OSU-ECE Report NASA 91-01

Annual Report on

NONLINEAR STABILITY AND CONTROL STUDY OF HIGHLY MANEUVERABLE HIGH PERFORMANCE AIRCRAFT

(NASA GRANT NO. NAG-1-1081)

Date: August 12, 1991

R.R. Mohler, Principal Investigator

Oregon State University
Department of Electrical and Computer Engineering
Corvallis, Oregon 97331-3211

(503) 737-3617/3470

Graduate Research Assistants: S. Cho, C. Koo, R. Zakrzewski
Undergraduate Participants (NSF Support): D. Aaberge, P. Shirkey
Visiting Researchers (ADA-Israel Support): J. Dory, Z. Halevy

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION AND SUMMARY

The purpose of this research is to develop and to apply new nonlinear system methodologies to the stability analysis and adaptive control of high angle-of-attack (α) aircraft such as the F18. The present progress report reviews the research of this project over the first year (actually 18 months with a no-cost extension).

Considerable progress is documented on nonlinear adaptive control and associated model development, identification, and simulation. Also, it appears that previously derived results for bilinear system (BLS) stability [1,2] as well as describing functions [3] can be adapted to the Ostroff PIF controller and the thrust vectoring component with dead zone. The latter will receive more emphasis in conjunction with the PIF and other controls studied here.

The analysis has considered linear and nonlinear, longitudinal, high-α aircraft dynamics with varying degrees of approximation dependent on the purpose as summarized in Table 1. In all cases, angle of attack (α) or pitch rate (q) has been controlled primarily by a horizontal stabilizer (δ_h). In most cases studied, a linear adaptive controller provides sufficient stability. However, it has been demonstrated by simulation of a simplified nonlinear model [4] that certain large rapid maneuvers were not readily stabilized by the investigated linear adaptive control but were by means of a nonlinear time-series based adaptive control. More details of the research competed by this period thus far are reported in Sections 2 and 3 below and in previous semiannual reports. With regards to nonlinear simulation programming, it is shown that with C language it is possible to improve computation speed by two orders of magnitude over the previously used MATLAB.

Table 1. Aircraft Models

Type	Purpose	Remarks/Limitations
1. Linear perturbations at $\alpha = 5°, 15°, 35°, 60°$	Local control, check of nonlinear system, application of well developed linear control methodologies Local stability	Only valid for small maneuvers Special case of types 2-5
2. Gain scheduled (nonlinear function of α) from 1	Gain-scheduled adaptive control based on well developed methodologies Simplified description of complex system Approximate stability	May have stability problems with small number of reference states and/or large fast maneuvers
3. Volterra series a) at reference states b) general case	Nonlinear adaptive control via cross-correlation and/or á priori dynamic structure stability approximation Simplified dynamic description of complex system	Non-orthogonal series approximation Sufficiency of 2 or 3 kernels Large computation time for adaptation
4. Bilinear system a) continuous b) BARMA 5. Polynomial time series	Nonlinear adaptive control via model reference identification (NLMRAC) Stability approximation Simplified dynamic description	Large computation time Bilinearizing controllers may be more practical than linearizing ones Polynomial approximation may be more accurate but more time consuming than linear or bilinear approximation
6. Neural network	Potential application to adaptive control	Probably less accurate than 4 or 5 for a given data set but accuracy may be more robust outside the available data set
7. Nonlinear ordinary differential model	Accurate approximation to fast large maneuvers for "final" design and simulation Stability	Neglects flexible modes and other complications

2. MODELS AND SIMULATION

2.1 Basic Nonlinear Dynamics

The dynamic equations of motion are established by a nonlinear six-degree-of-freedom aerodynamic model. In general, the aerodynamic-force components referred to as the center of gravity (CG) is denoted as (X, Y, Z). The aerodynamic angular moment vector about the CG is given by (FL, FM, FN). The thrust vector T is represented in body coordinates as (T_x, T_y, T_z). Then the four equations with respect to body axes become

$$\dot{u} = rv - q\omega\sin(\theta) + \frac{X}{m} + \frac{T_x}{m} \tag{2.1}$$

$$\dot{v} = p\omega - ru + g\cos(\theta)\sin(\phi) + \frac{Y}{m} + \frac{T_y}{m} \tag{2.2}$$

$$\dot{w} = qu - pv + g\cos(\theta)\cos(\phi) + \frac{Z}{m} + \frac{T_z}{m} \tag{2.3}$$

The moment equations with respect to CG and body axes:

$$\dot{p} = C_{41}pq + C_{42}qr + C_{43}FN + C_{40}FL + \frac{C_{43}}{I_{zz}}(P_{z\bullet}T_y - P_{y\bullet}T_x) + \frac{C_{40}}{I_{xx}}(P_{y\bullet}T_z - P_{z\bullet}T_y) \tag{2.4}$$

$$\dot{q} = C_{51}pr + C_{52}(r^2 - p^2) + FM + \frac{(P_{z\bullet}T_x - P_{x\bullet}T_z)}{I_{yy}} \tag{2.5}$$

$$\dot{r} = C_{61}pq + C_{62}qr + C_{63}FL + C_{40}FN + \frac{C_{63}(P_{y\bullet}T_z - P_{z\bullet}T_y)}{I_{xx}} + \frac{C_{40}(P_{x\bullet}T_y - P_{y\bullet}T_x)}{I_{zz}} \tag{2.6}$$

The Euler equations:

$$\dot{\theta} = q\cos(\phi) - r\sin(\phi) \tag{2.7}$$

$$\dot{\phi} = p + q\tan(\theta)\sin(\phi) + r\tan(\theta)\cos(\phi) \tag{2.8}$$

where the vector (P_x, P_y, P_z) denotes the position vector from the center of mass (CG) to the aerodynamic center (AC) and the vector (P_{xe}, P_{ye}, P_{ze}) denotes position vector from the center of mass to the engine thrust center. The constants in the moment equations (1.4-1.6) are functions of the moment of inertia quantities (I_{xx}, I_{yy}, I_{zz}, and I_{xz}) as follows:

$$C_{40} = I_{xx}I_{zz}\left(I_{xx}I_{zz} - I_{xz}^2\right) \tag{2.9}$$

$$C_{41} = C_{40}I_{xz}\left(I_{zz} + I_{xx} - I_{yy}\right)/I_{xx}I_{zz} \tag{2.10}$$

$$C_{42} = C_{40}\left(I_{zz}\left(I_{yy} - I_{zz}\right) - I_{xz}^2\right)/I_{xx}I_{zz} \tag{2.11}$$

$$C_{43} = C_{40}I_{xz}/I_{xx} \tag{2.12}$$

$$C_{51} = \left(I_{zz} - I_{xx}\right)/I_{yy} \tag{2.13}$$

$$C_{52} = I_{xz}/I_{yy} \tag{2.14}$$

$$C_{61} = C_{40}\left(I_{xx}\left(I_{xx} - I_{yy}\right) + I_{xz}^2\right)/I_{xx}I_{zz} \tag{2.15}$$

$$C_{62} = C_{40}I_{xz}\left(I_{yy} - I_{zz} - I_{xx}\right)/I_{xx}I_{zz} \tag{2.16}$$

$$C_{63} = C_{40}I_{xz}/I_{zz} \tag{2.17}$$

The quantities X, Y, Z, FL, FM, and FN depend on the aerodynamic coefficients C_D, C_y, C_L, C_ℓ, C_m, C_n as follows:

$$D = \bar{q}sC_D \tag{2.18}$$

$$L = \bar{q}sC_L \tag{2.19}$$

$$X = -D\cos(\alpha) + L\sin(\alpha) \tag{2.20}$$

$$Y = \bar{q}sC_y \tag{2.21}$$

$$Z = -D\sin(\alpha) - L\cos(\alpha) \tag{6.22}$$

$$FL = \frac{(\bar{q}sbC_\ell + P_yZ - P_zY)}{I_{xx}} \tag{2.23}$$

$$FM = \frac{(\bar{q}s\bar{c}C_m + P_zX - P_xZ)}{I_{yy}} \tag{2.24}$$

$$FN = \frac{(\bar{q}sbC_n + P_xY - P_yX)}{I_{zz}} \tag{2.25}$$

$\bar{q}$ is dynamic pressure, s effective area, and a, b, $\bar{c}$ moment arms.

Angle of Attack, Sideslip, and Total Speed

With respect to body axes, the angle of attack, α, the sideslip, β, the total speed, V, are defined as

$$\alpha = \tan^{-1}\left(\frac{w}{u}\right) \tag{2.26}$$

$$\beta = \sin^{-1}\left(\frac{v}{V}\right) \tag{2.27}$$

$$V = u^2 + v^2 + w^2 \tag{2.28}$$

Mathematical Structure of Aerodynamic Coefficients

The mathematical structure of the aerodynamic coefficients are based on the wind tunnel test data for the high angle of attack vehicle. The aerodynamic coefficients are considered to be functions of the following control variables as well as angle of attack, sideslip, Mach number, altitude, roll, pitch, and yaw rates: aileron deflection, rudder deflection, and stabilator deflection. The effects of leading edge flap, trailing edge flap, speed brake, landing gear, etc., are not considered.

Drag Coefficient:

$$C_D = C_0(\alpha, M, h, \delta_h) \tag{2.29}$$

Lift Coefficient:

$$C_L = C_{L0}(\alpha, M, h, \delta_h) + \frac{\bar{c}}{2V}\left[C_{Lq}(\alpha, M, h)\, q + C_{L\dot{\alpha}}(\alpha, M, h)\, \dot{\alpha}\right] \tag{2.30}$$

Pitching Moment:

$$C_m = C_{m0}(\alpha, M, h, \delta_h) + \frac{\bar{c}}{2V}\left[C_{mq}(\alpha, M, h)\, q + C_{m\dot{\alpha}}(\alpha, M, h)\, \dot{\alpha}\right] \tag{2.31}$$

Side Force Coefficient:

$$C_y = C_{y0}(\alpha, B, M, \delta_a, \delta_r) + C_{yp}(\alpha, M, h)\, \beta + \frac{T_o}{2V}\left[C_{yp}(\alpha, M, h)\, p + C_{yr}(\alpha, M, h)\, r\right] \tag{2.32}$$

Rolling Moment Coefficient:

$$C_\ell = C_{\ell 0}(\alpha, \beta, M, \delta_a, \delta_r) + C_{\ell p}(\alpha, M, h)\, \beta + \frac{\bar{b}}{2V}\left[C_{\ell p}(\alpha, M, h)\, p + C_{\ell r}(\alpha, M, h)\, r\right] \tag{2.33}$$

Yawing Moment Coefficient:

$$C_n = C_{n0}(\alpha, \beta, M, \delta_a, \delta_r, \delta_h) + C_{n\beta}(\alpha, M, h)\, \beta + \frac{\bar{b}}{2V}\left[C_{np}(\alpha, M, h)\, p + C_{nr}(\alpha, M, h)\, r\right] \tag{2.34}$$

Range of States Variables in Aerodynamic Coefficients

α (angle of attack)	-10° to 90°
p (sideslip)	-20° to 20°
M (Mach number)	0.2 to 2.0
h (altitude)	0 to 60,000 ft

Control Variables and Their Limits

δ_a (aileron deflection)	-25° to 25°
δ_r (rudder deflection)	-30° to 30°
δ_h (stabilator deflection)	-24° to 10.5°
δ_T (throttle)	30° to 131°

Longitudinal Dynamic Equation

Assume that the motion of the airplane can be analyzed by separating the equations into two groups. The X-force, Z-force, and pitching moment equations comprise the longitudinal equations and the Y-force, rolling, and yawing moment equations are called the lateral equations. Longitudinal dynamic equations are given by (2.1), (2.3), and (2.5).

Aerodynamic coefficients are functions of angle of attack, total speed, Mach number, etc.

We can choose state variables as α, V, q, and θ instead of u, w, q, and θ and we assume that $v = 0$, $p = 0$, $r = 0$, $\phi = 0$.

From relationship between angle of attack (α) and air speed (V)

$$u = V\cos\alpha$$

$$w = V\sin\alpha$$

$$\dot{\alpha} = \frac{u\dot{w} - \dot{u}w}{V^2}, \qquad -10^\circ < \alpha < 90^\circ$$

From above, we define normal acceleration by

$$\ddot{n}_z = \dot{V}\sin(\theta - \alpha) + V\cos(\theta - \alpha)\,(\dot{\theta} - \dot{\alpha})$$

$$= \dot{V}\sin(\theta - \alpha) + V\cos(\theta - \alpha)\,(q - \dot{\alpha})$$

This normal acceleration term can be very important in state feedback and in calculating controller gains in general.

2.2 Linear Perturbation Simulation

The linear perturbation equations were derived by means of a Taylor series of the above model at the four reference states corresponding to $\alpha = 5°, 15°, 35°, 60°$. For the F18 data the short period eigenvalues are given by $\lambda_{1,2} = -0.559 \pm j\,0.337$ (damping $\zeta = 0.386$, period T = 4.7 sec) and for the phugoid $\lambda_{3,4} = -0.0085 \pm j\,0.073$ ($\zeta' = 0.117$, $T' = 86$ sec). Figures 1 through 8 show example angle of attack and pitch rate responses to step changes in horizontal stabilator at 5° and at 60° with and without phugoid mode component in the simulation. Neglecting the phugoid is similar to assuming constant air speed. Note that the short-period responses (and eigenvalues) agree quite well with NASA simulations [4].

2.3 Simplified Nonlinear Models

The question of interest here was to investigate the possibility of developing an accurate but simplified nonlinear model that would remain valid in a large range of operating conditions and at the same time would be capable of rendering nonlinear phenomena occurring in high angle of attack post stall flight regime. Usual practice in aircraft modeling is to characterize its dynamics by providing so-called stability derivatives for different operating conditions. Stalford et al. [5] proposed using a Volterra series approach for longitudinal as well as lateral aircraft dynamics, claiming that obtained models characterize the systems behavior much better than piecewise linear ones. However, they did not try to find a global model (i.e., valid for a large range of angle of attack), developing instead four Volterra series submodels obtained from the expansion around for equilibria corresponding to different ranges of angle of attack. Although their approximate piecewise Volterra model indeed gives results that agree very accurately with the original model derived from wind tunnel experiments data, yet this is hardly due to the inclusion of higher-order terms. In

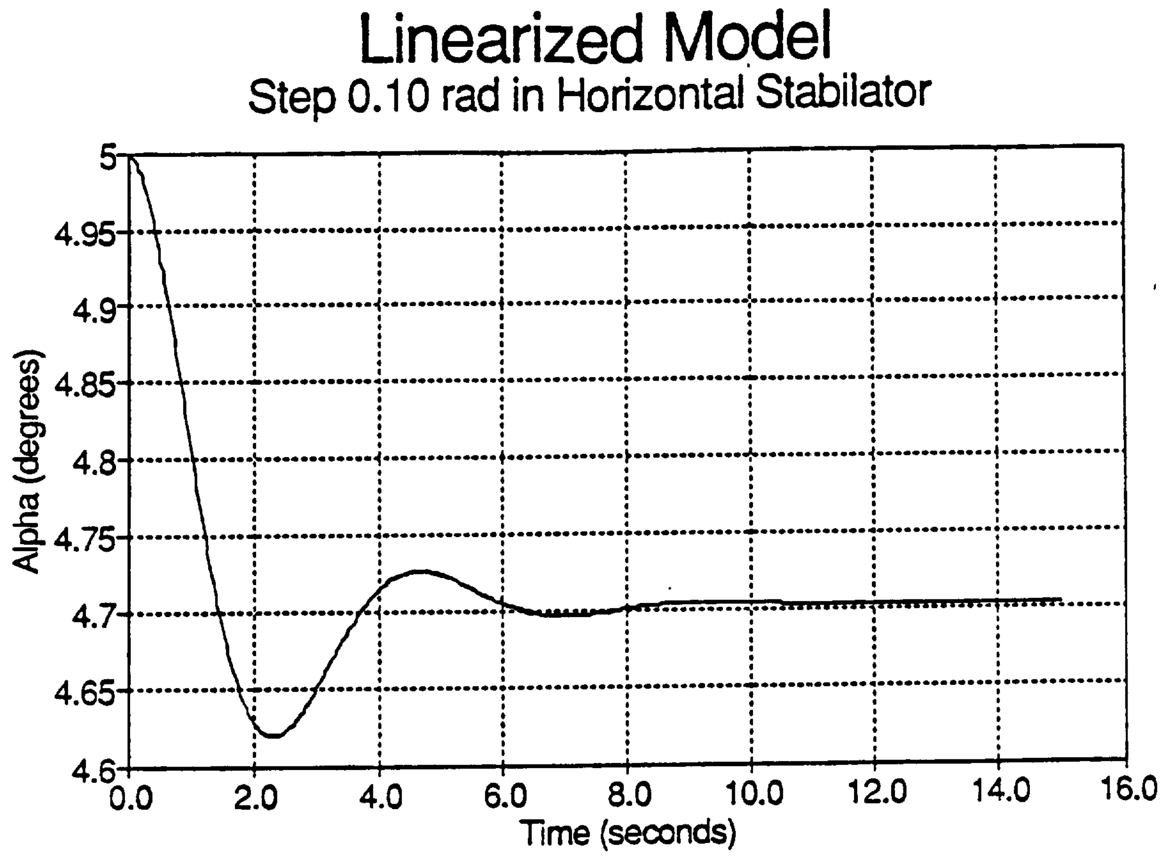


Figure 1. Angle of Attack Response without Phugoid Mode, α_{ref} = 5°, δ_h = 0.10 rad-step

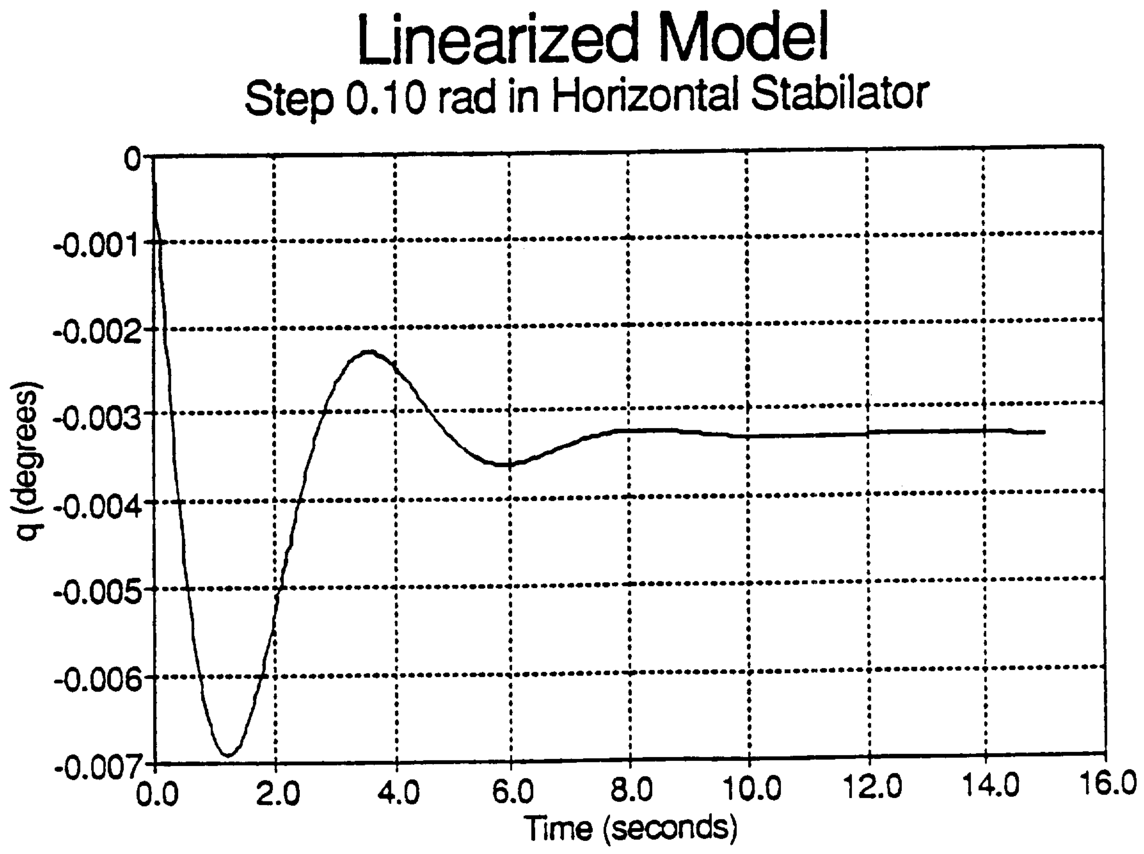


Figure 2. Pitch Rate Response without Phugoid Model, α_{ref} = 5°, δ_h = 0.10 rad-step

