NEURAL CONTROL OF HELICOPTER FLIGHT

C Saade and F Zinsch

Department of Aeronautics and Astronautics, University of Southampton, UK

and

ENSICA Toulouse, France

Abstract

The work reported here relates to the design of a flight control system which provides all-axis stability for a single main rotor helicopter and provides good robustness to sensor or control failure. The work was based upon a simulation of a Bell UH-1H helicopter in forward flight and at hover. Initially, a feedback controller was designed for each flight condition using an optimal feedback controller. A linear quadratic regulator formulation was used where the weighting matrices of the performance index were evaluated in accordance with the specified bandwidths of the vehicle.

The paper provides a brief discussion of Neural Network, with particular emphasis on the architecture and structure of two possible candidate networks viz the Multi-Layered Perceptron with back-propagation, and the Radial Basis Function. Training times, appropriate structures, and network error convergence are important considerations and the paper presents results and findings relating to these factors. The Neural Network which was finally chosen provided, after appropriate training, results which were indistinguishable from the results obtained from a continuous optimal feedback controller. The behaviour of the control systems was assessed when subject to motion sensor failures of several different types. By adding another failure-correcting neural controller, it was found that the same dynamic performance could still be achieved even in the presence of feedback signal failure.

Nomenclature

A	State matrix
B	Control matrix
C	Output matrix
G	Control weighting matrix
a_z	Normal acceleration (m/s^2)
h_E	height error (m)
p	Roll rate, (rad/s)
p_r	Roll rate of the rotor, (rad/s)
q	Pitch rate, (rad/s)
q_r	Pitch rate of the rotor, (rad/s)
r	Yaw rate, (rad/s)
u	Forward speed, (m/s)
v	Lateral speed (m/s)
w	Vertical speed (m/s)
$\dot{x}$	Ground speed error (m/s)
x_{bar}	Stabilizer bar deflection (rad)
$\underline{u}$	Control vector of dimension p
x	State vector of dimension n
$\underline{y}$	Output vector of dimension m
δ_c	Collective pitch control (rad)
δ_ϕ	Lateral cyclic pitch control (rad)
δ_θ	Longitudinal cyclic pitch control (rad)
δ_ψ	Tail rotor collective pitch control (rad)
ϕ	Bank angle (rad)
γ	Flight path angle (rad)
θ	Pitch attitude (rad)
θ_r	Pitch attitude of the rotor (rad)
ζ	Damping ratio
ψ	Yaw angle (rad)
ω	Frequency (rad/s)
ω_c	Bandwidth requirement (rad/s)

Introduction

Helicopters require the use of a flight control system (FCS) to achieve satisfactory flight. It is believed that the use of neural computing, with its potential for providing a controller which can learn and rapidly accommodate to the wide variety of changes which arise in any helicopter flight,will provide, an effective means of controlling the helicopter. This paper is designed to confirm that belief, by achieving the design of a neural controller for a helicopter FCS, and by studying how reliable and robust is the performance of a neurally-controlled helicopter to system and sensor failures.

The first part of the paper will be concerned with the design of a baseline controller using the linear quadratic regulator method Such a system was necessary to provide training data for the determination of a neural controller studied in the second part.

Helicopter Dynamics

A digital simulation of a Bell UH-1H helicopter for both hovering and forward flight was carried out. The mathematical model used was based on Ref 1. The aircraft was considered initially as a single rigid body with instantaneous rotor tilting. To obtain the equations of motion, a body axis system was chosen for analytical convenience.

Study of the helicopter at hover.

The first mode of flight considered was hovering flight, since much of the total flight time can be spent at hover. Because there are significant cross-coupling effects between the longitudinal and lateral modes at hover, both motions were considered simultaneously in the simulation.

Using stability axes, the state vector was defined as :

$$\underline{x}' \triangleq [u \;\; w \;\; q \;\; \theta \;\; v \;\; p \;\; r \;\; \phi] \qquad (1)$$

where ' denotes the transpose operation.

The control vector comprised the four command inputs needed to control any helicopter flight viz.

$$\underline{u}' \triangleq [\delta_C \;\; \delta_\theta \;\; \delta\phi \;\; \delta\psi] \qquad (2)$$

The equations of motions were expressed in canonical form as:

$$\dot{\underline{x}} \triangleq A\underline{x} + B\underline{u} \qquad (3)$$

A is the coefficient matrix of order [8 x 8] and B is the driving matrix of order [8 x 4]. These matrices are given in annex 1. The elements of these matrices comprise the stability and control derivative evaluated for small deviations from trim conditions (Ref 2).

To assess the helicopter's stability properties, the eigenvalues of the matrix A were determined.

$$\begin{aligned}
&\lambda_1 = -1.2433 \\
&\lambda_2 = -0.4018 \\
&\lambda_3, \lambda_4 = 0.20007 \pm 0.8294j \\
&\lambda_5, \lambda_6 = 0.1448 \pm 0.4324j \\
&\lambda_7, \lambda_8 = -0.5946 \pm 0.3436i
\end{aligned} \qquad (4)$$

Note that the helicopter has eigenvalues with positive real parts and, therefore, is dynamically unstable.

