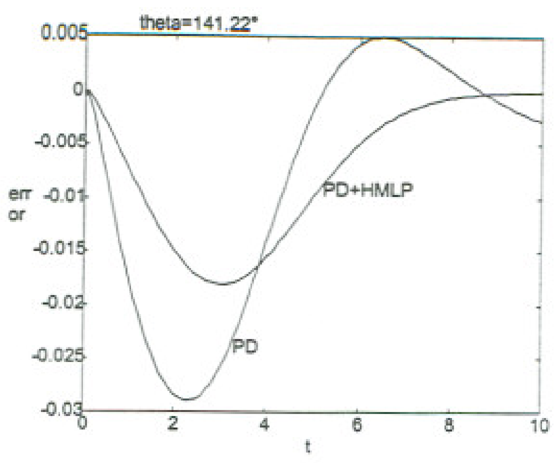

Figure 7.

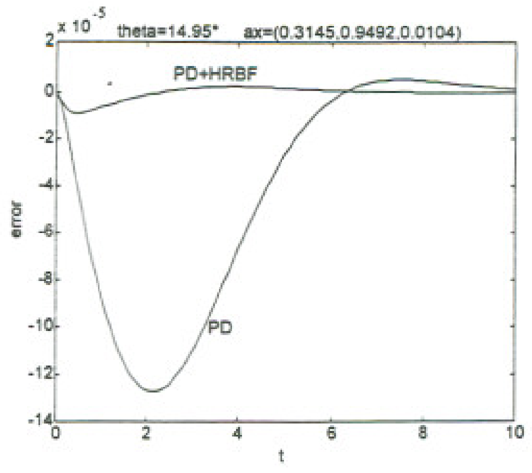

Figure 8.

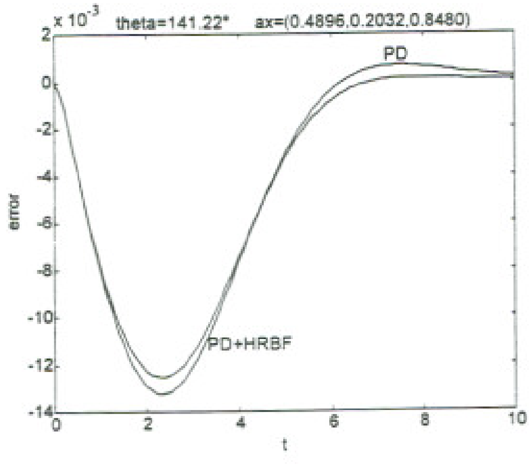

Figure 9.

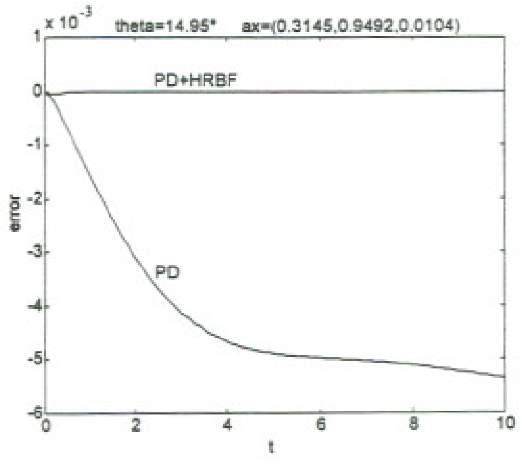

Figure 10.

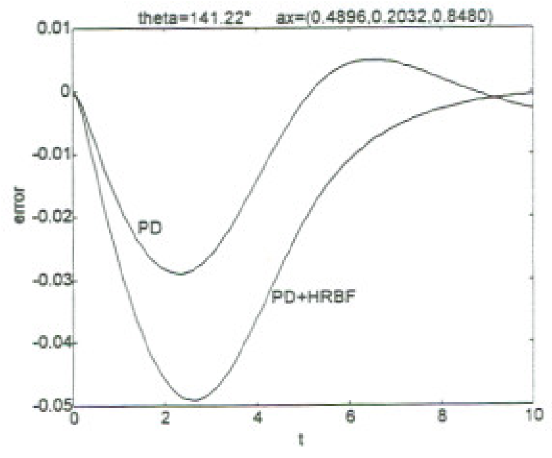

Figure 11.