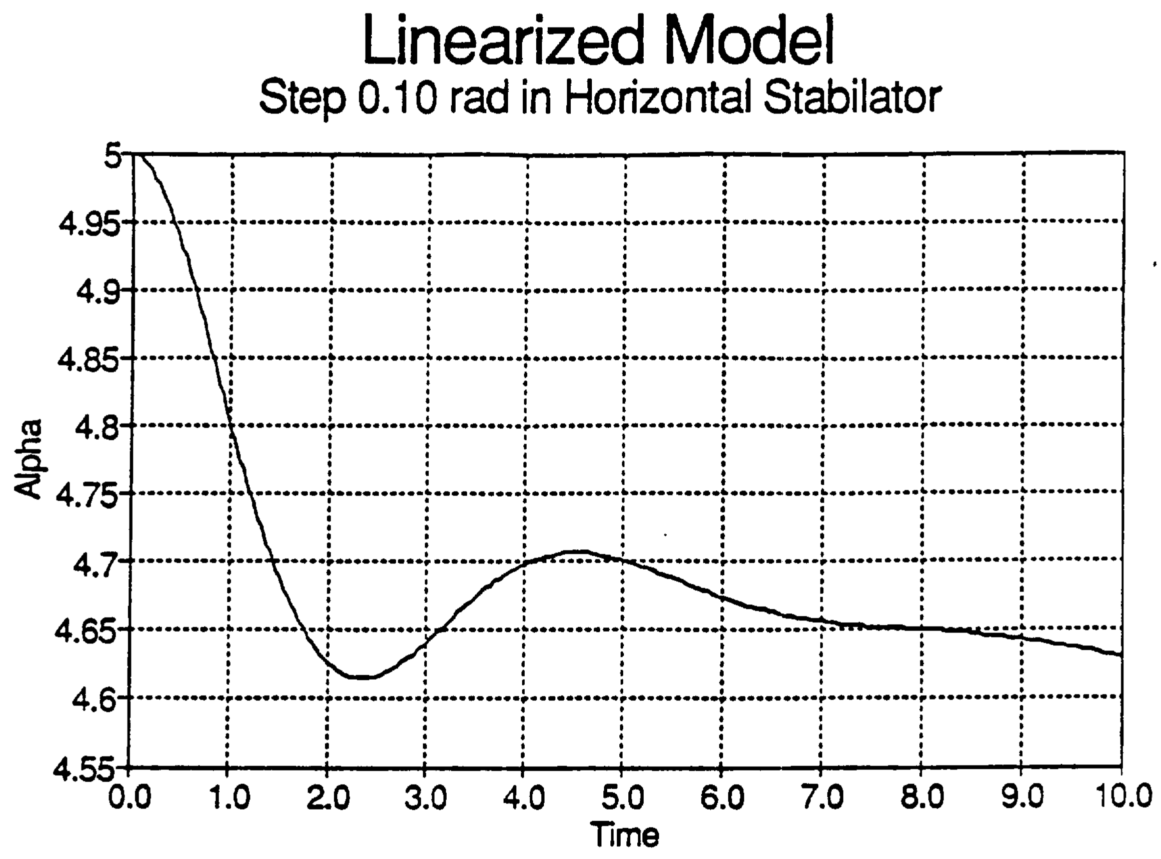


Figure 3. Angle of Attack Response, $\alpha_{ref} = 5°$, δ_h = 0.10 rad-step

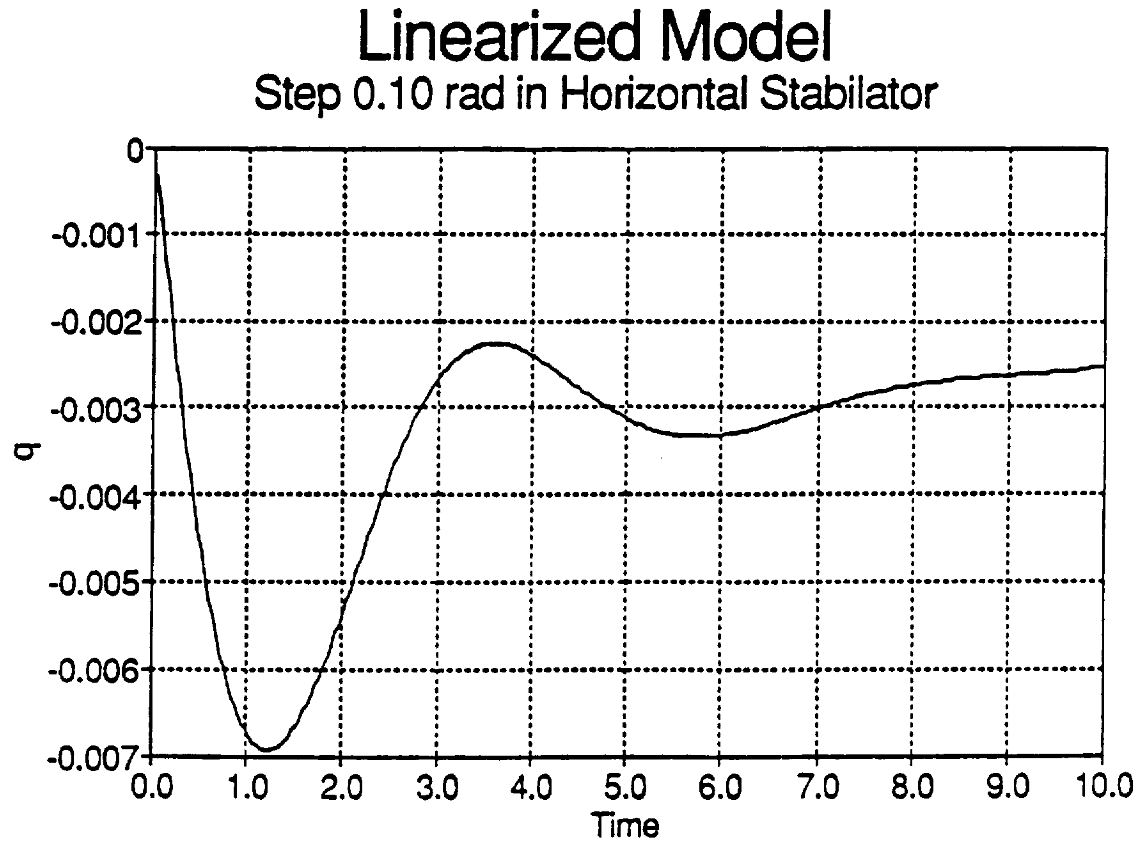


Figure 4. Pitch Rate Response, $\alpha_{ref} = 5°$, δ_h = 0.10 rad-step

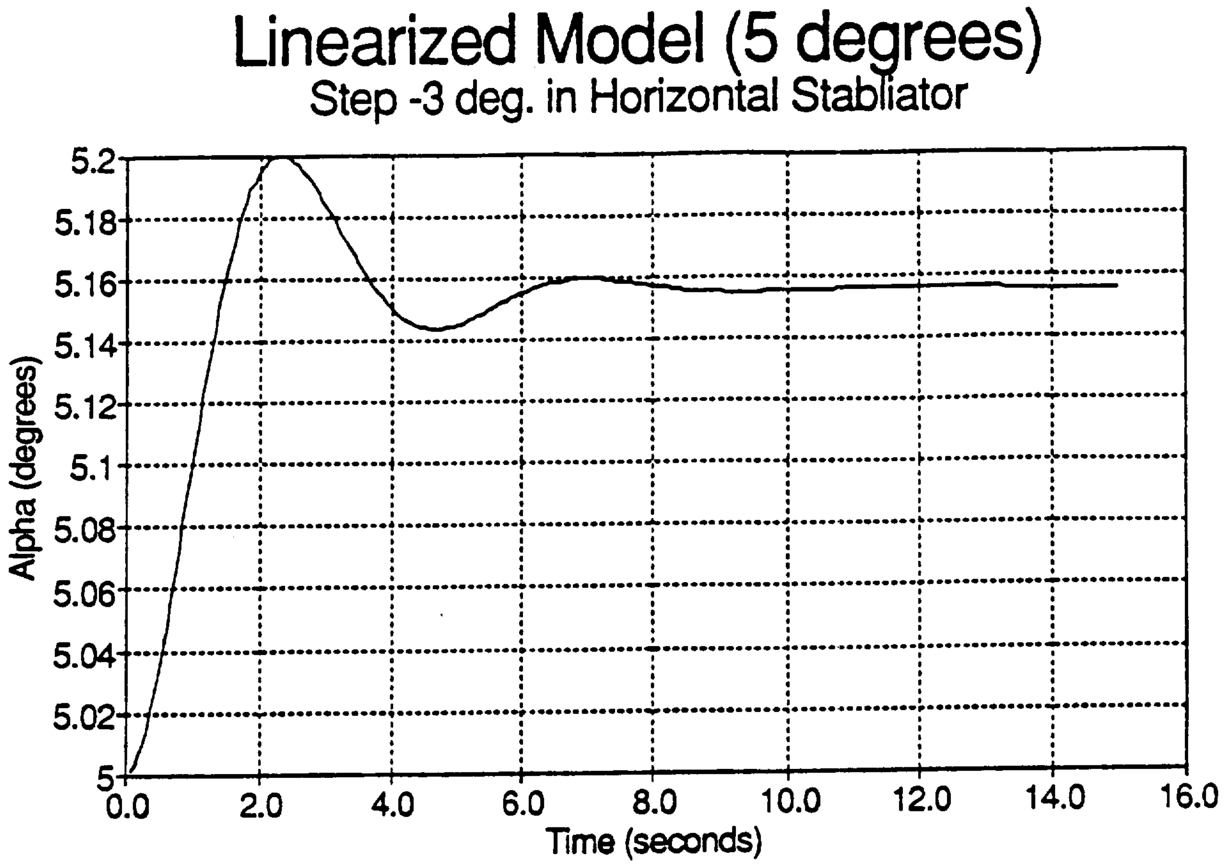


Figure 5. Angle of Attack Response without Phugoid Mode, α_{ref} = 5°, δ_h = -3° step

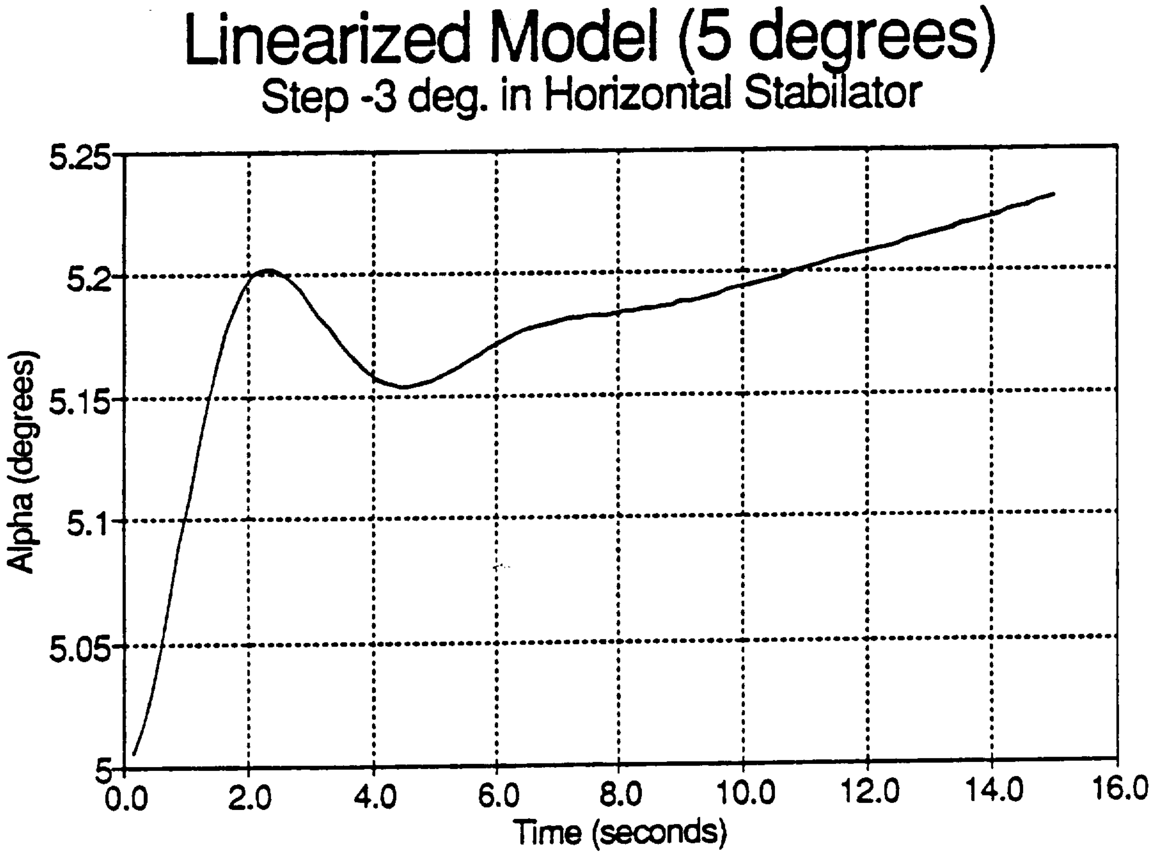


Figure 6. Angle of Attack Response, α_{ref} = 5°, δ_h = -3° step

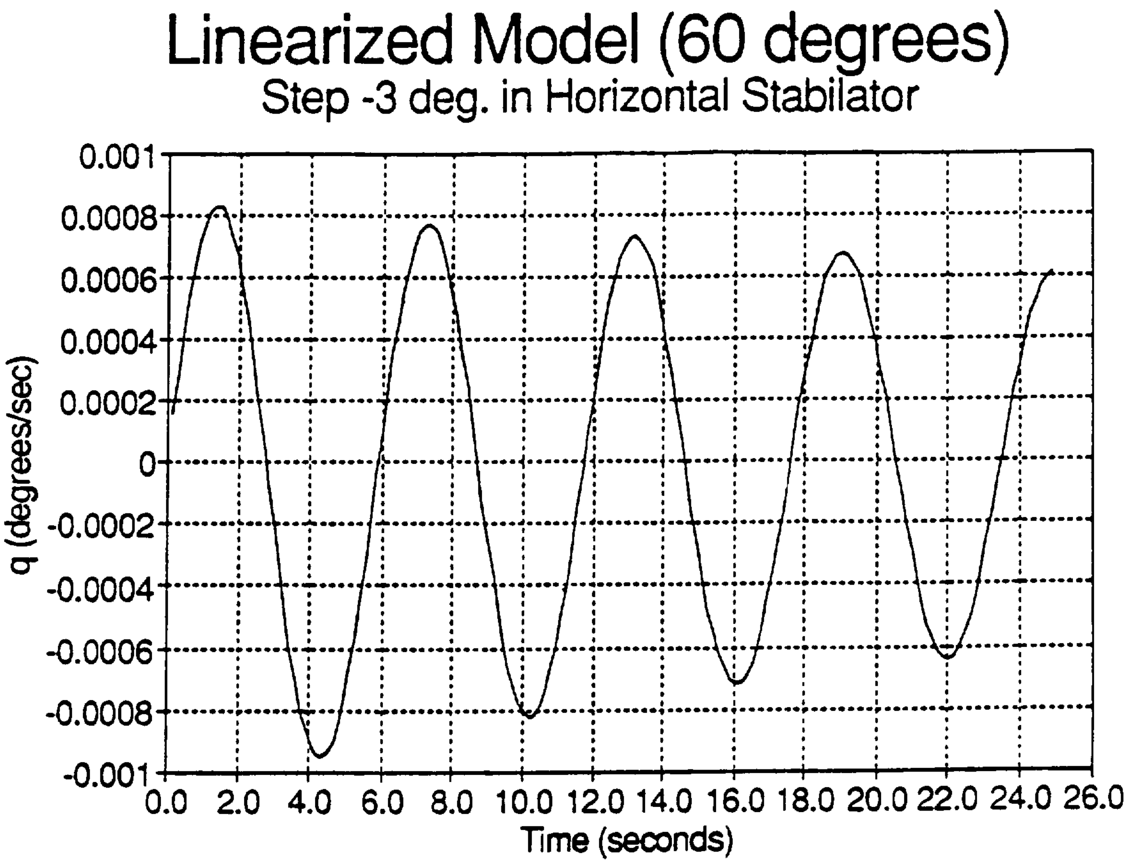


Figure 7. Pitch Rate Response without Phugoid Mode, $\alpha_{ref} = 60°$

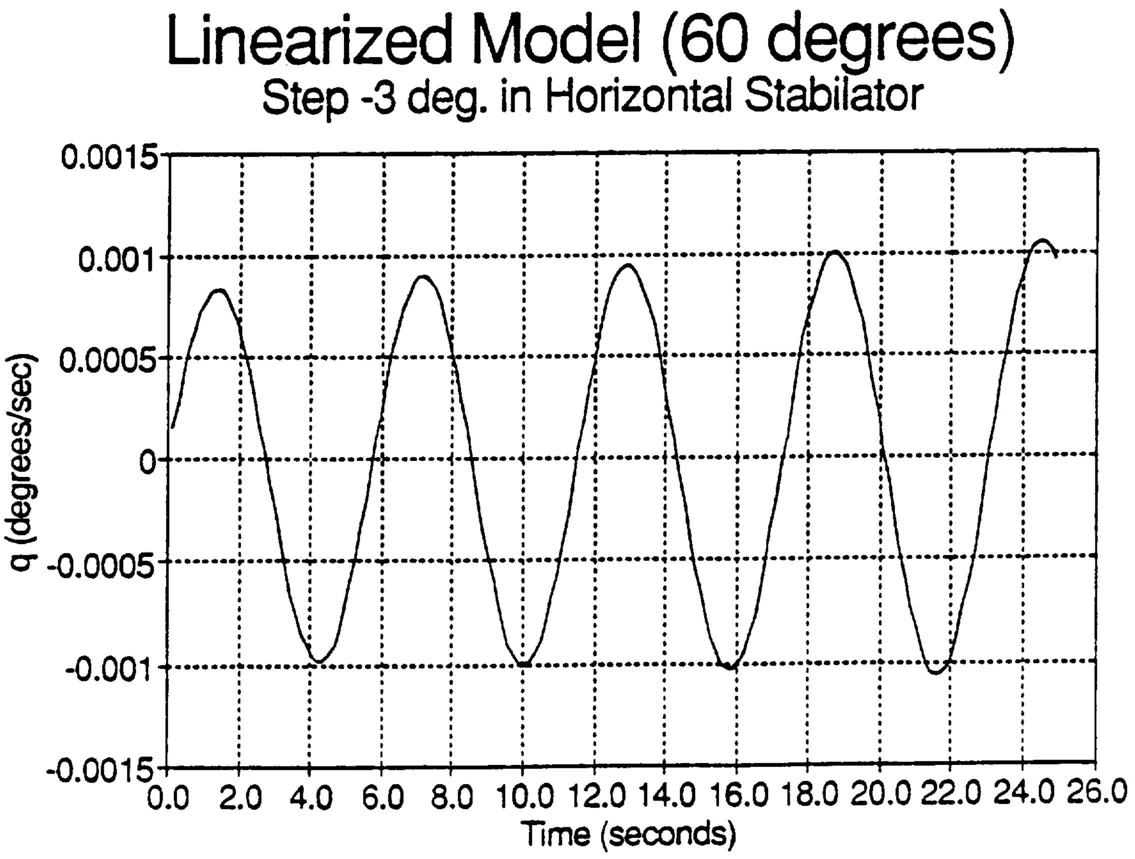


Figure 8. Pitch Rate Response, $\alpha_{ref} = 60°$, $\delta_h = -3°$ step

fact, the contribution of the latter to the final solution is almost negligible. The piecewise-linear model obtained by omitting higher-order terms from the Volterra approximation is seen to give almost the same results as the original nonlinear model, as can be observed in Figures 9-10. The comparison in [5] was made with a piecewise linear model obtained from linearization around equilibria different than those used for Volterra series expansion, although the approximation of plunging force coefficient resulting from such a linearization seemed much better than from first terms of Volterra series expansion. This, however, does not mean that nonlinear modeling of aircraft dynamics has no advantages over piecewise linear models. The conclusion that can be drawn is merely that while piecewise linear models may accurately predict complex nonlinear behavior they are very sensitive to the choice of points of linearization and that the best piecewise linear fit to the curve of one of the model coefficients does not necessarily have to give the best dynamical model.