Forward flight.

The study was restricted here solely to longitudinal motion, but the mathematical model included three additional state variables:

- the height error, h_E
- the ground speed error, $\dot{x}$
- the horizontal stabilizer bar angular displacement, x_{bar}.

The resulting state vector was then expressed as:

$$\underline{x}' \triangleq [u \; w \; q \; \theta \; \delta_\theta \; \delta_c \; h_E \; x_{bar} \; \int \dot{x} dt] \qquad (5)$$

The control vector corresponded only to longitudinal commands. However, since controlling the rates of change was as important as controlling the changes themselves, the control vector was altered to:

$$\underline{u}' \triangleq [\delta_\theta \;\; \delta_c] \qquad (6)$$

Equation 3 was still used to represent the expanded longitudinal equations of motion,but the corresponding matrices, A and B, were as shown in annex 2.

For the hovering case, the stability and control derivatives were taken from (Ref 2) and the eigenvalues of the uncontrolled helicopter were now found to be:

$$\begin{aligned}
&\lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = 0 \\
&\lambda_5 = -0.4254 \\
&\lambda_6, \lambda_7 = -0.7781 \pm 1.1910j \\
&\lambda_8, \lambda_9 = 0.0112 \pm 0.2492j
\end{aligned} \qquad (7)$$

Inclusion of rotor dynamics.

The rotor dynamics were also taken into consideration to represent the helicopter more realistically. A two-rigid-body model was considered next. The fuselage was regarded as one of the rigid bodies and the spinning rotor as the other; the rotor could be tilted with respect to the fuselage (Ref 3).

This inclusion of rotor dynamics resulted in the addition of new state variables related to the rotor motion.

At hover, the state vector now became

$$\underline{x}'_r \triangleq [\theta_r \; \phi_r \; q_r \; p_r \; u \; w \; q \; \theta \; v \; p \; r \; \phi] \qquad (8)$$

whereas in forward flight:

$$\underline{x}'_r \triangleq [q_r \; \theta_r \; u \; w \; q \; \theta \; \delta_\theta \; \delta_c \; xbar \; h_E \; \int \dot{x} dt] \qquad (9)$$

where θ_r, ϕ_r, q_r and p_r denote respectively the pitch angle of the rotor, the roll tilt angle of the rotor, the rate of change of pitch angle and the rate of change of roll tilt angle.

The coefficient and control matrices, A_{rotor} and B_{rotor}, corresponding to eq (3), are given in Annex 3.
If

$$A_{rotor} \triangleq \begin{bmatrix} A_1 & A_2 \\ A_3 & A_4 \end{bmatrix} \qquad (10)$$

Then the sub-matrix A1 represents the rotor dynamics, sub-matrix A2 represents the fuselage coupling effects, sub-matrix A3 represents the rotor coupling effects and sub-matrix A4 represents the fuselage dynamics.

Inclusion of actuator dynamics.

Actuator systems have their own dynamic characteristics which may affect the basic aircraft dynamics.

For the hover flight condition, two different actuator dynamics models were considered. It was considered that, for the tail rotor, the dynamic response of the actuator was so rapid, that the dynamics could be regarded as instantaneous. Hence, the transfer function of eq (11) was used viz:

$$\frac{\delta_p(s)}{\delta_{pcomm}(s)} = K_1 \qquad (11)$$

For the other control surfaces, the actuator dynamics were modelled by the following transfer function:

$$\frac{\delta(s)}{\delta_{comm}(s)} = \frac{K_2\lambda}{(s+\lambda)} \qquad (12)$$

In forward flight, the dynamics of both control surface actuators were considered to be:

$$\frac{\delta(s)}{\delta_{comm}(s)} = \frac{K_3 \cdot \omega^2}{s^2 + 2\zeta\omega s + \omega^2} \qquad (13)$$

The existence of non-linearities was another feature of the actuator which had to be accounted for.

One of these non-linearities is caused by the fact that control surfaces cannot be moved faster than a maximum rate. So a maximum rate of change for the deflection had to be introduced into the dynamic representation.

Another non-linearity was a threshold value to represent the fact that, below a certain value of input, the actuator does not respond.

Figure 1 represents a block diagram of such an actuator.

The results of these simulations showed the instability of the basic helicopter since every real eigenvalue and every real part of any complex eigenvalue must be negative for complete dynamic stability. Hence, a feedback controller had to be designed to stabilize the helicopter's flight. The investigation of the actuators' dynamics showed that the dynamic performance of the baseline controller was impaired when those dynamics were included. Moreover, non-linearities, such as threshold or rate limiting, also impaired the resulting closed-loop performance. However, including the rotor dynamics resulted in a noticeable increase in the helicopter's stability.

Optimal Linear Control

The purpose of a feedback control system is to alter the dynamic behaviour of a physical process so that its controlled response more nearly corresponds with the user's requirements. Many methods can be used to obtain a linear control law (Ref 4); in this work, the feedback control system was designed using linear optimal control theory because it guarantees closed-loop stability and is robust. The problem was to determine an optimal control function, $u^o(t)$, which minimizes a performance index, J, given by:

$$J = \frac{1}{2}\int_0^\infty (\underline{y}'Q\underline{y} + \underline{u}'G\underline{u})\,dt \qquad (14)$$

where $\underline{y}$ is the output vector ($\underline{y} = C\underline{x}$)[1], Q is the state weighting matrix and G the control weighting matrix.

These weighting matrices were calculated according to a major feature of aircraft modelling ie the bandwidths associated with the aircraft, since these are significant in the study of the stability and control of any aircraft.

An iterative procedure was used to determine Q and G as detailed below.

First the G matrix was set to accommodate the limits imposed on control deflections. Then the bandwidth requirements were presented in

[1] See equations (21) and (22)

descending order and the highest one not yet addressed was identified from Table 1.

Table 1 Bandwidth requirements

Forward Flight		Hover	
State variable	Bandwidth (rad/s)	State variable	Bandwidth (rad/s)
δ_θ	25.7	w	4.0
δ_C	25.7	p	2.5
θ	1.5	q	2.0
$\dot{h}_E$	1.0	ϕ	1.8
h_E	0.82	v	1.3
x	0.5	u	1.0
$\int x dt$	0.1	r	0.5
		θ	0.4

The next step was to define and approximate the transfer function, whose output was the variable subject to the bandwidth requirement and whose input wasselected from the controls and all those variables satisfying prior bandwidth requirements. The form of approximation was

$$\frac{X}{s^{n+1}}$$

The quantity

$$(2^n \omega_c^2)^{n+1} \frac{r}{X^2}$$

was determined for each transfer function and the lowest value was chosen to be the cost function weighting coefficient.

This sequence of steps was repeated until all bandwidth requirements were met.

The final results were:
(a) at hover
G = diag (1 2.25 10 100) (15)
Q = diag (0.001 16 0.001 0.001 0.001 0.001 0.001 0.001) (16)
resulting in a feedback gain matrix:

$$K = \begin{bmatrix} 0.0978 & -3.6560 & -0.4059 & 0.0713 & 0.0117 & 0.1120. & -0.3952 & -0.5818 \\ 0.1100 & 0.0168 & -7.9678 & -4.6411 & -0.1297 & 0.7982 & -0.0441 & 0.9697 \\ 0.0360 & -0.1099 & -0.3894 & 0.7436 & -0.0178 & 2.6965 & 0.1296 & 3.1243 \\ 0.0002 & -0.0108 & -0.0226 & 0.0455 & 0.0019 & 0.1710 & -0.0012 & 0.2450 \end{bmatrix} \quad (17)$$

(b) in forward flight
G = diag (1 10) (18)
Q = diag (0 0 355000 662 6620 477 46.6 357 3.57 0) (19)
and

$$K = \begin{bmatrix} 15.3 & 16 & -485 & -1846 & 27.5 & -0.6 & 4.4 & 54.4\ 1.4 \\ 6.9 & -2.1 & 11.7 & 46.7 & -0.1 & 25.9 & -1.6 & 17.5\ 0.4 \end{bmatrix} \quad (20)$$

The corresponding eigenvalues of the closed loop systems were found to be:

Table 2: Closed-loop Eigenvalues

Forward Flight	Hover
λ_1 = -26.2629	λ_1 = -4.0720
λ_2 = -25.8552	λ_2 = -1.0262
λ_3 = -1.3943	λ_3, λ_4 = -0.6643 ± 0.1j
λ_4, λ_5 = -0.8546 ± 1.5586j	λ_5, λ_6 = -0.2237 ± 0.8257j
λ_6, λ_7 = -0.2942 ± 0.0580j	λ_7, λ_8 = -0.1312 ± 0.4284j
λ_8, λ_9 = -0.0746 ± 0.0471j	

Note that these closed-loop systems were now dynamically stable (see Figure 2). Indeed, these stabilized systems were next used to provide data corresponding to the controlled aircraft's behaviour in response to several inputs. These data were stored, with the output vectors being chosen to be:
At hover:

$$\underline{y}' \triangleq [u \ w \ q \ \theta \ v \ p \ r \ \phi] \quad (21)$$

In forward flight:

$$\underline{y}' \triangleq [u \ w \ q \ \theta \ \delta_\theta \ \delta_C \ h_E \ xbar \int \dot{x}.dt \ \gamma \ a_{z_{cg}}] \quad (22)$$

These stored data sets were used to train the Neural Network controller.

Artificial Neural Networks

Neural Networks are information processing systems, which can be regarded as "black box" devices which accept inputs and produce outputs (Ref 5). An ANN consists of layers of processing elements (PEs) and weighted connections. Each layer in a neural network consists of a collection of PEs. Each PE collects the values from all of its input connections, performs a pre-defined mathematical operation (typically a dot-product followed by a threshold function), and produces a single output value.

Using an information process analogous to that used by the human brain, a major feature of an ANN is their ability to learn to perform some particular function by adjusting the values of the connections between elements. This training precedes the implementation phase where the network is finally used.