The simplified longitudinal aircraft dynamics model described in [5] was taken as a basis for investigation of nonlinear phenomena that may occur in high angle of attack regime of flight. The model is as follows:

$$\begin{aligned} \dot{\alpha} &= q + 9.168c_z(\alpha) - 1.8336(\delta_h + 7^\circ) + 7.361904 \\ \dot{q} &= 5.73(\alpha - 1.5\delta) + 2.865 \end{aligned} \tag{2.35}$$

where:

α = angle of attack (deg)

q = pitch rate (deg/s)

δ_h = elevator control (deg), horizontal stabilator

$c_z(\alpha)$ = plunging force coefficient (Figure 11)

The nonlinearity is seen to come from the angle of attack. The state plane portraits of the system for different constant values of control are shown in Figures 12-17. Stable equilibria of the system correspond to values of control less than 9.49 or greater than 12.24 or to angle of attack less than 14.74 or greater than 18.87. In the region between those equilibria an unstable and a stable limit cycle occurs. Interesting phenomena can be observed for the zone near to the onset of unstability. As seen in Figures 15 and 16, the equilibrium is still stable and at the same time two limit cycles exist, the inner of which is unstable and

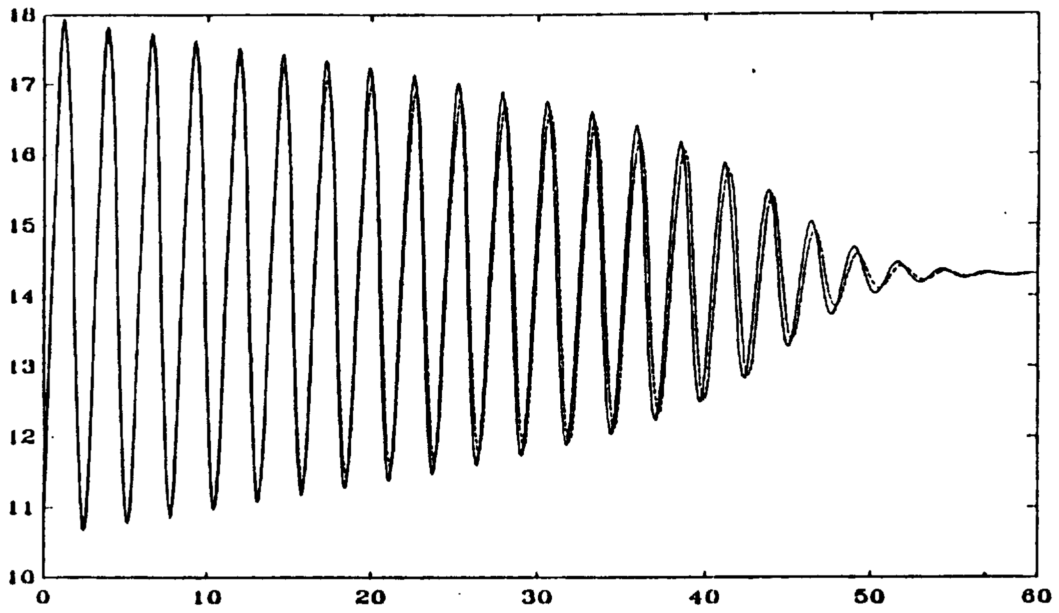


Figure 9a. Nonlinear Solution vs. Third-Order Volterra Solution

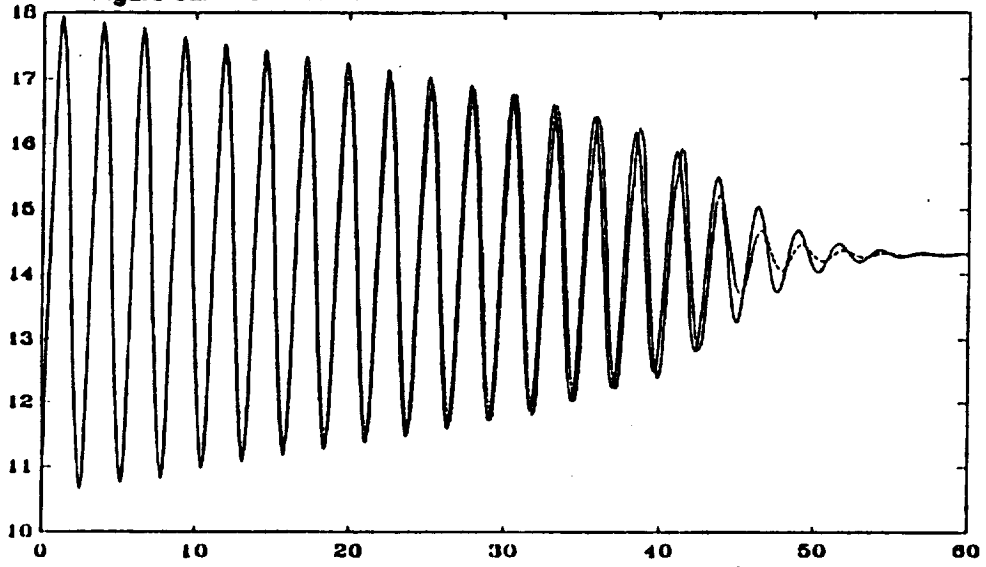


Figure 9b. Nonlinear Solution vs.First-Order (Linear) Solution

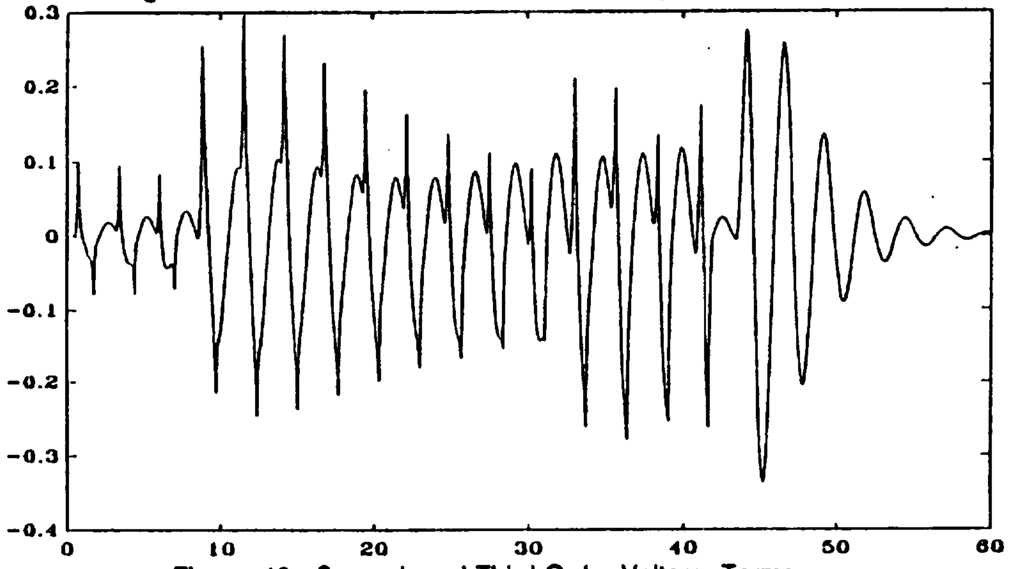


Figure. 10. Second- and Third-Order Volterra Terms

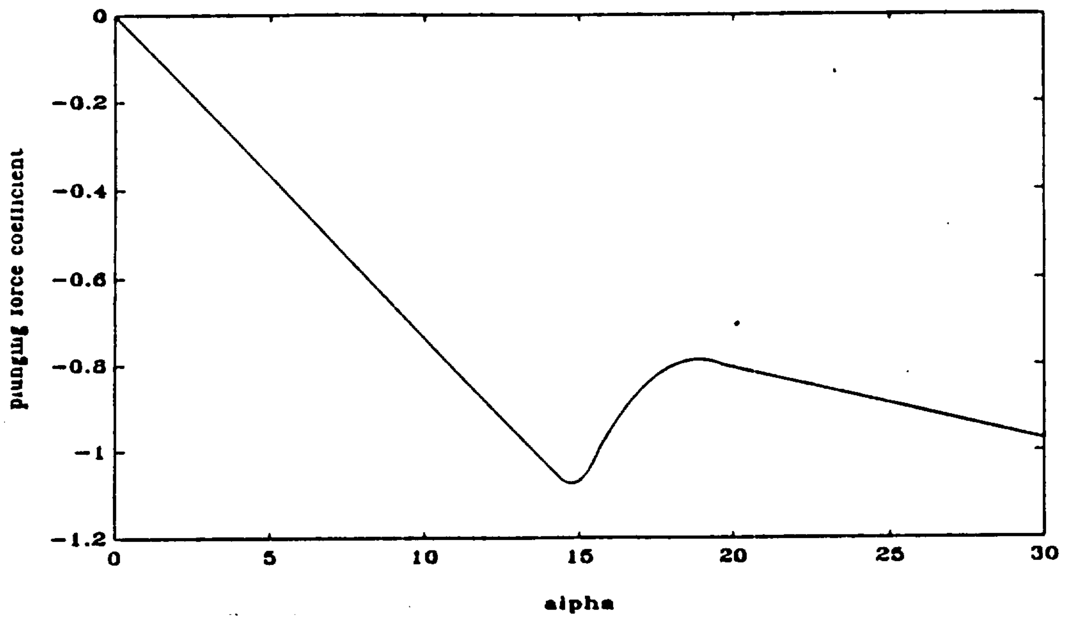


Figure 11. Plunging Force Coefficient

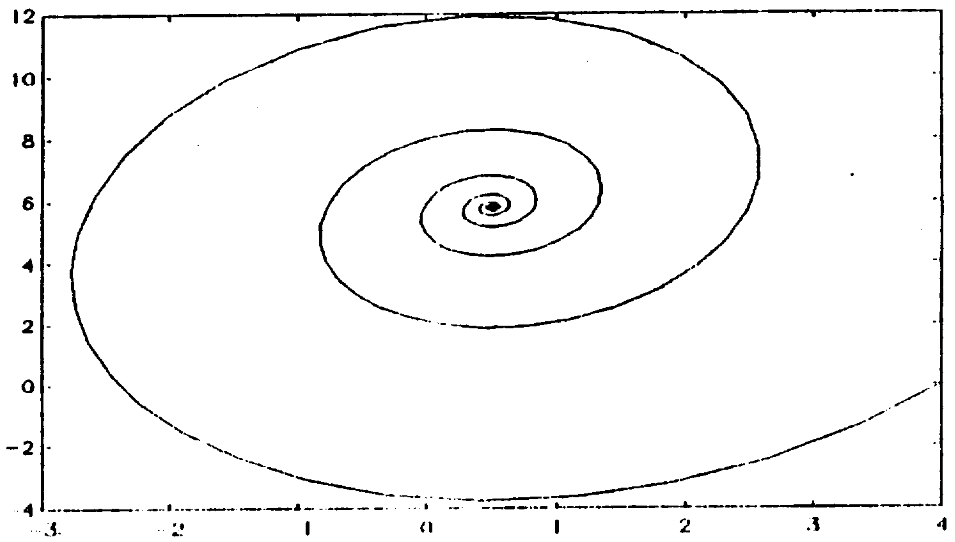


Figure 12. State-Space Portrait, $\delta_h = 0$

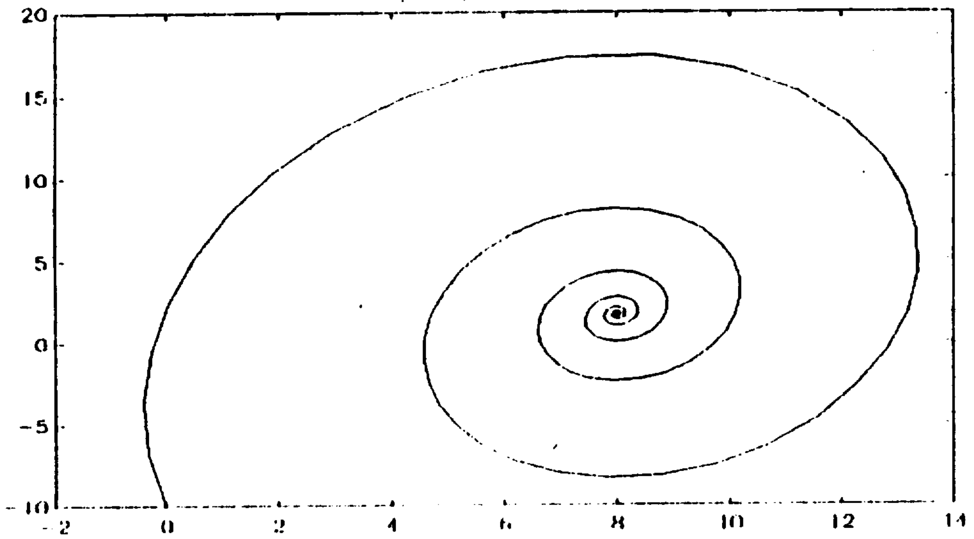


Figure 13. State-Space Portrait, $\delta_h = -5°$

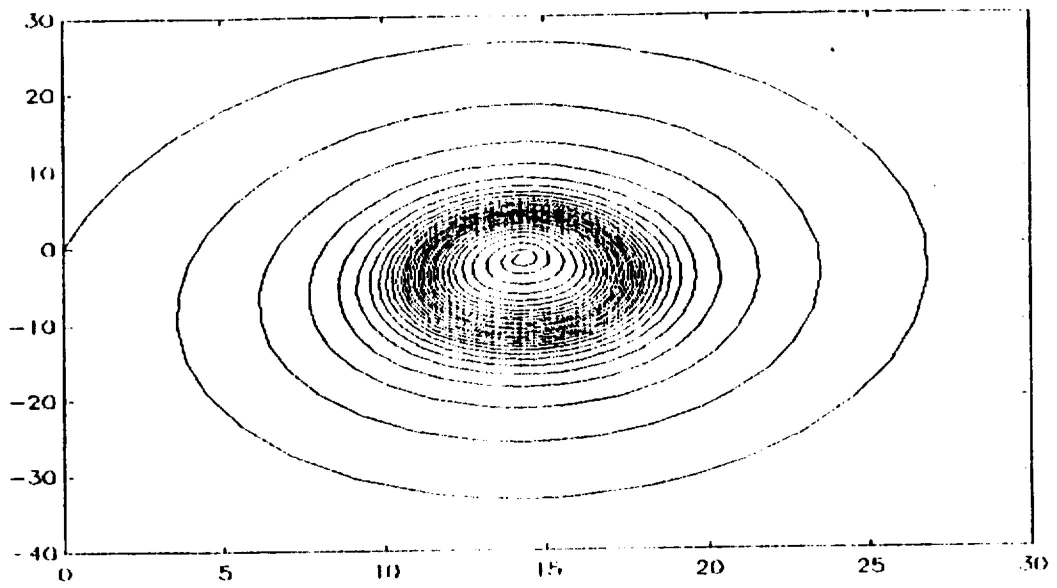


Figure 14. State-Space Portrait, δ_h = -9.2°

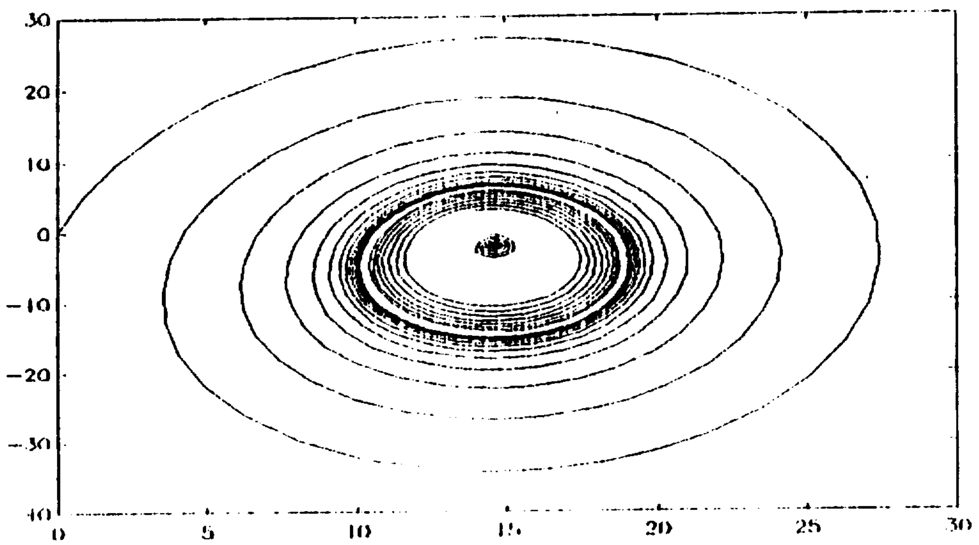


Figure 15. State-Space Portrait, δ_h = -9.4°

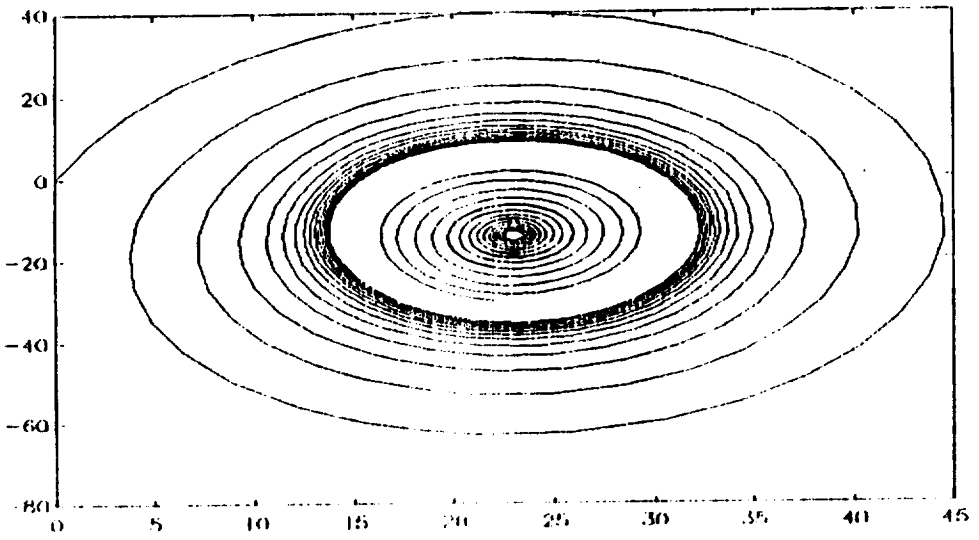


Figure 16. State-Space Portrait, δ_h = -15°

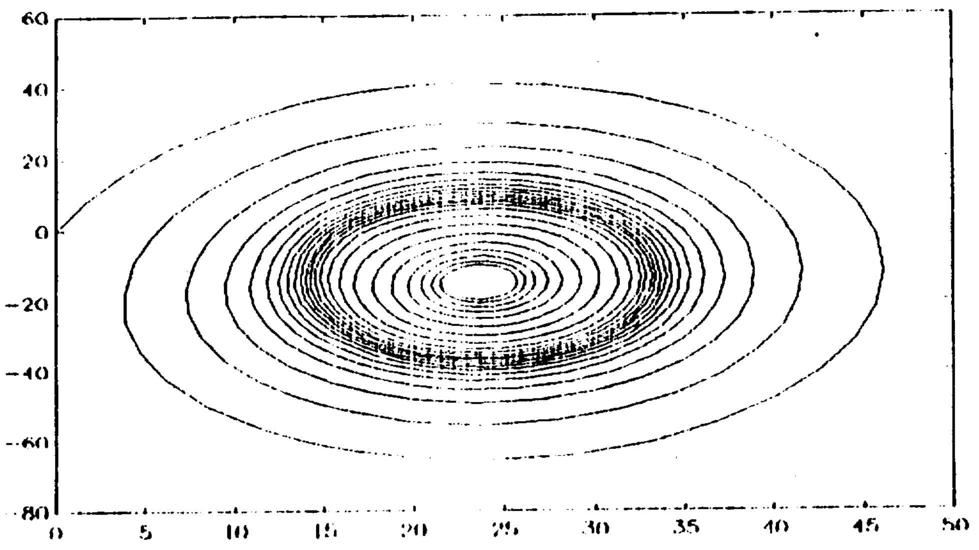


Figure 17. State-Space Portrait, δ_h = -15.5°

separates the areas of attraction of the equilibrium and of the stable limit cycle. So for different values of control and state variables the behavior of the system is essentially different. Some investigation was made about possible ways of characterization of such a behavior by means of a global discrete time nonlinear model. A proposed form of the model was as follows

$$\begin{aligned} \alpha(k+1) &= \sum_{i,j,\ell} p_{ij\ell}\, \alpha(k)^i q(k)^j u(k)^\ell \\ q(k+1) &= \sum_{i,j,\ell} p_{ij\ell}\, \alpha(k)^i q(k)^j u(k)^\ell \end{aligned} \tag{2.36}$$

where the summation is over all possible products of powers that give as a result exponent not greater than N. Of course, for a specific model only some of the terms will be taken - the choice based on á priori knowledge about the systems nonlinearities and the significance of a given coefficient's contribution to the approximation.