The design of any ANN starts with a selection of input and output patterns. Then, the selection of an appropriate learning algorithm is undertaken, followed by the determination of the corresponding architecture (ie the number of layers, the number of hidden neurons and the choice of threshold functions, etc).

Many networks are available for use. The two most suitable networks for this problem were considered to be the Multi-Layered Perceptron with Back-Propagation (referred to as BP) and Radial Basis Networks.

Multi-Layered Perceptron with Back-Propagation Network.

The reason for considering BP was that such networks tend to give good results when presented with unknown inputs.

Moreover, the architecture of such BP networks is partly chosen by the designer. The process for training the weights is the following:

- a set of inputs is applied to the network
- the outputs are calculated and compared to the desired outputs
- the error is first calculated for the final layer and is propagated back through the network
- preceding connection weights are then adjusted proportional to the error in the unit to which it is connected
- and the process is repeated for as many layers as are present until the error reduces to a suitable level.

For the neuron model, the most commonly used threshold functions are the Log-sigmoid, the Tan-sigmoid, and the Linear Function.

For the training sequence, several algorithms were available for use in BP networks. Those chosen for this study were:

- Standard BP with a constant learning rate.
- BP with momentum and an adaptive learning rate.
- BP with Levenberg-Marquardt optimization.

Radial Basis Network

Radial Basis (RB) Networks are alternatives to standard feed-forward BP networks (Ref 6). Although they may require more neurons, they can often be designed in a fraction of the time it takes to train standard BP networks.

Only one possible architecture can be used in RB networks: a hidden layer of RB neurons and an output layer of linear neurons.

Two algorithms are available for the training of the network. The first introduces as many hidden neurons as there are training inputs, while the second uses an iterative method, adding one hidden neuron at a time until the error goal, or the maximum number of neurons, is reached.

Artificial Neural Network Control

The two networks which were considered to obtain the helicopter neural control were: the Multi Layered Perceptron with back-propagation and the Radial Basis function.

Multi Layered Perceptron with Back-Propagation Design

In this case, the most convenient algorithm was determined first.

To achieve this, an initial architecture was selected and retained as reference to provide comparisons between the three candidate algorithms. The initial architecture consisted of a two-layer network, with five hidden tan-sigmoid neurons and a linear output layer since

the output values had to go over the range of real numbers.

The first algorithm considered was the standard BP which used a constant learning rate: its role was to act as a gain on any weight and bias changes. The principal reason for rejection of this algorithm was related to the choice of a convenient learning rate. When the learning rate was too small, it resulted in exceedingly long training times, whereas too large a learning rate led to unstable learning with increasing errors.

This problem was solved by considering the BP algorithm with a momentum term and an adaptive learning rate (BP+) (Ref 7). The process used by this algorithm attempted to keep the learning step size as large as possible, while keeping the learning process stable. Thus, the BP+ controller gave acceptable results, but the training time remained so long that consideration of a third algorithm was required.

To overcome the excessive training times, the BP algorithm with Levenberg-Marquardt optimization (BP++) was used and was found to be the most convenient for the work (see Figure3). The optimization technique (Ref 8) used with BP++ is more sophisticated than the gradient descent method used with the BP+.

Once the most efficient algorithm for the work had been established, the architecture was considered. Starting with the architecture taken as a reference, the different available parameters were considered one at a a time. The first parameter was the number of hidden neurons. Too few neurons could lead to and unsatisfactory error minimum, resulting in a system of insufficient intelligence.

However, increasing the number of neurons introduced longer training times. Thus, it became the objective to find the smallest number of hidden neurons which gave an acceptable value of error minimum. Finally, the number of five hidden neurons was selected.

Next, the choice of the transfer function of the hidden layer had to be made. The three candidates were the linear, tan-sigmoid, and log-sigmoid transfer function. The most efficient, for both flight modes,was found to be the linear function.

Finally, the number of layers was considered. Since three layers was the largest size allowed by the software use, all the possible architectures were considered. The work concerning two layer networks has been detailed above. For three layer networks a similar study was undertaken, whereas the architecture of a single layer network was constrained by the problem.

Finally, the network chosen was a single linear layer network trained using the BP++ algorithm. Comparisons between the responses obtained using the neural controller and the optimal continuous controller are shown in Figure 4.

Radial Basis Design

The time required to design a Radial Basis Network was much shorter than that required for a BP network because only one main parameter had to be fixed: the spread constant. This value determines the width of an area in the input space to which each neuron responds.

However, two different algorithms were available for the training.

The first created RB networks with as many hidden RB neurons as there were input vectors in the training data. For both flight modes, the number of training inputs needed for a satisfactory model was very large; this method required too much memory. The aim of the second method was to find the smallest network which can solve the problem within a given error goal. To avoid an "out of memory" problem, a maximum number of RB neurons was set (200) which actually proved to be too small to solve the problem (see Figure 5).

Because of these unsatisfactory results, the RB network was rejected for this application.

Application to Sensor Failures

Sensor failure is a common problem over the life time of any helicopter (Ref 9). In this section of the paper a neural-network-based approach is presented for solving the problem of sensor failure detection. To complete this application, several types of sensor failures were taken into account. These were:

- the sensor's output remained at constant zero value.
- the sensor's output was always equal to its maximum value.
- the output was randomly intermittent.
- a bias value was introduced to the correct output.

- noise was introduced to the correct output.

These failures were simulated and included one at a time in the system. The neural controller was unable to accommodate these failures, so the introduction of a second neural controller, actually a failure-recovery neural controller, seemed to be necessary. It was implemented in parallel with the first neural controller. Its role was to keep the control vector of the closed-loop system unchanged whenever a failure occurred in a sensor. The block diagram representing sensor failure accommodation is illustrated in Figure 6.

This new controller used an error-based approach. Indeed, its input consisted of the difference between the state vector resulting from the damaged sensor and the one obtained from an ideal (model) one. The output was then the difference between the corrected control vector and the damaged one. Finally, by adding the outputs of both neural controllers, the recovered control vector was provided to the system and the problem was solved (see Figure 7). When no failure occurred, the second network did not operate since its input was zero.

For its training, a similar procedure to the one used for the first controller was tried. These training data being fixed, the selection of the network was conducted in the same way as explained above and led again to a single linear layer network.

Conclusions

The research work reported in this paper concerned the design of a neural network control system which could provide the required dynamic performance for a helicopter in both station-keeping and forward flights. First, a baseline feedback controller was designed using Linear Quadratic Regulator theory to provide a 'standard' dynamic response against which the performance of the neural controller was compared. Neural networks have proven, throughout this research, to be very efficient in helicopter control problem. Indeed, when using a Multi-Layered Perceptron network with Back-Propagation training, with a suitable architecture, the neural controller achieved remarkable results in terms of training time and error convergence. However, the Radial Basis Function network did not give acceptable results for the reported research.

Neural computing has been shown to be effective in cases of sensor failures. By implementing a neural sensor failure accommodation system in parallel with the neural controller, the desired helicopter motion was wholly recovered, even when a severe sensor failure occurred.

References

1. Bramwell A.R.S. Helicopter Dynamics, 1976.

2. Heffley R.K., Jewell W.F., Lehman J.M. and Van Winkle R.A. A Compilation and Analysis of Helicopter Handling Qualities Data, Vol 1: Data Compilation, *NASA CR-3144*, 1979.

3. Hall W.E. and Bryson A.E. Inclusion of Rotor Dynamics in Controller Design for Helicopters, *Journal of Aircraft*, 10 (4), 200-6, 1973.

4. Murphy R.D. and Narendra K.S. Design of Helicopter Stabilization Systems using Optimal Control Theory, *Journal of Aircraft*, 6(2), 129-36, 1969.

5. Simpson P.K. Neural Networks Paradigms, AGARD-LS-179, 1991.

6. Armitage A.F. Neural Networks in Measurement and Control, *Measurement and Control*, 28, 208-15, 1995.

7. Vogl T.P. Mangis J.K., Rigler A.K., Zink W.T. and Alkon D.L. Accelerating the Convergence of the Back-Propagation Method, *Biological Cybernetics*, 59, 257-63, 1988.

8. Sperduti A. and Starita A. Speed Up Learning and Network Optimization with Extended Back Propagation, *Neural Networks*, 6, 365-83, 1993.

9. Napolitano M.R., Neppach C., Casdorph V., Naylor S., Innocenti M. and Silvestri G. Neural Network-Based Scheme for Sensor Failure Detection, Identification and Accommodation, *Journal of Guidance*, Control and Dynamics, 18(6), 1280-6, 1995.