In order to identify the model of form (2.36), the experimental data was first collected. The experiment consisted of observing the outputs of the system (given by (2.35)) subject to random steps of control. To capture such phenomena like limits cycles in the data, the steps were rather long - 40 sec. There were 64 such steps. The time discretization was chosen to be 0.1 sec. As a result, the identification data contained 25,600 points in a state plane for 64 values of control. Then, for a few arbitrarily chosen models of form (2.36), the parameters were found by minimization of the quadratic criterion

$$\min_{p}\left(\sum_{k}(y(k)-\hat{y}(k))^2\right) \tag{2.37}$$

where y(k) stands for α or and $\hat{y}(k)$ is obtained from (2.36). Among the models tried, the most accurate approximation of (2.35) was given by the following one:

$$\begin{aligned}
\alpha(k+1) = {} & p_{1\alpha}\alpha(k) + p_{2\alpha}\alpha^2(k) + p_{3\alpha}\alpha^3(k) + \\
& p_{4\alpha}q(k) + p_{5\alpha}q(k)\alpha(k) + p_{6\alpha}q(k)\alpha^2(k) + p_{7\alpha}q(k)\alpha^3(k) + \\
& p_{8\alpha}u(k) + p_{9\alpha}u(k)\alpha(k) + p_{10}u(k)\alpha^2(k) + p_{11\alpha}u(k)\alpha^3(k) + p_{12\alpha} \\
q(k+1) = {} & p_{1q}\alpha(k) + p_{2q}\alpha^2(k) + p_{3q}\alpha^3(k) + \\
& p_{4q}q(k) + p_{5q}q(k)\alpha(k) + p_{6q}q(k)\alpha^2(k) + p_{7q}q(k)\alpha^3(k) + \\
& p_{8q}u(k) + p_{9q}u(k)\alpha(k) + p_{10q}u(k)\alpha^2(k) + p_{11q}u(k)\alpha^3(k) + p_{12q}
\end{aligned} \tag{2.38}$$

with the following values of parameters

$p_{1\alpha}$	=	$8.4320*10^{-1}$	p_{1q}	=	$-5.3094*10^{-1}$
$p_{2\alpha}$	=	$6.7979*10^{-1}$	p_{2q}	=	$-1.1410*10^{-3}$
$p_{3\alpha}$	=	$-1.2527*10^{-4}$	p_{3q}	=	$4.3451*10^{-6}$
$p_{4\alpha}$	=	$9.6900*10^{-2}$	p_{4q}	=	$9.7427*10^{-1}$
$p_{5\alpha}$	=	$5.8142*10^{-4}$	p_{5q}	=	$-4.7215*10^{-5}$
$p_{6\alpha}$	=	$-5.4326*10^{-5}$	p_{6q}	=	$-1.8024*10^{-5}$
$p_{7\alpha}$	=	$1.3799*10^{-6}$	p_{7q}	=	$6.3993*10^{-7}$
$p_{8\alpha}$	=	$-2.2902*10^{-1}$	p_{8q}	=	$-8.2199*10^{-1}$
$p_{9\alpha}$	=	$2.9968*10^{-2}$	p_{9q}	=	$2.3537*10^{-3}$
$p_{10\alpha}$	=	$-1.2158*10^{-4}$	p_{10q}	=	$-5.7795*10^{-5}$
$p_{11\alpha}$	=	$4.2410*10^{-7}$	p_{11q}	=	$-2.6468*10^{-7}$
$p_{12\alpha}$	=	$-3.9140*10^{-1}$	p_{12q}	=	$3.4865*10^{-1}$

Although some of these values seem negligible, it should be noticed that with the values of α going to 20 the terms multiplied by the coefficient in question become of order 10^5 which makes their contributions significant enough. The identified model (2.38) was tested by calculating its responses for the same initial conditions and values of control as used to get state space portraits on Figures 12-17. The comparison between the original data obtained by simulation of (2.35) and the identified model behavior is shown on Figures 18-23. It can be seen that limit cycles are accurately rendered by the model, as well as the stable zone behavior, although large discrepancies occur with the control values close to the stable/unstable zones border. These inaccuracies may be due to insufficient identification data and/or improper choice of nonlinear terms in (2.36). This can be helped by testing the hypotheses about the significance of every particular coefficient based on residuals with and without it. This procedure could be performed once for a given aircraft and the values of resulting set of parameters could then be updated based on on-line identification during the flight itself.

The model (2.38) was then used for developing a nonlinear controller for (2.35).

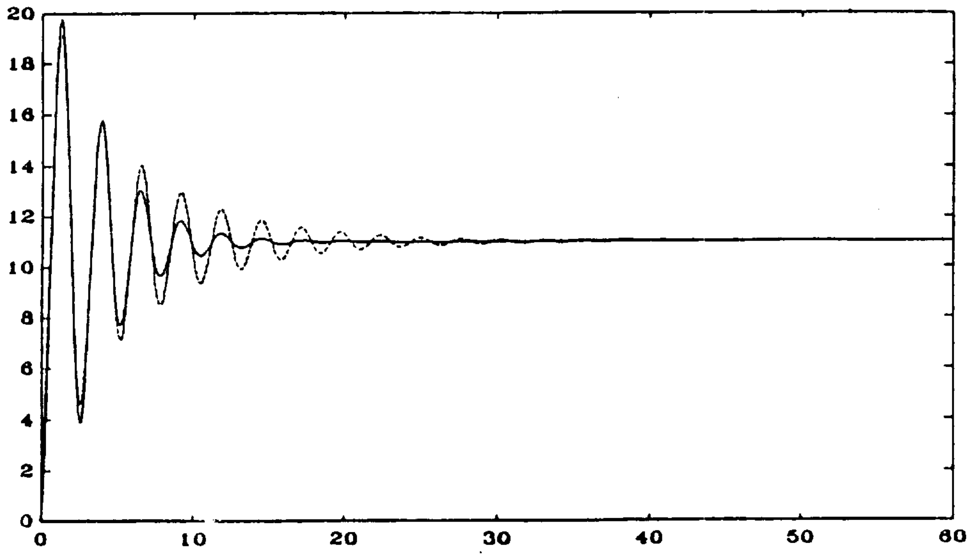


Figure 18. Original vs. Discrete Time Identified Model

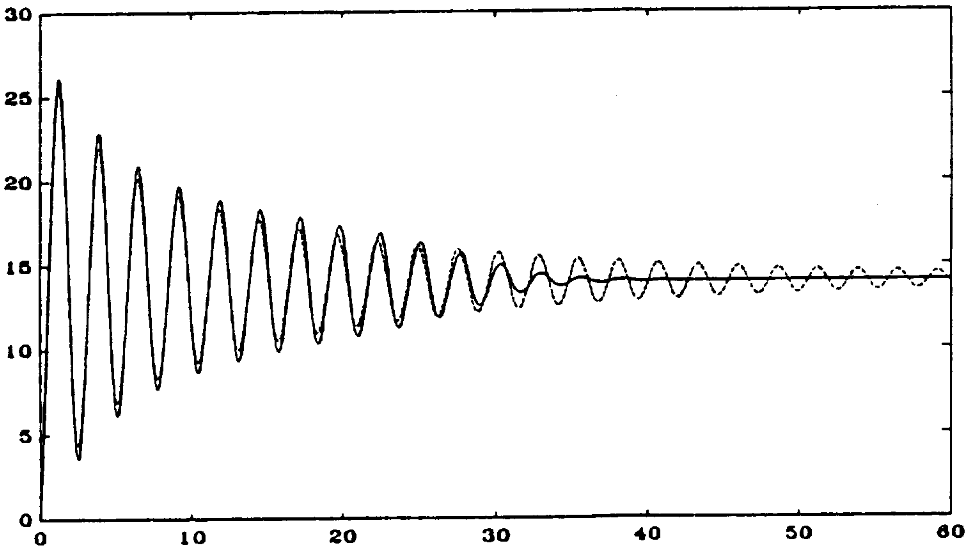


Figure 19. Original vs. Discrete Time Identified Model

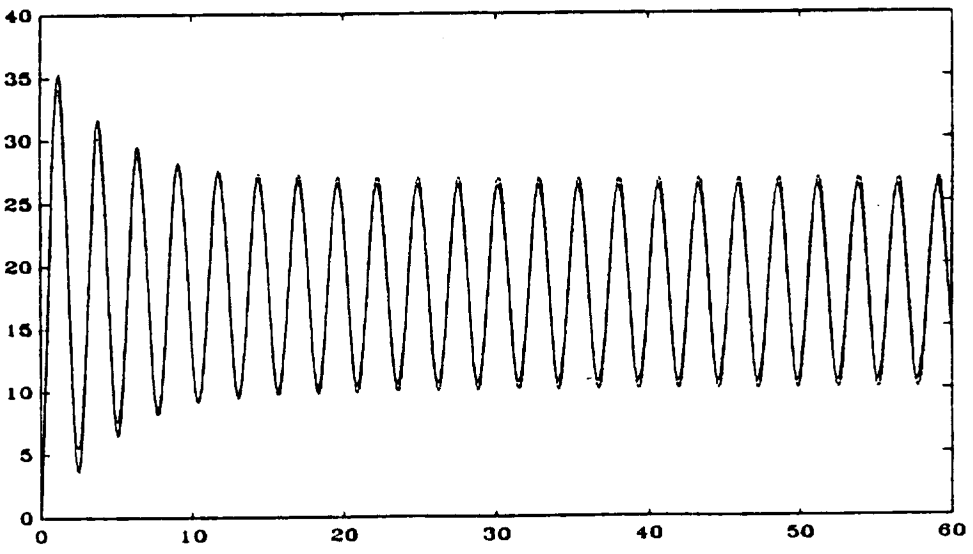


Figure 20. Original vs. Discrete Time Identified Model

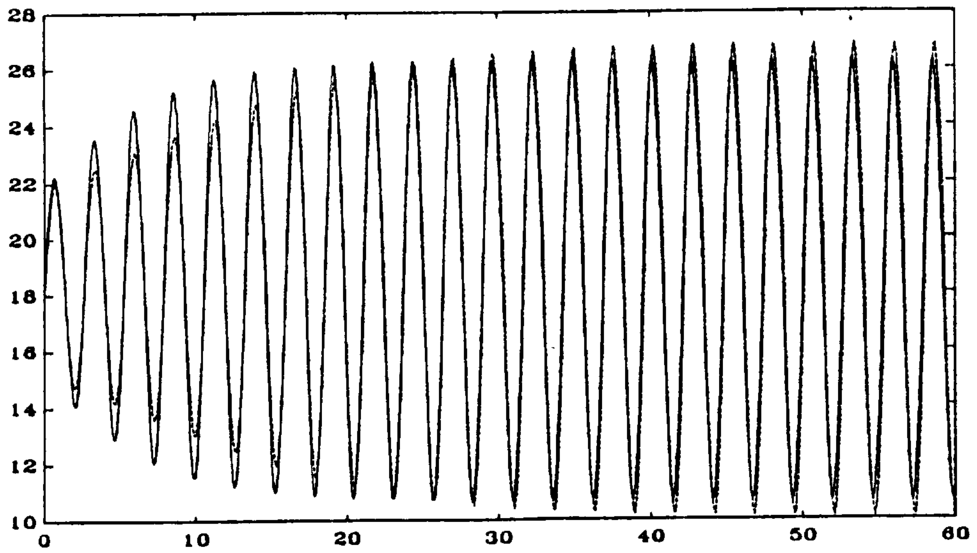


Figure 21. Original vs. Discrete Time Identified Model

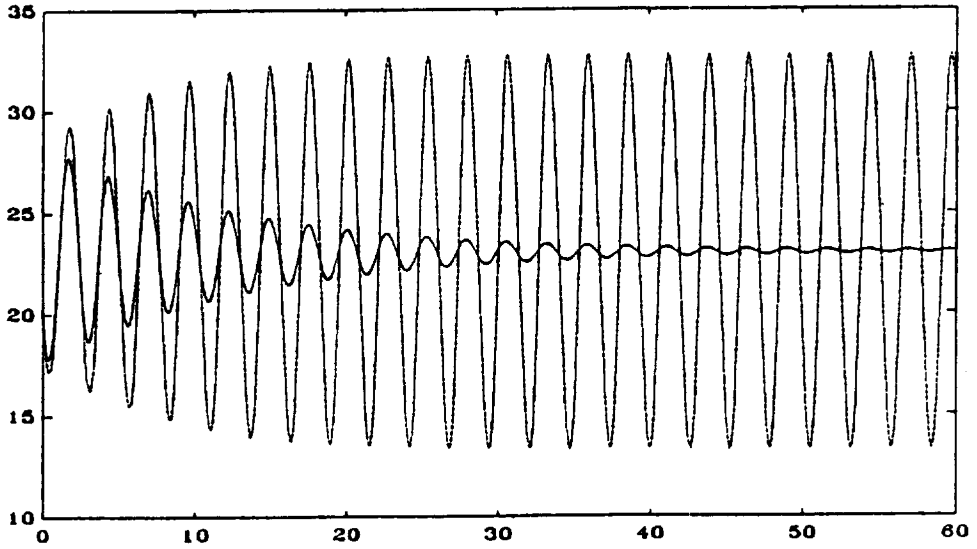


Figure 22. Original vs. Discrete Time Identified Model

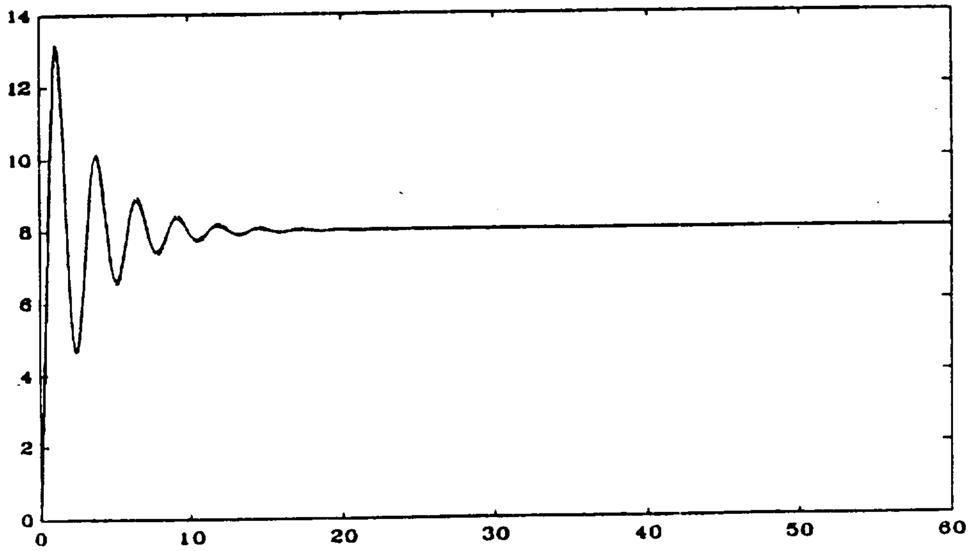


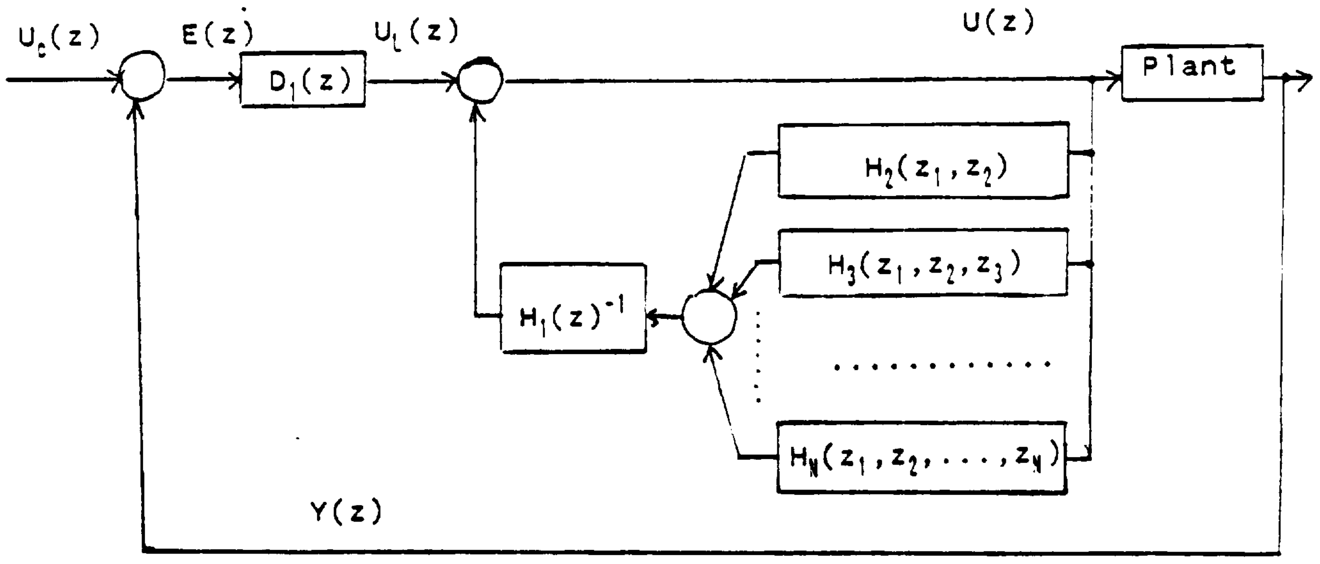
Figure 23. Original vs. Discrete Time Identified Model

3. ADAPTIVE CONTROL APPROACHES

3.1 Nonlinear Volterra-Based Control

As the Volterra series representation of the nonlinear plant dynamics is a natural generalization of the linear system characterization by impulse response or by transfer function, it seems natural to incorporate the concepts of the Volterra series model into the nonlinear control. Although a lot has been published on mathematical theory of Volterra series, including existence, realization, and (to a much smaller extent) identification, very little has been done to apply Volterra series to control. In some papers (e.g., [6]), Volterra series serve only as a conceptual starting point from which a switch follows to discrete time nonlinear time series. Among a few that attempt to build Volterra series controllers, the majority of them deals with discrete time systems. The controllers proposed are mainly predictive ones where the control is obtained by solving the Nth-order polynomial equation [7,8]. The model is given in the form of discrete time kernels which require a tremendous amount of data. The kernels are, of course, truncated at certain time value which results in characteristic jump in the control and the output step response after the time corresponding to the truncation.

Continuous time controllers based on Volterra series were systematically developed in [9] with formulae for the controller's kernels given those of the plant and of the desired feedback system. In particular, the problem of so-called exact feedback linearization was solved here. However, those formulae are of limited practical value because of the properties of Volterra series under feedback. The problem is that even finite (e.g., second-order) Volterra series of the open loop results in infinite Volterra series of the closed loop. This makes it necessary for the controller to include theoretically an infinite number of compensating terms even for a quadratic system. The same problem for the discrete time systems was treated in [10]. Instead of time kernels, they used multidimensional Z transforms and they arrived at the set of formulae equivalent to those in [9]. However, they provided also a very elegant transformation of the exact linearization problem solution which results in a controller requiring only as many Volterra terms as there are in the controlled plant. The control system obtained in [10] is shown below.



$H_1,\ldots,H_N$ are the multidimensional discrete transfer functions of the controlled plant and D_1 is a linear controller designed only for the linear part of the plant, i.e.,

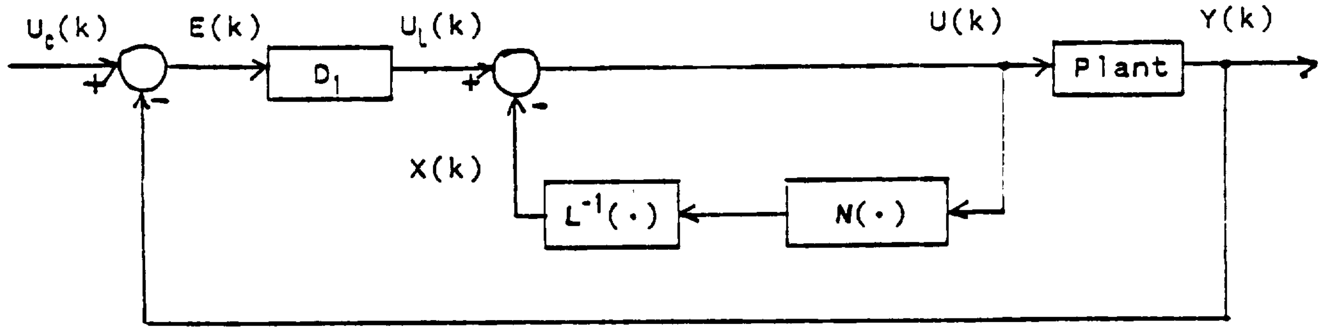
$$D_1(z) = G_{ref}(z)/(H_1(z)(1 - G_{ref}(z)) \tag{3.1}$$

where G_{ref} is the desired transfer function from $U_c(z)$ to $Y(z)$. Thus, with exact knowledge of the nonlinear part of the plant's dynamics, the whole design reduces to the choice of the linear controller using any of well known methods. The inner feedback loop will compensate for all nonlinearities. One of the drawbacks of this method is the necessity of inverting the linear part of the plant. However, if only the linear and nonlinear parts exhibit the same time lag this operation does not represent a major problem. In the configuration of controller shown above finding a control amounts then to solving Nth-order polynomial equation in u. Of course, the problem arises whether and how many solutions exist to this equation. In the case of multiple solutions a rule of thumb would be to choose the one within the operating range of control values. In the lack of solutions (e.g., due to inaccurate modeling), a possible remedy could be, for example, to take the real part of a complex root. (In a few simulations performed with very inaccurate models this method worked surprisingly well.)

One very attractive feature of this controller is that its structure makes it possible to utilize it not only with models represented in the form of Volterra series, but in fact with any model with easily divided linear and nonlinear parts of the dynamic equations. Let us take, for example, a nonlinear time-series model of the form

$$y(k) = \mathcal{L}(y(k-1),...,y(k-M),u(k-1),...,u(k-M)) + \mathcal{N}(y(k-1),...,y(k-M),u(k-1),...u(k-M) \tag{3.2}$$

where L is the linear operator and N is the nonlinear part consisting of higher-order terms (i.e., with all first-order partial derivatives over u and y vanishing at zero). Now the structure of the controller will be as follows:



This diagram is equivalent to the following algorithm for the calculation of the control value at the moment k.

a) calculate the output of the linear controller $u_L(k)$

b) calculate the predicted value of the output at the moment k

$$\hat{y}(k) = \mathcal{L}(y(k-1),...y(k-M),u(k-1),...,u(k-M)) \\ \mathcal{N}(y(k-1),...,y(k-M),u(k-1),...,u(k-M)) \tag{3.3}$$

c) solve the equation for x(k)

$$\mathcal{N}(\hat{y}(k),y(k-1),...,y(k-M+1),u_L(k)-x(k),u(k-1),...,u(k-M+1)) = \\ = \mathcal{L}(x(k),x(k-1),...,x(k-M+1),\hat{y}(k),y(k-1),...,y(k-M+1)) \tag{3.4}$$

d) *calculate the control*

$$u(k) = u_L(k) - x(k)$$

It is worth noting that in such a discrete time realization there is one more feedback loop interaction than shown on the diagram. The models in the inner feedback loop do not use the previous values of output estimates $\hat{y}(k-1)$, $\hat{y}(k-2)$, ... (as would be necessary in the continuous time case) but the real measured values of output. So it becomes clear that the above algorithm becomes a sort of prediction controller which tries to estimate the effects of the previous controller which tries to estimate the effects of the previous controls knowing the previous values of outputs and then to adjust the current value of control so that the nonlinear part of predicted output is canceled.

Nonlinear Control of the Longitudinal Aircraft

The discrete time nonlinear control algorithm presented above was used for angle of attack stabilization and control of the nonlinear longitudinal aircraft model described in Section 2.3. For the purpose of controller design, the model (2.38) was used with the parameter set derived from an off-line identification process. The linear controller D_1 was designed for the linear model with controlled output chosen to be

$$\alpha_L(k+1) = p_{1\alpha}\alpha_L(k) + p_{4\alpha}q_L(k) + p_{8\alpha}u(k)$$
$$q_L(k+1) = p_{1q}\alpha_L(k) + p_{4q}q_L(k) + p_{8q}u(k)$$

The design was performed to obtain the closed loop behavior of the form

$$G(z) = 0.05/(z^2 - 1.6z + 0.65)$$

In order not to cancel the zero of the plant, the observer polynomial (z-0.7) was also introduced. The algorithm for the control value u(k) is as follows. First the estimate of the output at moment k is calculated.

$$
\begin{aligned}
\hat{\alpha}(k) = {} & p_{1\alpha}\alpha_1 + p_{2\alpha}\alpha_1^2 + p_{3\alpha}\alpha_1^3 + \\
& p_{4\alpha}q_1 + p_{5\alpha}q_1\alpha_1 + p_{6\alpha}q_1\alpha_1^2 + p_{7\alpha}q_1\alpha_1^3 + \\
& p_{8\alpha}u_1 + p_{9\alpha}u_1\alpha_1 + p_{10\alpha}u_1\alpha_1^2 + p_{11\alpha}u_1\alpha_1^3 + p_{12\alpha} \\
\hat{q}(k) = {} & p_{1q}\alpha_1 + p_{2q}\alpha_1^2 + p_{3q}\alpha_1^3 \\
& p_{4q}q_1 + p_{5q}q_1\alpha_1 + p_{6q}q_1\alpha_1^2 + p_{7q}q_1\alpha_1^3 + \\
& p_{8q}u_1 + p_{9q}u_1\alpha_1 + p_{10q}u_1\alpha_1^2 + p_{11q}u_1\alpha_1^3 + p_{12q}
\end{aligned}
\tag{3.5}
$$

where $\alpha_1 = \alpha(k\text{-}1)$, $q_1 = q(k\text{-}1)$, $u_1 = u(k\text{-}1)$.

The linear portion of this estimate will be

$$
\begin{aligned}
\hat{\alpha}_L(k) &= p_{1\alpha}\alpha_1 + p_{4\alpha}q_1 + p_{8\alpha}u_1 \\
\hat{q}_L(k) &= p_{1q}\alpha_1 + p_{4q}q_1 + p_{8q}u_1
\end{aligned}
$$

and the nonlinear portion

$$
\begin{aligned}
\hat{\alpha}_N(k) &= \hat{\alpha}(k) - \hat{\alpha}_L(k) \\
\hat{q}_N(k) &= \hat{q}(k) - \hat{q}_L(k)
\end{aligned}
$$

Then the nonlinear portion of the output estimate in the moment k+1 given control u(k) is equal:

$$
\begin{aligned}
\hat{\alpha}_N(k+1) &= \hat{\alpha}(k+1) - \hat{\alpha}_L(k+1) \\
&= p_{1\alpha}(\alpha - \alpha_L) + p_{2\alpha}\alpha^3 + p_{4\alpha}(q - q_L) + p_{5\alpha}q_1 + p_{6\alpha}q\alpha^2 + p_{7\alpha}q\alpha^3 + \\
&\quad p_{9\alpha}u + p_{10\alpha}u\alpha^2 + p_{11\alpha}u\alpha^3 + p_{12\alpha}
\end{aligned}
$$

where $\alpha = \hat{\alpha}(k)$, $\alpha_L = \hat{\alpha}_L(k)$, $q = \hat{q}(k)$, $q_L = \hat{q}_L(k)$, $u = u(k)$.

The control value comes from the equation

$$
\hat{\alpha}_N(k+1) = p_{1\alpha}\hat{\alpha}_N(k) + p_{4\alpha}\hat{q}_N(k) + p_{8\alpha}(u_L(k) - u(k))
$$

which finally yields a solution

$$u(k) = \frac{p_{8\alpha}u_L(k) - \left(p_{2\alpha}\hat{\alpha}^2 + p_{3\alpha}\hat{\alpha}^3 + p_{5\alpha}\hat{\alpha}q + p_{6\alpha}\hat{q}\hat{\alpha}^2 + p_{7\alpha}\hat{q}\hat{\alpha}^3 + p_{12\alpha}\right)}{\left(p_{8\alpha} + p_{9\alpha}\hat{\alpha} + p_{10\alpha}\hat{\alpha}^2 + p_{11\alpha}\hat{\alpha}^3\right)} \tag{3.6}$$

with $\hat{\alpha}(k)$ and $\hat{q}(k)$ taken form (3.5). It is seen that if there are no nonlinearities in the model the control reduces to a regular linear controller $u = u_L$.

A number of simulations was run to test the controller performance, especially in the unstable range of angle of attack. Figures 24-28 show the response of the system to the step change of the setpoint of angle of attack. The resulting trajectories are compared with desired trajectories following form the linear controller design. It can be noticed that modeling inaccuracies do not achieve prefect model following but, nevertheless, the system is successfully stabilized and the transients are very smooth and without significant overshoots. By different choice of the reference model it is possible to obtain much faster, but at the same time much more "nervous" transients. The elevator control as shown as an example on Figure 28a is also relatively smooth and, worth noting, its values doe not at all come out from the range corresponding to the terminal equilibria. This cautiousness of the controller is the main reason for rather slow regulation process. It also can be noticed that some kind of linearizing the closed loop system was indeed accomplished because the shape of trajectories is very similar regardless of the zone in which the regulation takes place. The reaction of the system for an input disturbance in the form of an impulse of magnitude -1° additive to the control (i.e., sudden displacement of elevator) is depicted in Figures 29-31. The performance is not astonishingly good but still the task of stabilization and disturbance rejection is successfully fulfilled. Of course, purely linear controller constant on the whole operation range is not able to stabilize and control the plant as can be seen in Figures 32-33. The linear model used for its design was obtained by identification from the same data as in the case of model (2.38).

Conclusions

The conclusion that comes from the above simulation experiments is that it is possible to model nonlinear aircraft dynamics in the form of nonlinear discrete time model containing a limited number of power nonlinearities. The proposed nonlinear controller structure was shown to give quite satisfactory

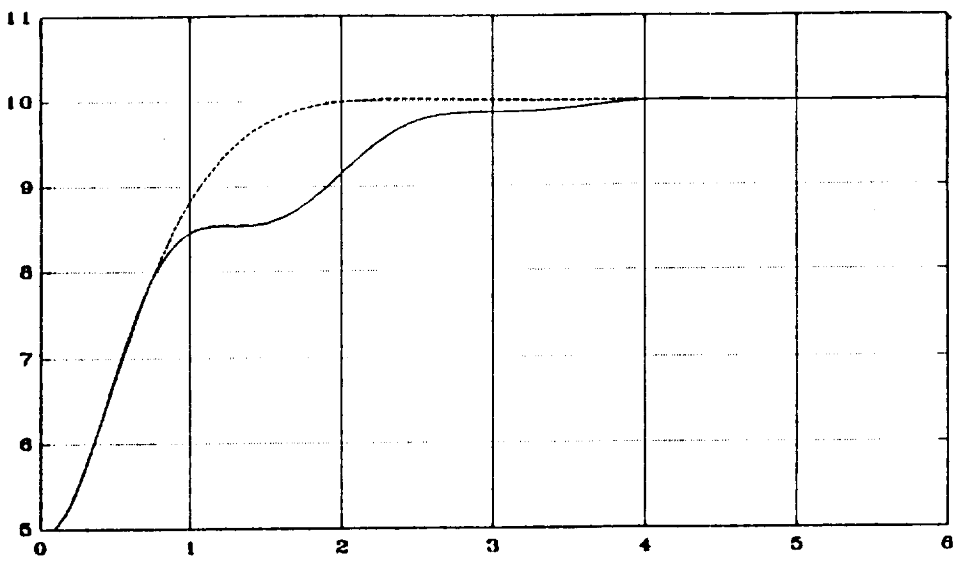


Figure 24. Step Response with Nonlinear Controller vs. Nominal Response

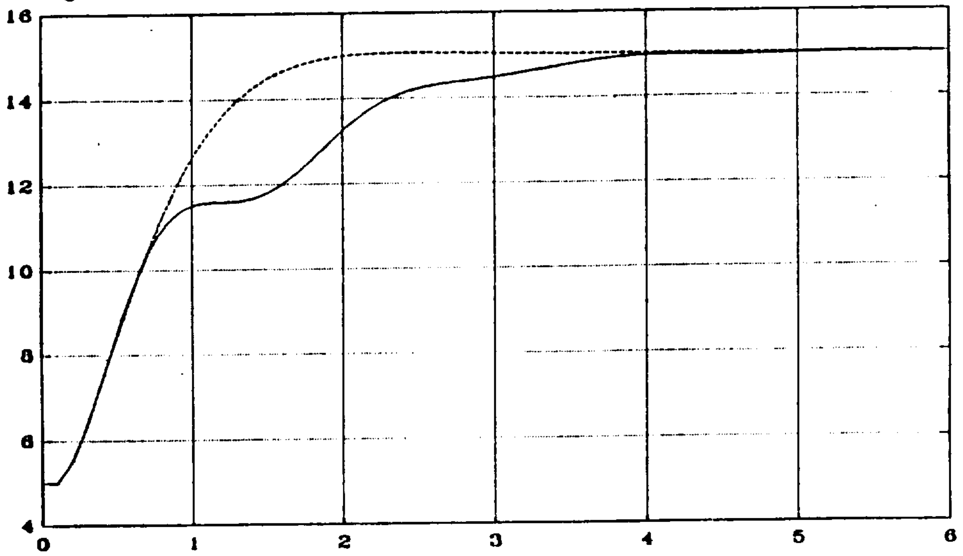


Figure 25. Step Response with Nonlinear Controller vs. Nominal Response

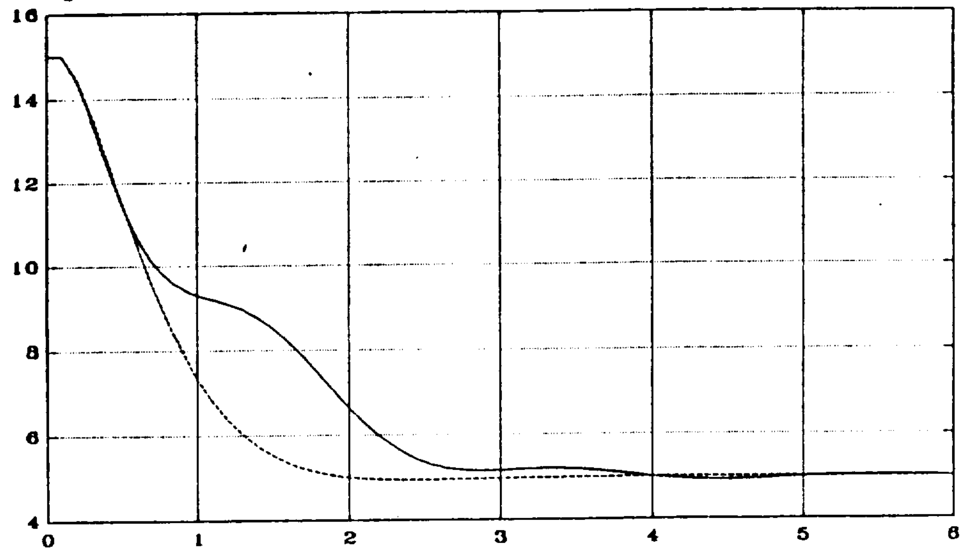


Figure 26. Step Response with Nonlinear Controller vs. Nominal Response

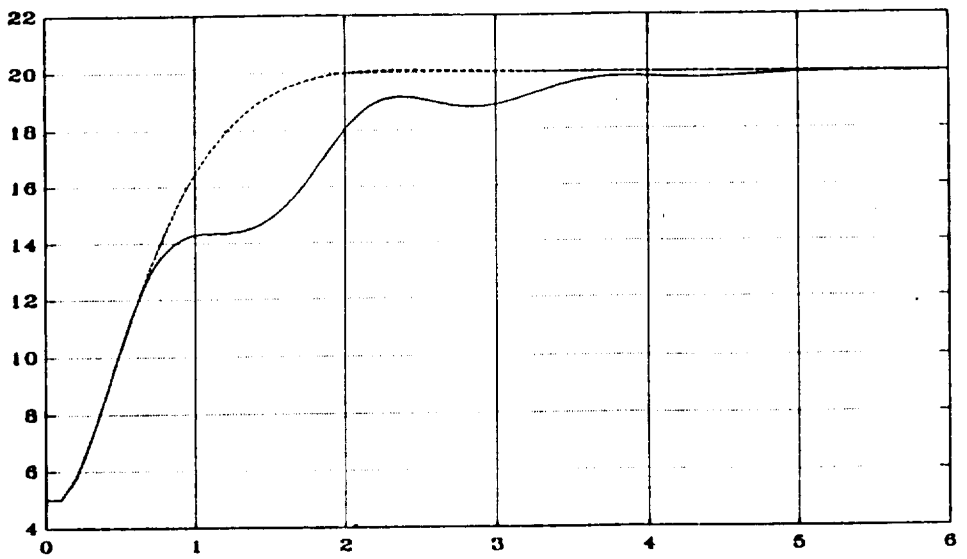


Figure 27. Step Response with Nonlinear Controller vs. Nominal Response

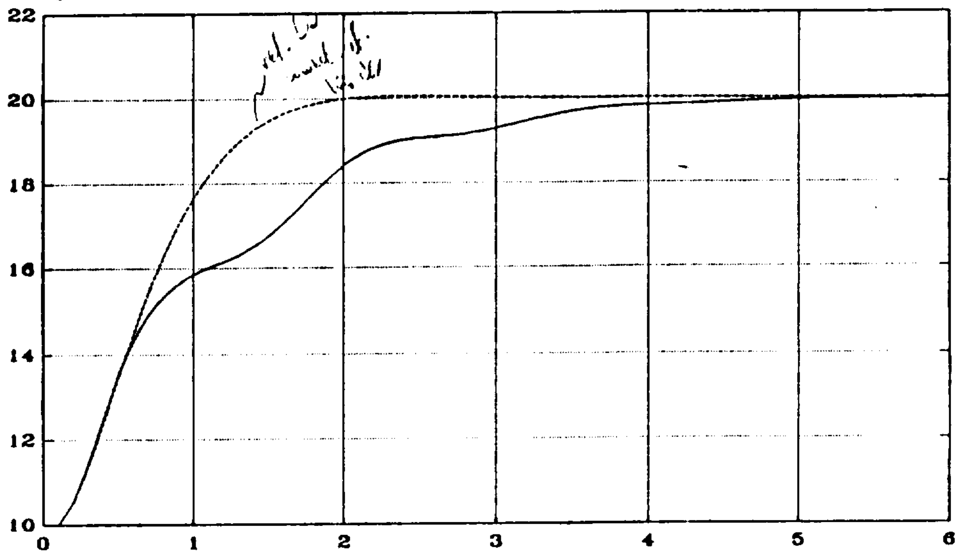


Figure 28. Step Response with Nonlinear Controller vs. Nominal Response

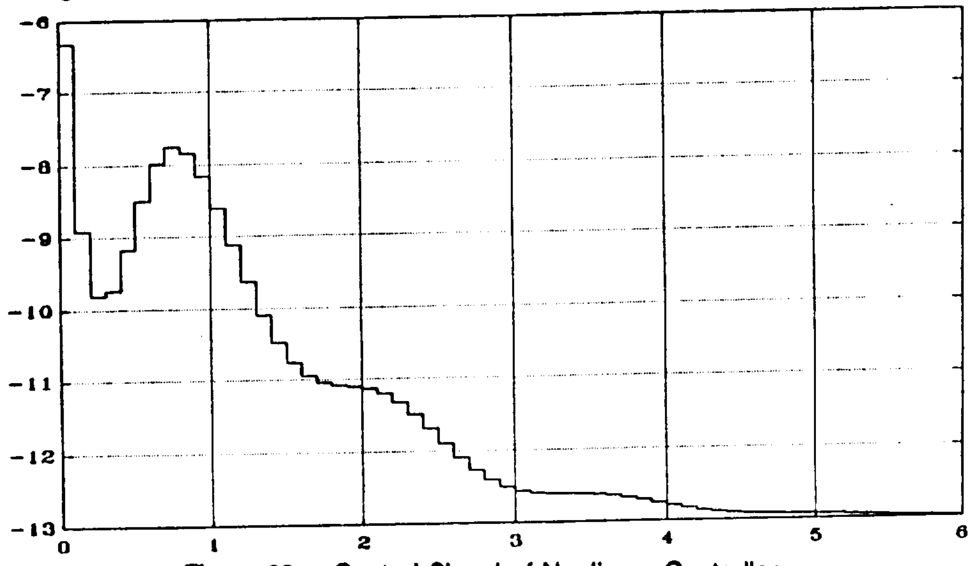


Figure 28a. Control Signal of Nonlinear Controller

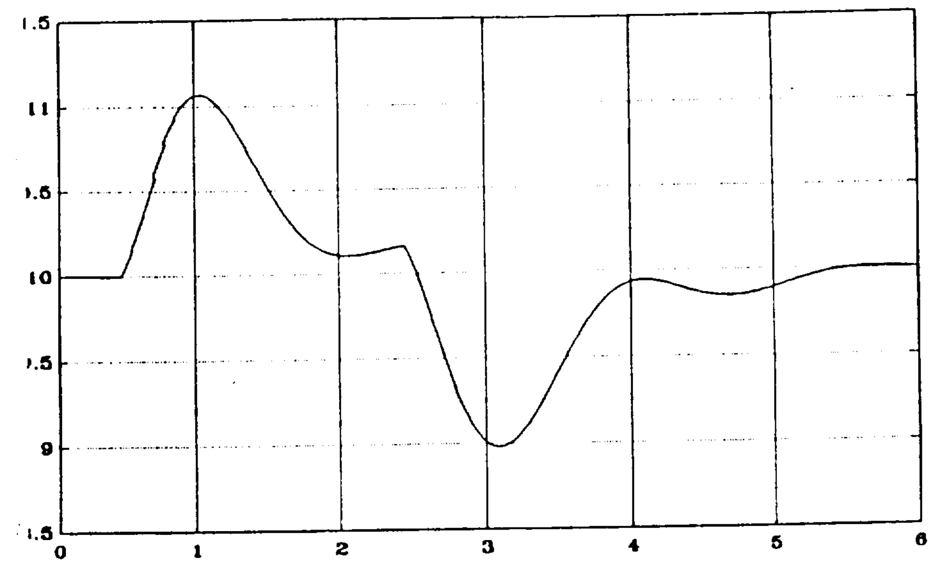


Figure 29. Response to Control Disturbance Impulse (0.45-2.45 sec)