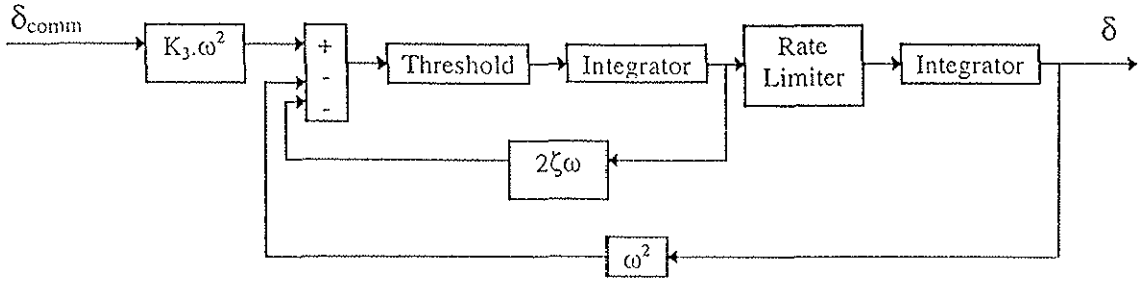

Figure 1

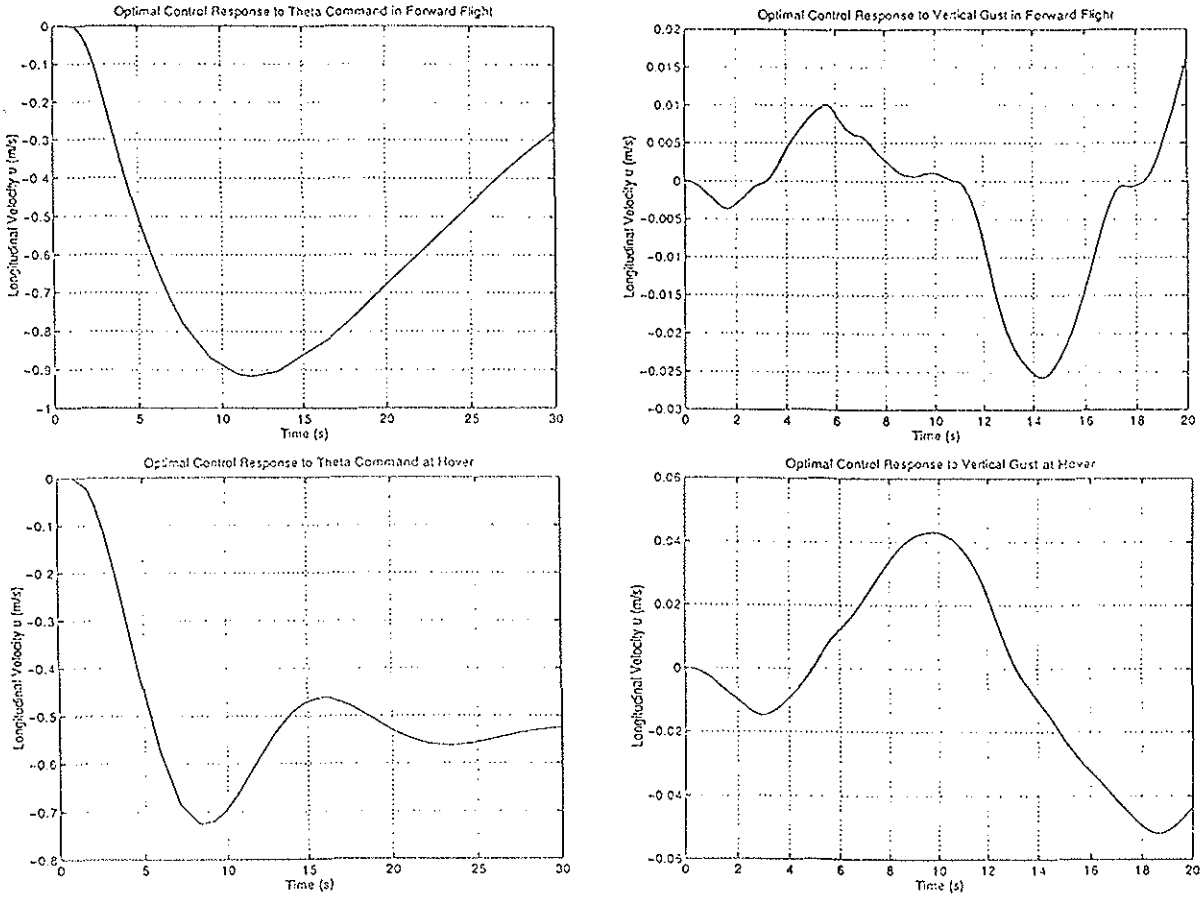

Figure 2

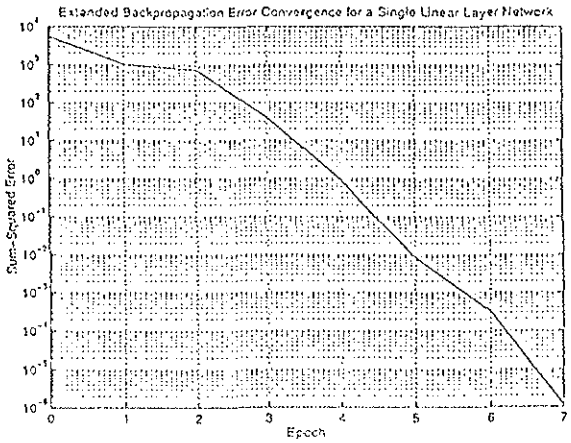

Figure 3

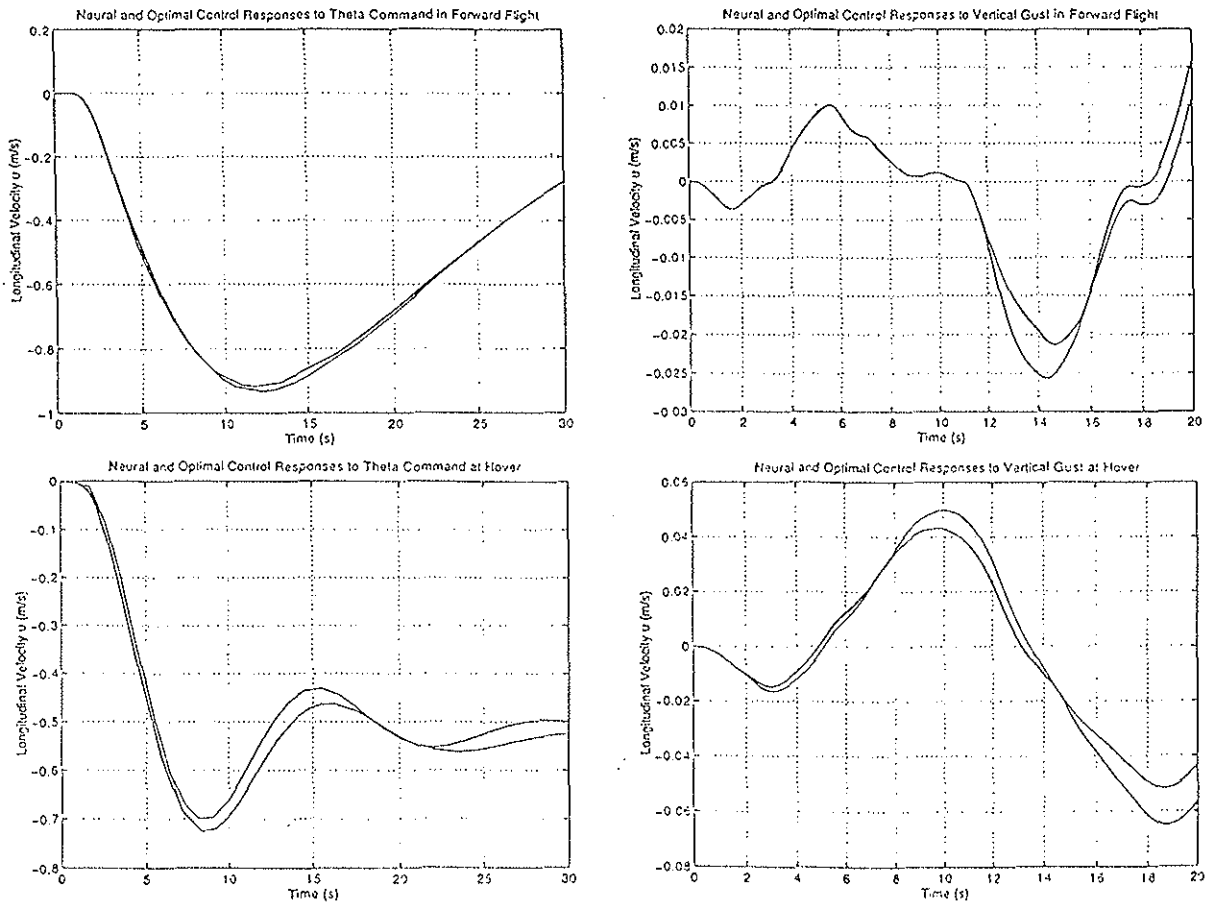

Figure 4

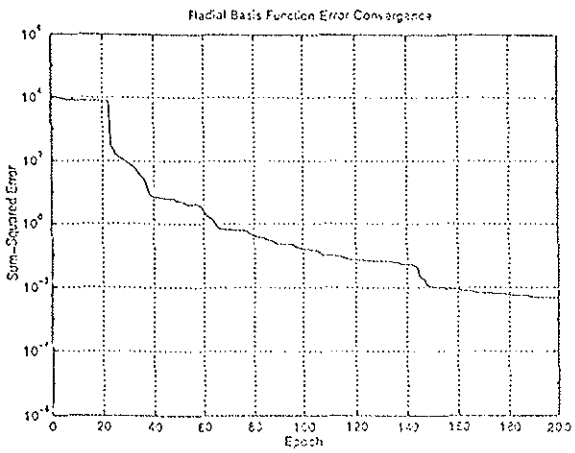

Figure 5

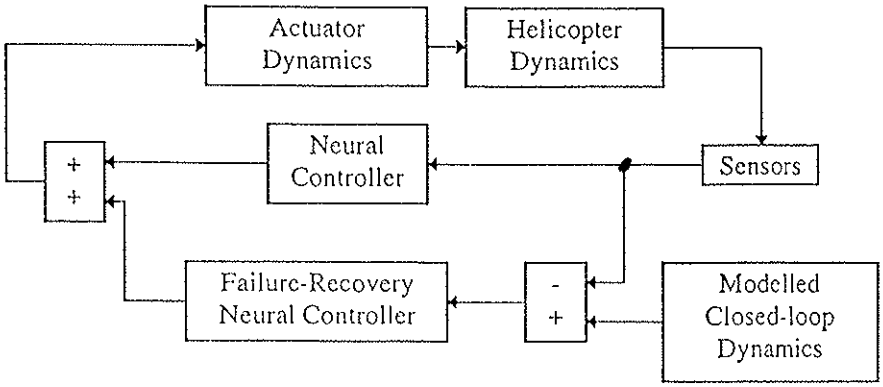

Figure 6

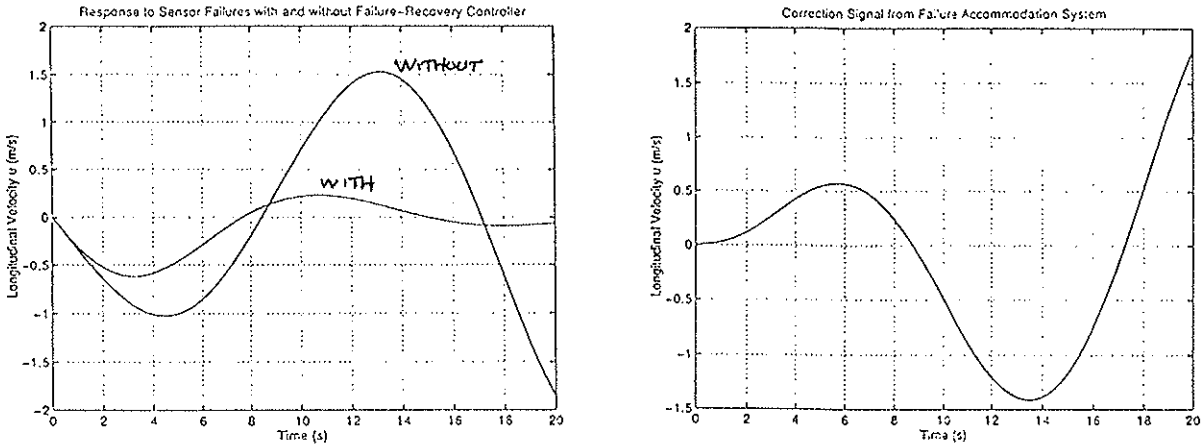

Figure 7

Annex 1

$$A_{hf} = \begin{bmatrix} -0.0056 & 0.0257 & 0.2206 & -9.81 & -0.0088 & -0.3844 & -0.1050 & 0 \\ -0.0716 & -0.3286 & 0.6056 & 0 & -0.0819 & -0.1184 & 0.6666 & 0 \\ 0.0077 & -0.0132 & -0.3135 & 0 & 0.0066 & 0.2504 & 0.0371 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0.0159 & -0.0039 & -0.3735 & 0 & -0.0439 & -0.3208 & 0.2326 & 19.0856 \\ 0.0322 & -0.0161 & -0.9162 & 0 & -0.0526 & -0.7415 & 0.1608 & 0 \\ -0.0038 & -0.0204 & -0.1973 & 0 & 0.0689 & -0.3041 & -0.7102 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