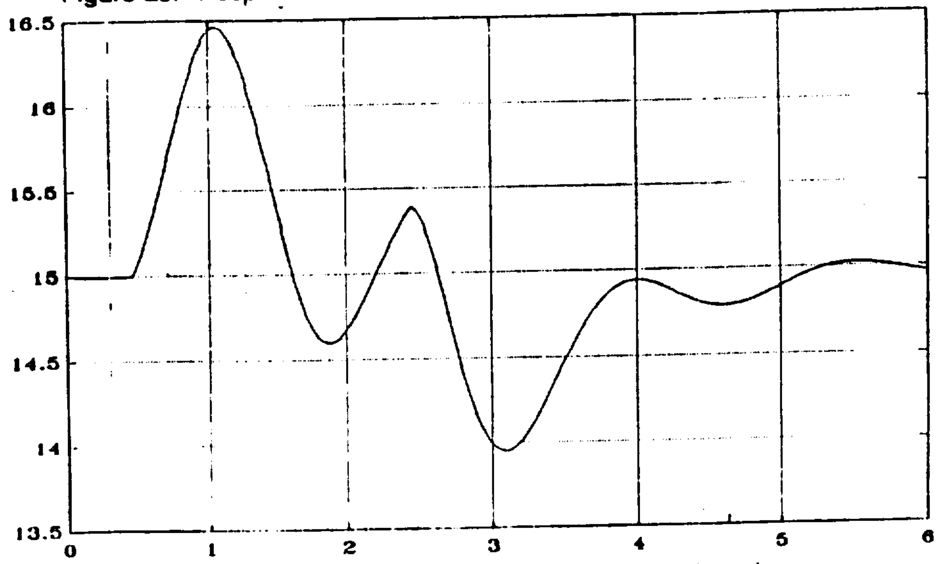


Figure 30. Response to Control Disturbance Impulse

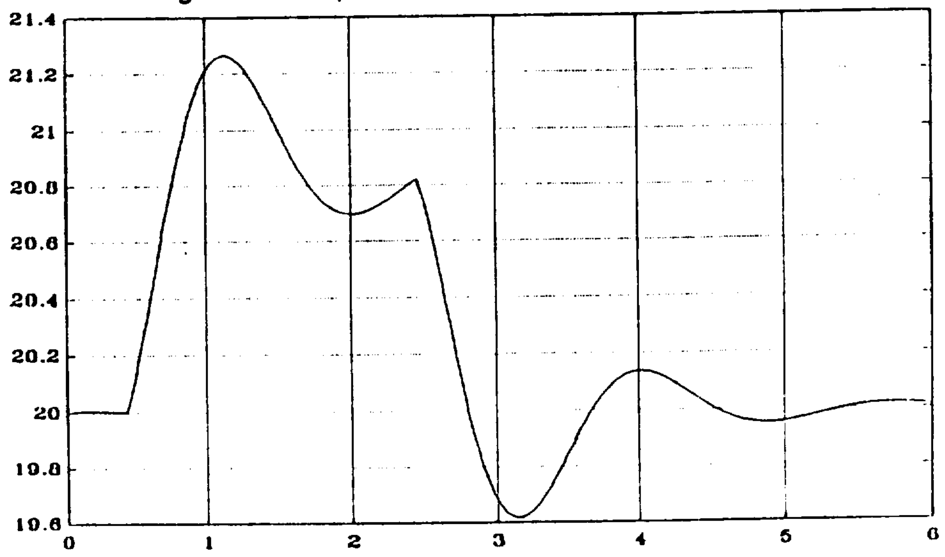


Figure 31. Response to Control Disturbance Impulse

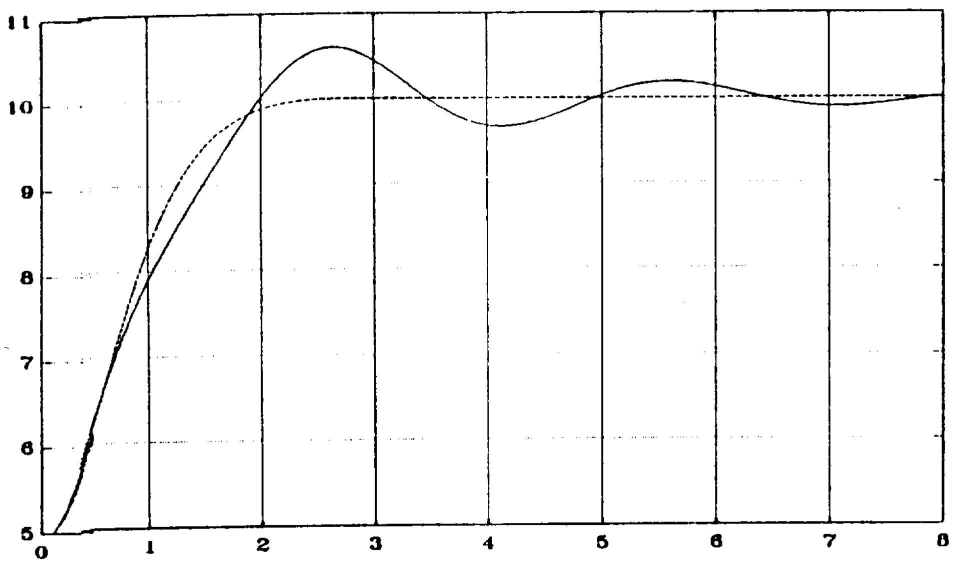


Figure 32. Step Response With Linear Controller vs. Nominal Response

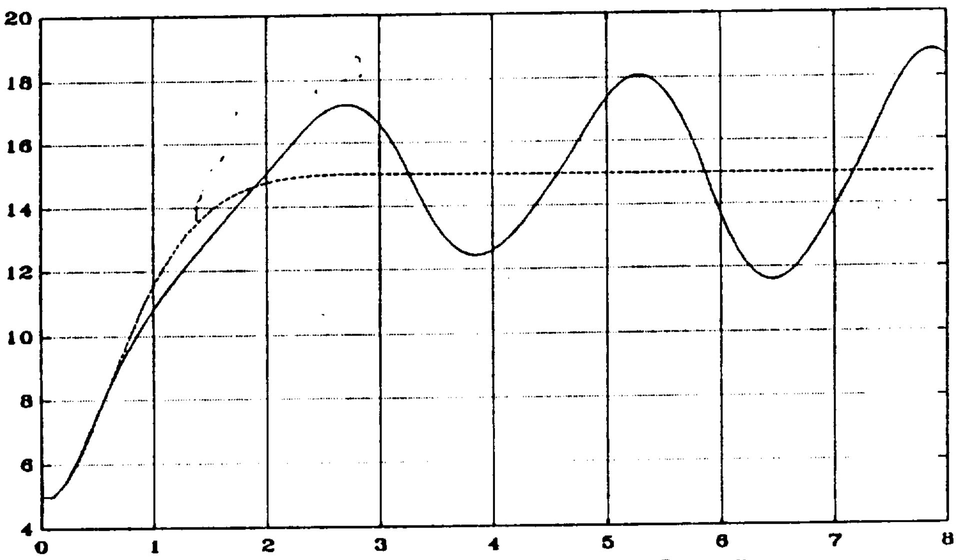


Figure 33. Step Response with Linear Controller

results even with the model that was far from perfect. This is, however, a purely empirical result and theoretical investigation of such properties of the discussed controller like stability sensitivity of modeling errors would be in order. Furthermore, a more accurate discrete time nonlinear presentation of nonlinear aircraft dynamics is necessary.

All simulations included in this section were obtained using PC-MATLAB and its Runge-Kutta integrating routine. The minimization of model square error for purpose of identification was performed using Nelder-Mead nonlinear simplex method coded in C and compiled by TURBO-C compiler.

3.2 Nonlinear MAC Algorithm

Model algorithmic control (MAC), described, for example, in [11], consists in general in solving the model equation for the value of control necessary to obtain required value of output. Usually this desired output trajectory is generated from the setpoint by means of a reference model. In case when this model is linear the algorithm in essence becomes a linearizing one. More precisely, the equation to be solved is

$$Y_{ref}(k+1) = y_{mod}(k+1) + (y(k) - y_{mod}(k)) \tag{3.7}$$

where y_{ref} is a desired reference output and y_{mod} is the prediction of the output based on the model of the plant. The correction term $(y(k) - y_{mod}(k))$ takes into consideration the possible error of the model and, in fact, introduces integral action into control. In a situation when y(k) is not yet available at the time when the control u(k) is computed, as is often the case due to time delays and/or time needed for solving (3.7), the correction term may be taken as $(y(k\text{-}1) - y_{mod}(k\text{-}1))$, and $y_{mod}(k+1)$ must be based on the measurements from moment k-1 which means that the algorithm becomes two-steps-ahead.

In case of model (2.35) with the controlled output assumed to be the angle of attack the algorithm takes the form:

$$\alpha_{ref}(k+1) = \alpha_{mod}(k+1) + (\alpha(k) - \alpha_{mod}(k)) \tag{3.8}$$

with

$$\alpha_{mod}(k+1) = p_a^T \phi(k) \tag{3.9}$$

$$\phi(k) = [\alpha, \alpha^2, \alpha^3, q, q\alpha, q\alpha^2, q\alpha^3, u, u\alpha, u\alpha^2, u\alpha^3, 1]^T(k) \tag{3.10}$$

As the control at the moment k must be already computed at moment k the values of $\alpha(k)$ and q(k) are not available for its computation so their estimates must be used instead. The correction term is taken to be the prediction error from the moment k-1 and the equation becomes

$$\alpha_{ref}(k+1) = \hat{\alpha}_{mod}(k+1) + (\alpha(k-1) - \alpha_{mod}(k-1)) \tag{3.11}$$

with

$$\hat{\alpha}_{mod}(k+1) = p_\alpha^T \hat{\phi}(k)$$

$$\hat{\phi}(k) = [\hat{\alpha}, \hat{\alpha}^2, \hat{\alpha}^3, q, q\hat{\alpha}, q\hat{\alpha}^2, q\hat{\alpha}^3, u, u\hat{\alpha}, u\hat{\alpha}^2, u\hat{\alpha}^3, 1]^T(k)$$

$$\hat{\alpha}(k) = p_\alpha^T \phi(k-1) + (\alpha(k-1) - \alpha_{mod}(k-1))$$

$$\hat{q}(k) = p_q^T \phi(k-1) + (q(k-1) - q_{mod}(k-1))$$

The controller is assumed to know the values of angle of attack and of pitch rate at the moment k-1. Then it estimates their current values $\alpha(k)$ and q(k) taking into consideration previous prediction errors and based on them it calculates the control required to achieve α_{ref} at the moment k+1. The value of control is found as:

$$u(k) = \frac{\tilde{\alpha}_r - p_{1\alpha}\hat{\alpha} - p_{2\alpha}\hat{\alpha}^2 - p_{3\alpha}\hat{\alpha}^3 - p_{4\alpha}\hat{q} - p_{5\alpha}\hat{q}\hat{\alpha} - p_{6\alpha}\hat{q}\hat{\alpha}^2 - p_{7\alpha}\hat{q}\hat{\alpha}^3 - p_{12\alpha}}{p_{8\alpha} + l p_{9\alpha}\hat{\alpha}\ p_{10\alpha}\hat{\alpha}^2 + p_{11\alpha}\hat{\alpha}^3} \tag{3.12}$$

where

$$\tilde{\alpha}_r = \alpha_{ref}(k+1) - (\alpha(k-1) - \alpha_{mod}(k-1)) \tag{3.13}$$

and $\hat{\alpha} = \hat{\alpha}(k)$, $\hat{q} = \hat{q}(k)$ as described above.

This algorithm was simulated for the plant (2.35) with model (2.38) and its parameter values. The results of the simulations are seen in Figures 34-35. The reference trajectory was chosen to be $1/(z^2 - 1.6z + 0.65)$. The actual output of the plant is seen to follow the reference very closely, even though the region of operation was that of the most severe nonlinearities. The control action is also remarkably smooth. It should be pointed out that for all simulations presented here setpoints of α correspond to equilibria with some negative pitch rate and in reality would result in some decrease of pitch angle, which is not included in the model (2.35). Thus, the conditions simulated are somewhat fictitious from the aeronautical point of view. Nevertheless, for the purpose of evaluating the performance of control strategies

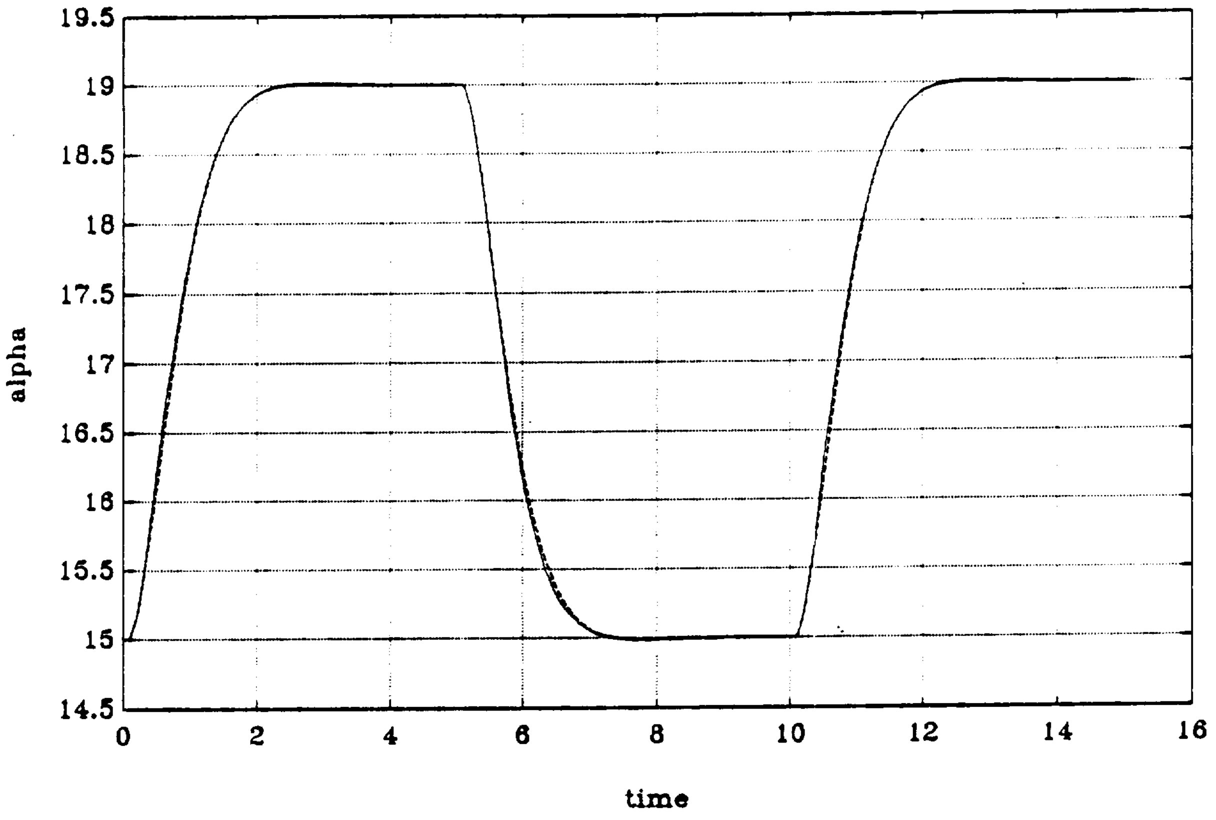


Figure 34. Nonlinear MAC α, δ_h Responses

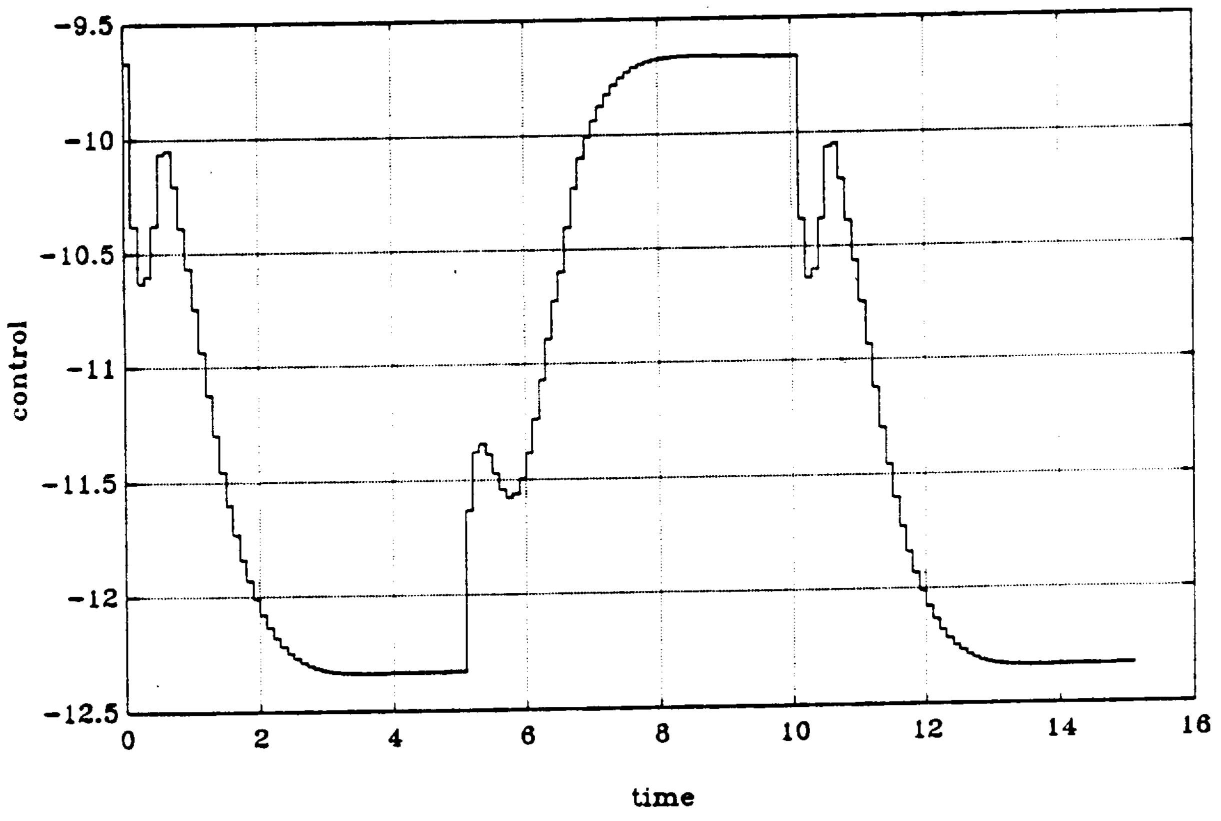


Figure 35. Nonlinear MAC α, δ_h Responses

they may provide useful information about the behavior of the system in nonlinear regime. However, more detailed study is required.

3.3 Adaptive MAC

The discrete time nonlinear state space model (2.36) describes the behavior of the plant (2.35) quite accurately in the entire region of operation. Very often, however, such a global model is rather difficult to fit and, consequently, one should look for local approximations, depending on the current operating conditions. In such a situation, adaptive control seems to offer an ideal solution. For a given model structure the controller identifies its local parameters and appropriately adjusts its action. At the same time it can compensate for the changes of the plant "true" global parameters. Linear adaptive control has been used for nonlinear systems trying to modify the first-order approximation depending on the operating conditions treating the nonlinear system as a time-varying linear one. In general this approach requires that the plants parameters change slowly, which excludes the case of rapid maneuvers. Nonlinear adaptive control is believed to be a proper solution to this problem. While it may be difficult to find a suitably simple global approximate model, local behavior may be still highly nonlinear thus causing the linear control to fail.

The algorithm discussed in the previous section was made to be adaptive, or self-tuning, by incorporating on-line identification of the parameters. A recursive least squares (RLS) algorithm was implemented in the form taken from

$$p(k) = \frac{Q(k-2)\,\phi(k-1)}{\lambda(k-1) + \phi(k-1)^T Q(k-2)\,\phi(k-1)} \tag{3.14}$$

$$Q(k-1) = \frac{1}{\lambda(k-1)}\left(Q(k-2) - \frac{Q(k-2)\,\phi(k-1)\,\phi(k-1)^T Q(k-2)}{\lambda(k-1) + \phi(k-1)^T Q(k-2)\,\phi(k-1)}\right) \tag{3.15}$$

$$e(k-1) = y(k) - p^T \phi(k-1) \tag{3.16}$$

where y may denote α or q and p may stand for p_α or p_q, respectively. The forgetting factor λ was introduced to enable the algorithm to change the estimates of parameters with the change of operating conditions. To avoid the unlimited growth of covariance matrix Q at the steady state when the input is not persistently exciting, the variable forgetting factor policy was implemented:

$$\lambda(k) = 1 - \epsilon \frac{e(k)^2}{\bar{e}(k)^2} \tag{3.17}$$

where e(k) is the current prediction error, $\bar{e}(k)$ is the acreage prediction error from last 10 samples and ϵ is equal to 0.01. As an additional precaution the trace of the covariance matrix Q was monitored and Q was reset to diagonal matrix whenever the threshold value was exceeded. Starting values of parameters were taken to be as in (2.36).

Figures 36-37 show the simulation results of the above algorithm for the same reference trajectory and initial conditions as discussed in the previous section. Figures 38-43 display the simulation results for another reference model specified as $1/(z^2 - 1.8 + 0.82)$. Remarkably exact following of the reference trajectory may be observed, although, surprisingly enough, the performance is slightly worse than in the nonadaptive case. Most probably, this is due to the fact that prediction error now changes much more quickly because of the ongoing identification process. Thus, approximating the term $(y(k+1) - y_{mod}(k+1))$ by $(y(k-1) - y_{mod}(k-1))$ may worsen the behavior of the system as two values of y_{mod} no longer correspond to the same parameter vector. Since the on-line identification process assures (at least in principle) that the prediction error should asymptotically converge to zero, it is possible that the correction terms in $\hat{\alpha}(k)$, $\hat{q}(k)$ and in control equation (3.11) ought to be omitted. This will be soon verified in proper simulation experiments.

The performance of the adaptive nonlinear MAC controller was compared to the linear one, which uses the same control strategy but with strictly linear model being identified and used for the calculation of the control action. The simulation results are shown in Figures 47-51. The initial conditions and reference trajectories were exactly the same as the ones for the nonlinear case in Figures 36-44. The starting values of model parameters were taken from off-line identification over the entire region of interest (similarly to those of model (2.35)). Clear difference between the performance of linear and nonlinear controller can be seen in Figures 40-41 and 48-49, particularly in control action at the setpoint $\alpha = 15°$. The linear identifier has

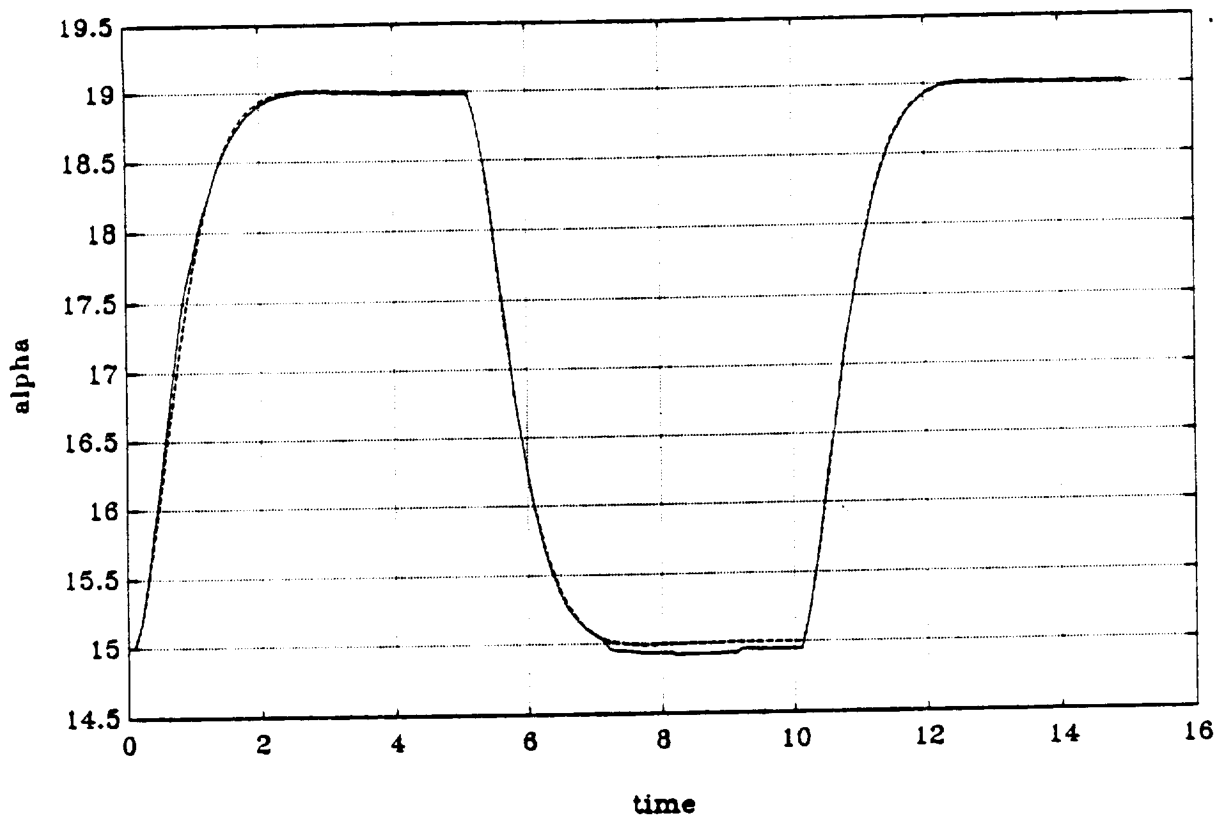


Figure 36. Nonlinear Adaptive MAC (with Reference Trajectory)

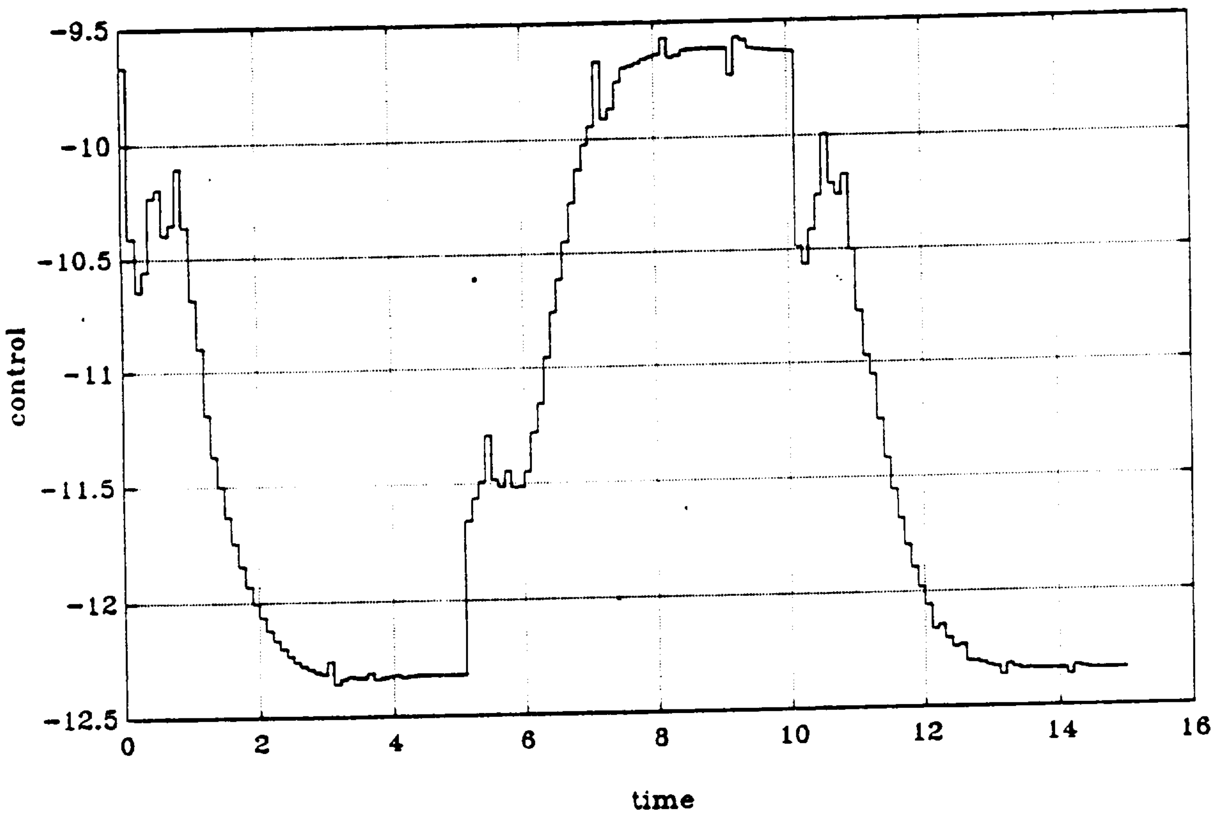


Figure 37. Nonlinear Adaptive MAC

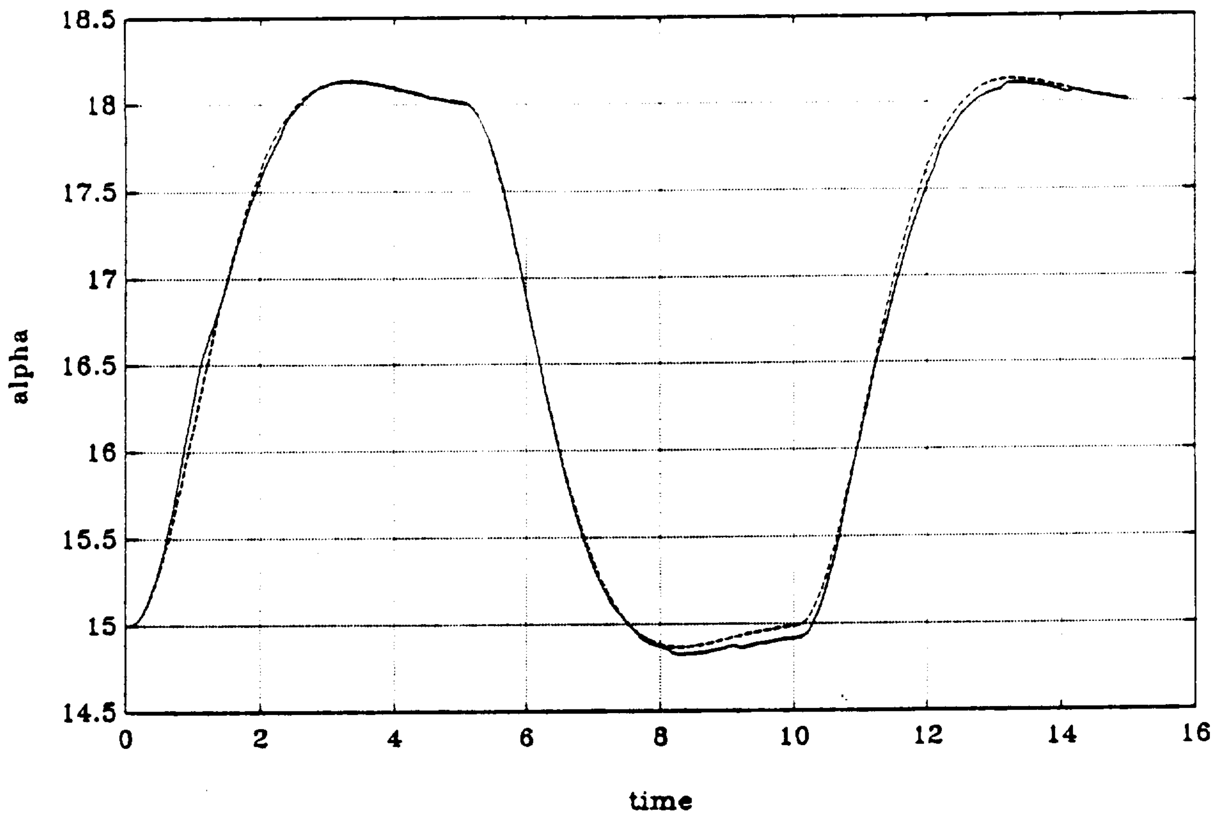


Figure 38. Nonlinear Adaptive MAC (with Reference Trajectory)

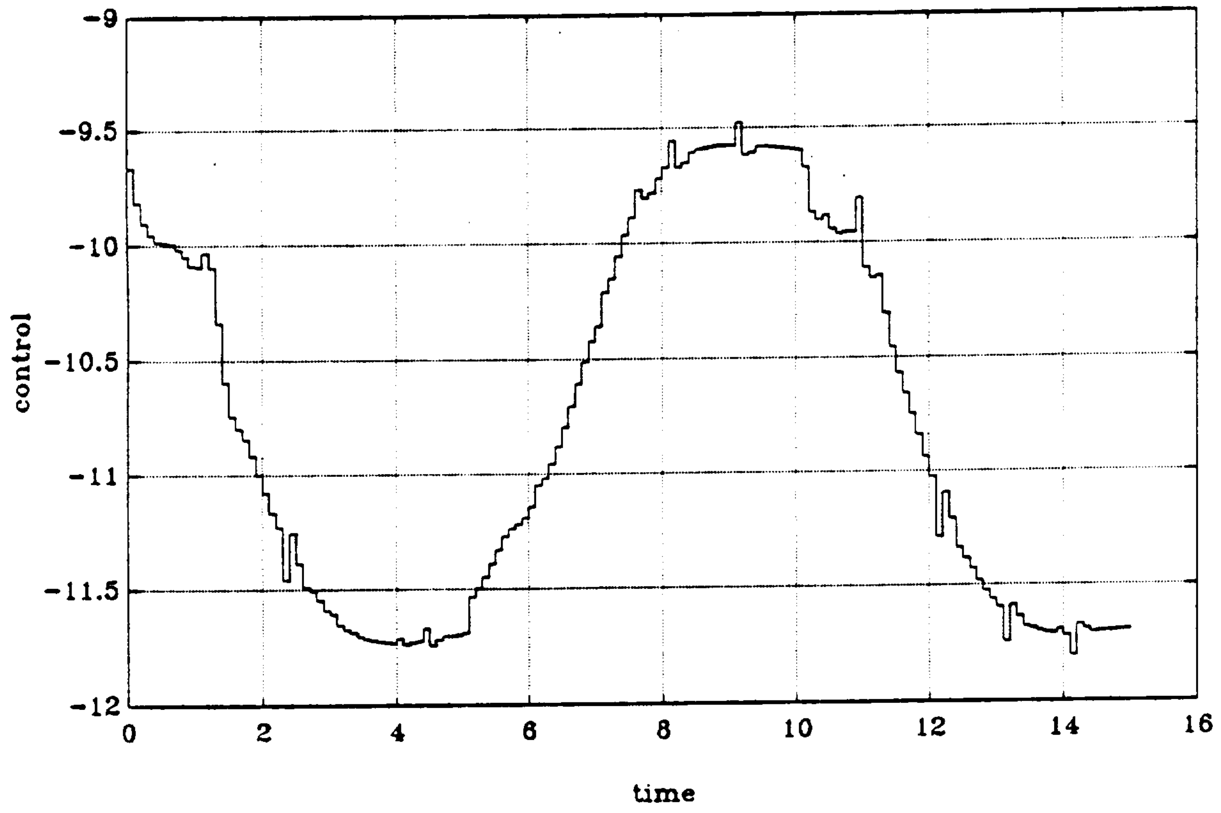


Figure 39. Nonlinear Adaptive MAC

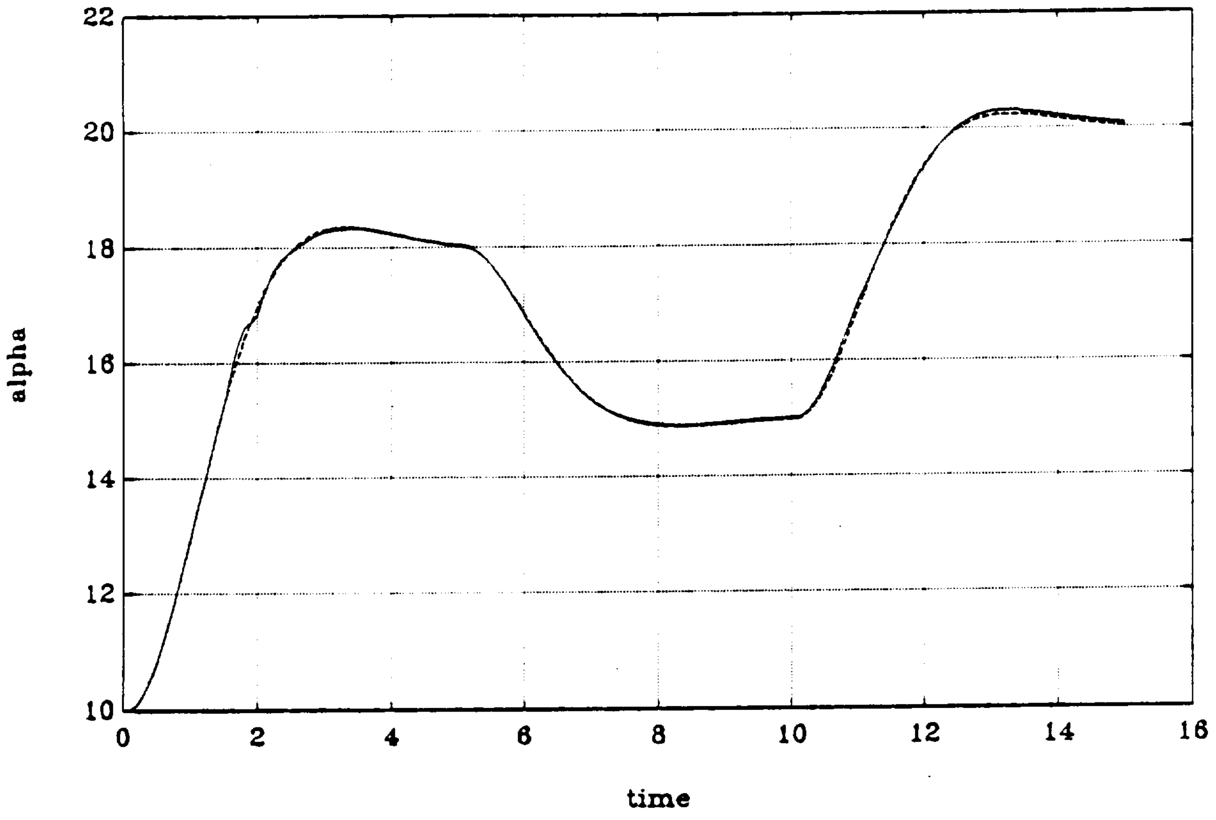


Figure 40. Nonlinear Adaptive MAC (with Reference Trajectory)

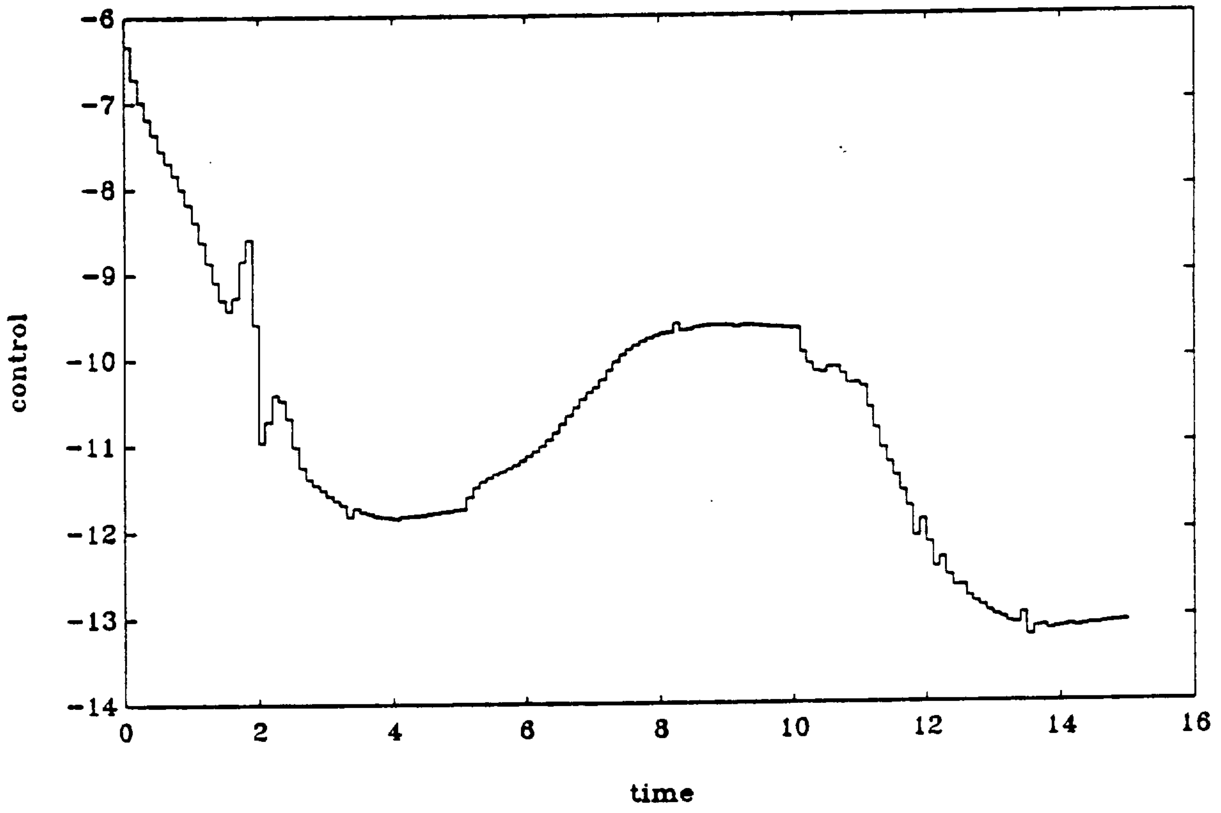


Figure 41. Nonlinear Adaptive MAC

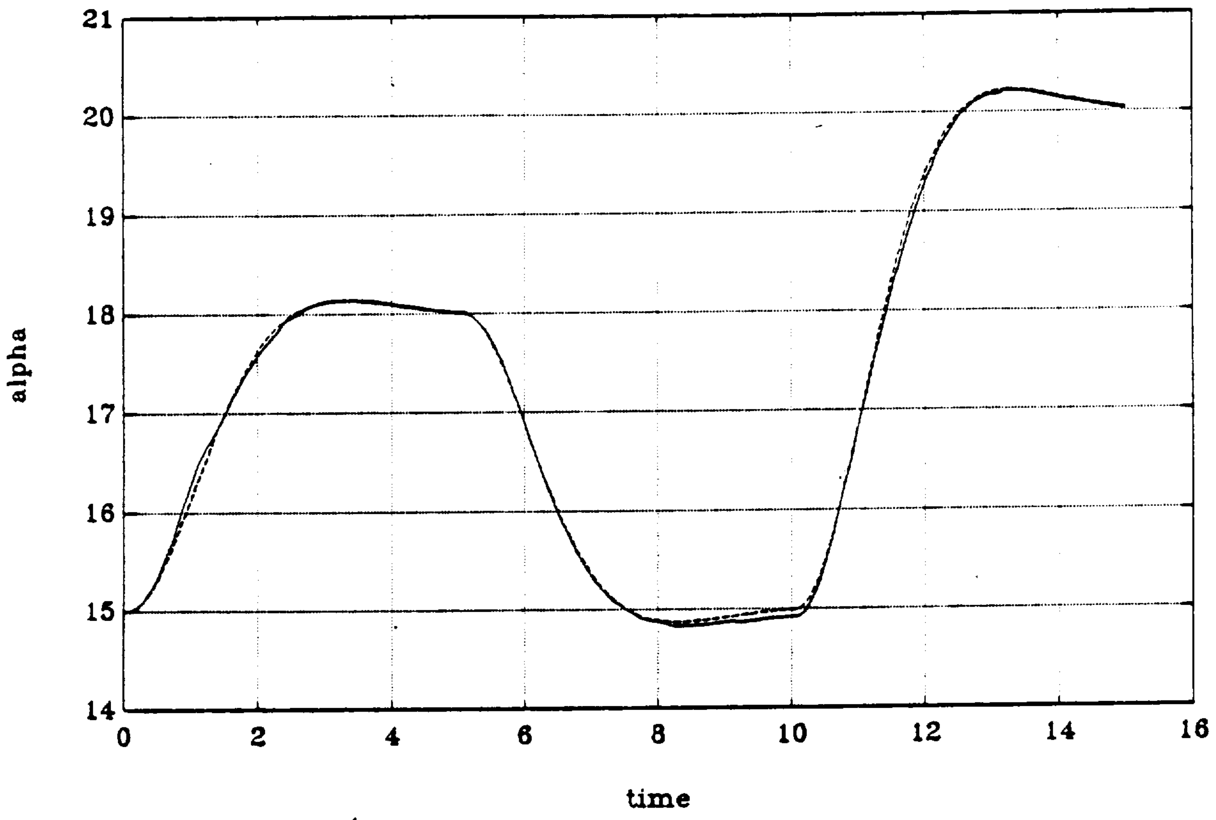


Figure 42. Nonlinear Adaptive MAC (with Reference Trajectory)