$$B_{hf} = \begin{bmatrix} 0.0861 & 0.1264 & -0.0008 & -0.0003 \\ -1.0106 & 0.0344 & 0.0011 & 0.0015 \\ -0.0021 & -0.0801 & 0.0005 & 0.0069 \\ 0 & 0 & 0 & 0 \\ -0.0326 & 0.0014 & 0.1072 & 0.1709 \\ -0.0514 & 0.003 & 0.2518 & 0.1844 \\ 0.1860 & -0.0001 & 0.0367 & -0.4585 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Annex 2

$$A_{ff} = \begin{bmatrix} -0.0516 & 0.0853 & 0.2706 & -9.81 & 0.0790 & 0.1507 & 0 & 1.04 & 0 \\ 0.1091 & -0.9613 & 49.9017 & 0 & 0.5776 & -1.5265 & 0 & 0.321 & 0 \\ 0.0203 & -0.0285 & -0.6133 & 0 & -0.0759 & -0.0147 & 0 & -0.169 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.1119 & 0.9937 & 0 & -51.44 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4.97 & 0 & 0 & 0 & 0 & -0.333 & 0 \\ 0.9937 & 0.1119 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$B'_{ff} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

Annex 3

$$A_{rotorhf} = \begin{bmatrix} \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -41.3 & -600 & -30.3 & -42.6 \\ 600 & -54.3 & 41.4 & -27.3 \end{pmatrix} & \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.14 & 0 & -30.4 & 0 & -0.26 & -50.6 & 0 & 0 \\ -0.28 & 0 & 50.2 & 0 & -0.12 & -30.3 & 0 & 0 \end{pmatrix} \\ \begin{pmatrix} -15 & 21.6 & 1 & -0.05 \\ 0 & 0 & 0 & 0 \\ 5 & -0.92 & -0.04 & 0.004 \\ 0 & 0 & 0 & 0 \\ 22 & 15 & -0.05 & -1 \\ 3.5 & 18.4 & -0.01 & -0.17 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} & A_{hf} \end{bmatrix}$$

$$B_{rotorhf} = \begin{bmatrix} \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & -1.5 & -600 & 0 \\ 0 & 600 & 5.5 & 0 \end{pmatrix} \\ B_{hf} \end{bmatrix}$$

$$A_{rotor\,ff} = \begin{bmatrix} \begin{pmatrix} -30.3 & -41.3 \\ 1 & 0 \end{pmatrix} & \begin{pmatrix} 0.14 & 0 & -30.4 & 0 & -600 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \\ \begin{pmatrix} 1 & -15 \\ 0 & 0 \\ -0.04 & 5 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix} & A_{ff} \end{bmatrix} \qquad B_{rotor\,ff} = \begin{bmatrix} \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \\ B_{ff} \end{bmatrix}$$