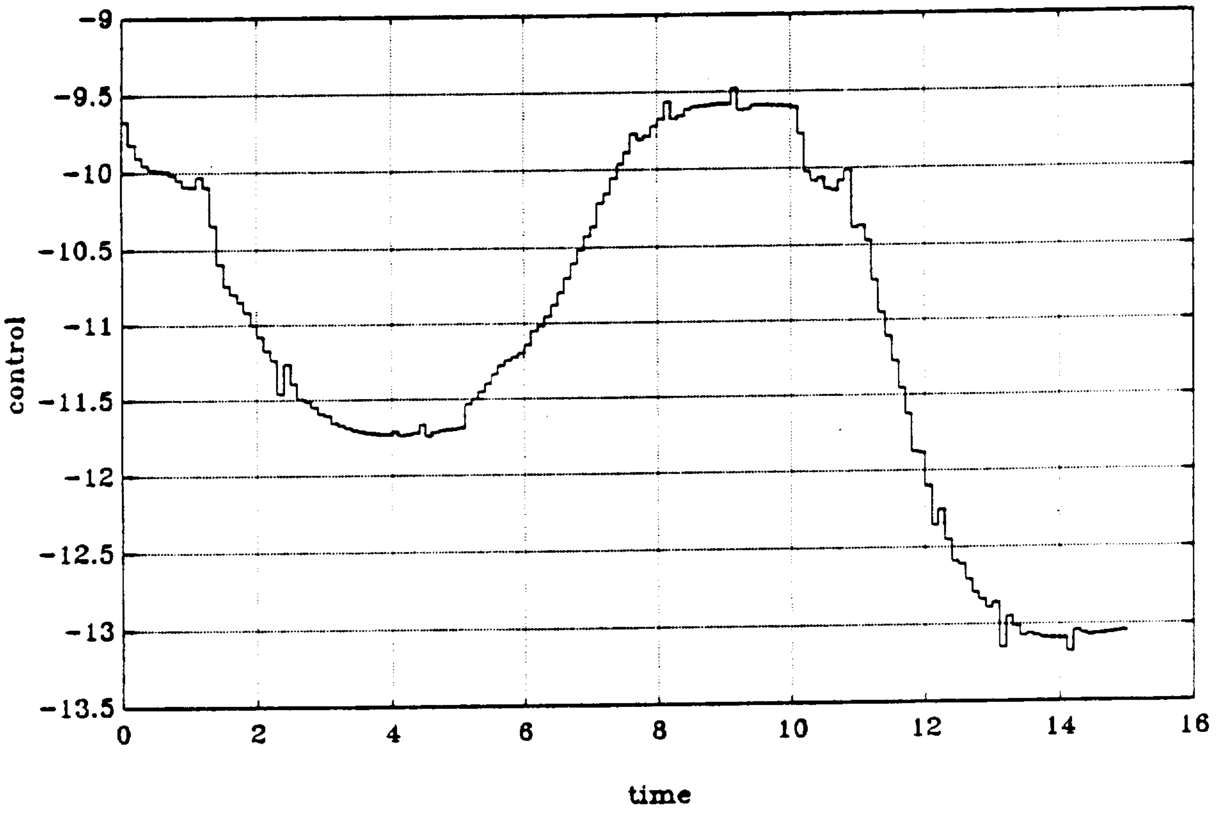


Figure 43. Nonlinear Adaptive MAC

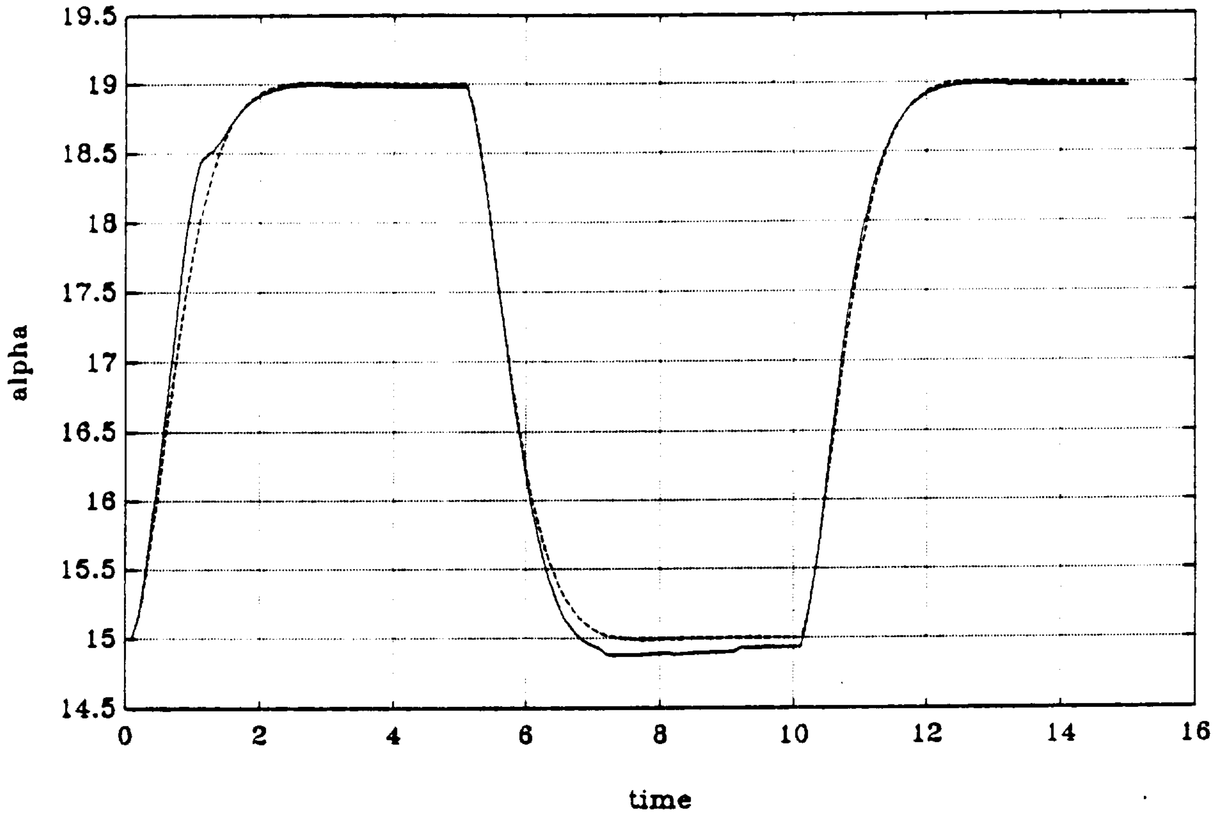


Figure 44. Linear Adaptive MAC (with Reference Trajectory)

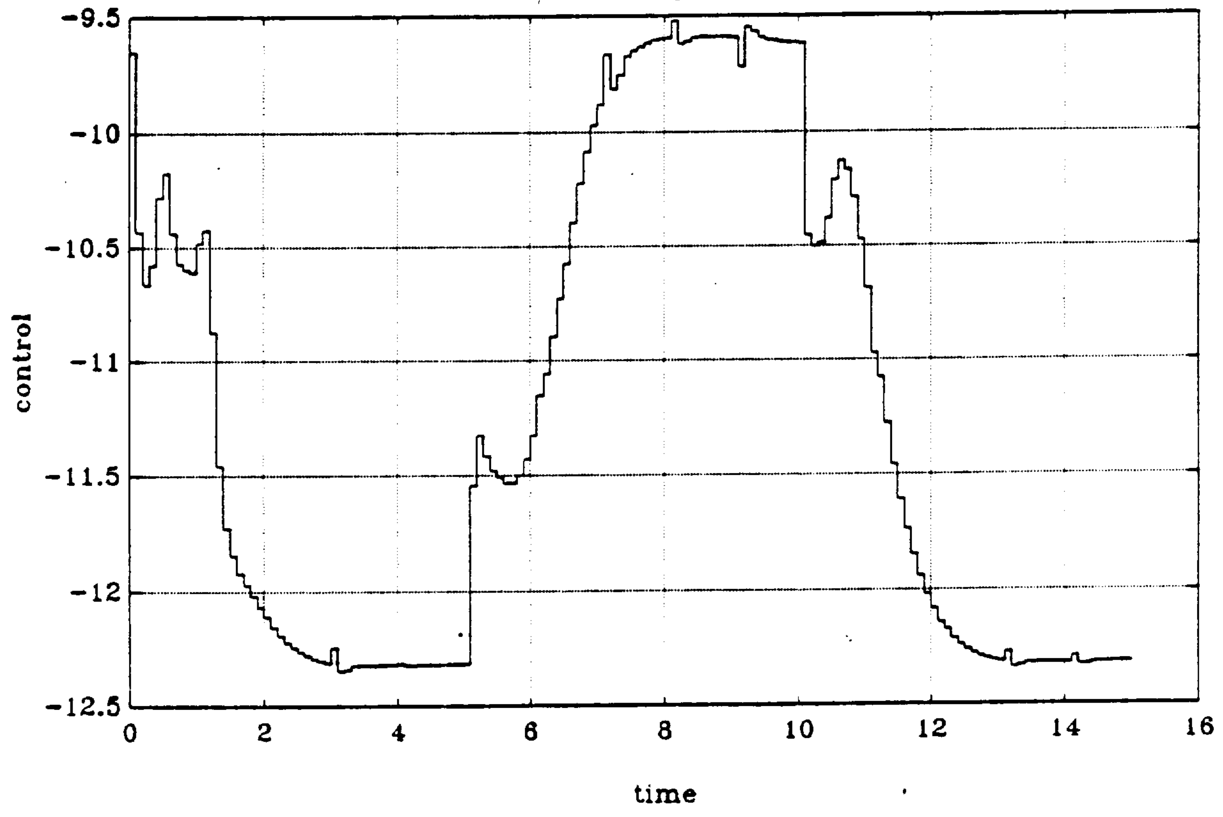


Figure 45. Linear Adaptive MAC

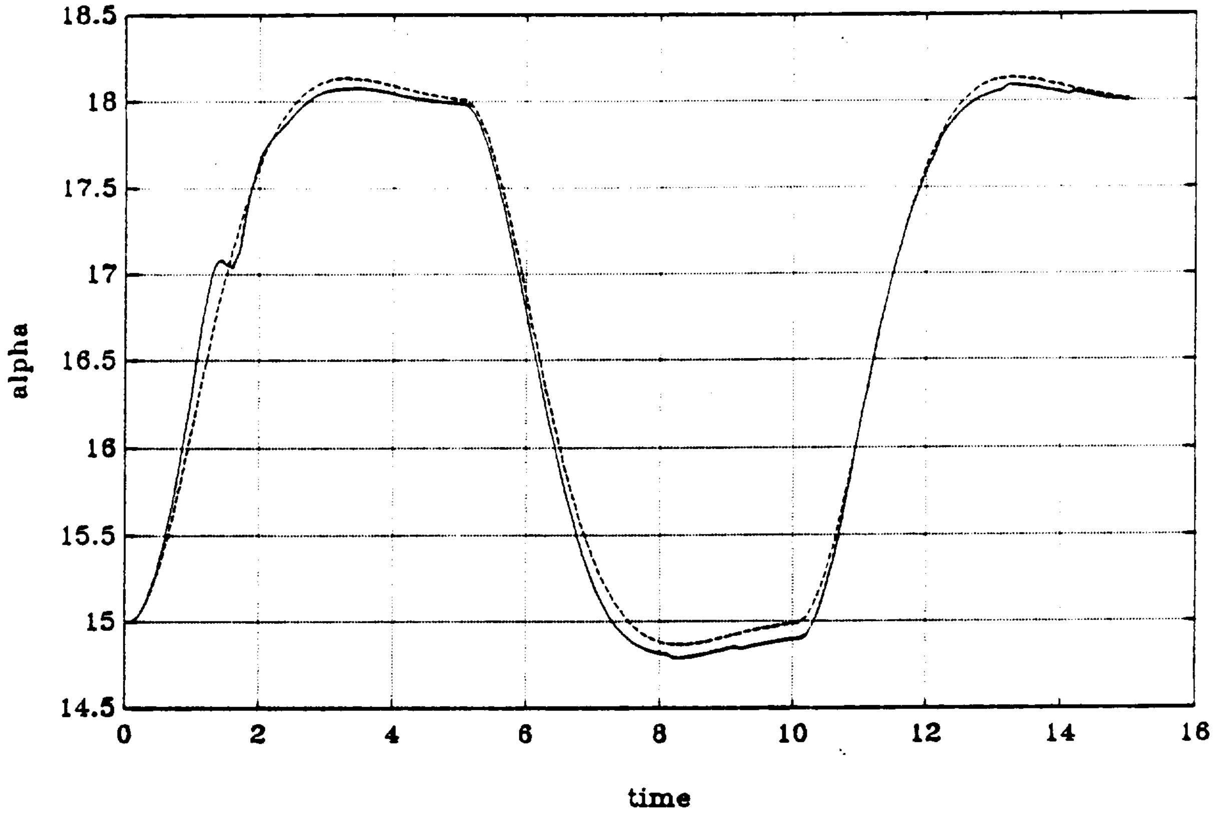


Figure 46. Linear Adaptive MAC (with Reference Trajectory)

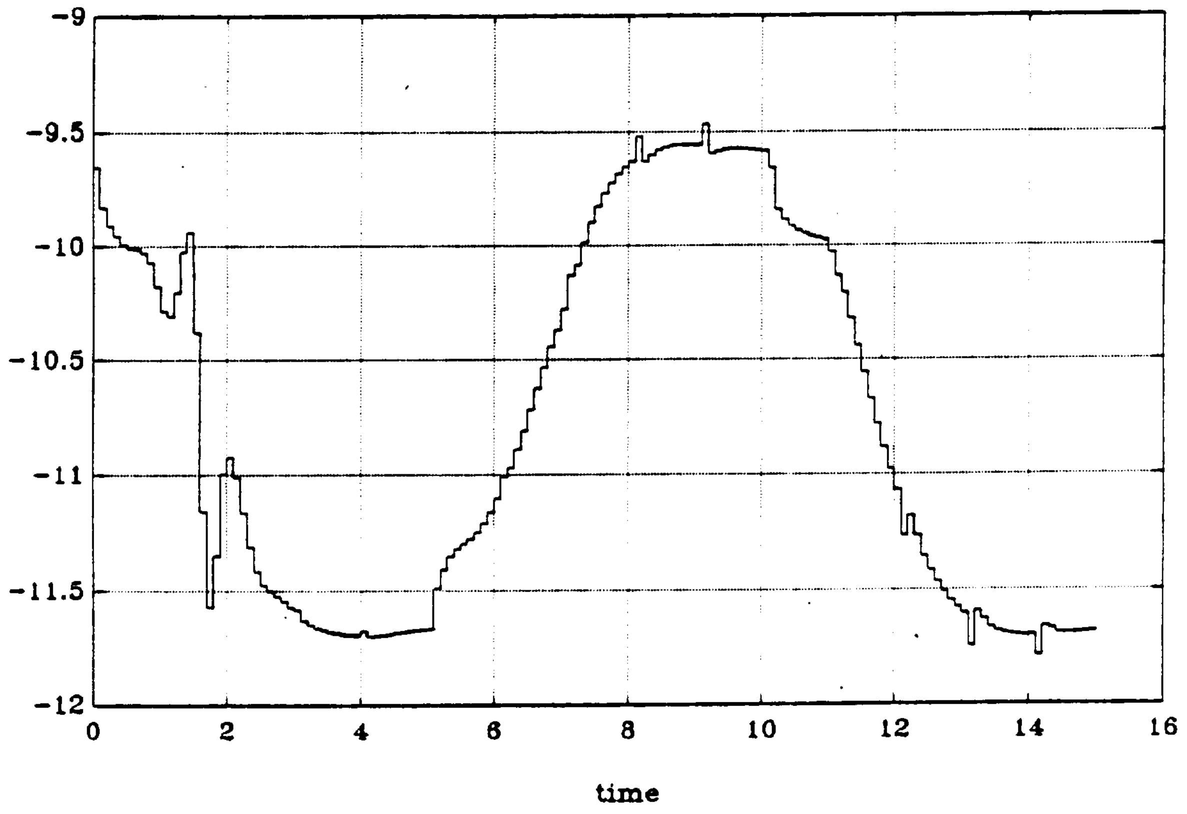


Figure 47. Linear Adaptive MAC

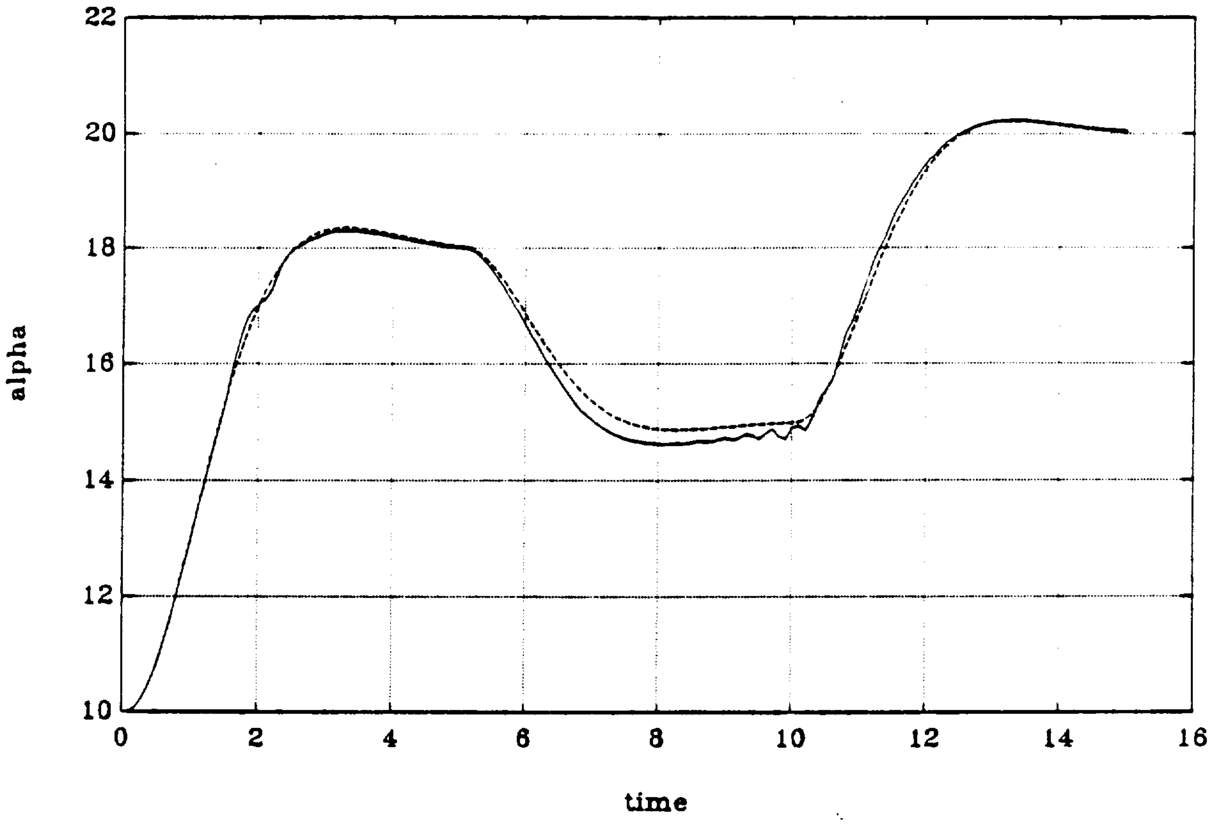


Figure 48. Linear Adaptive MAC (with Reference Trajectory)

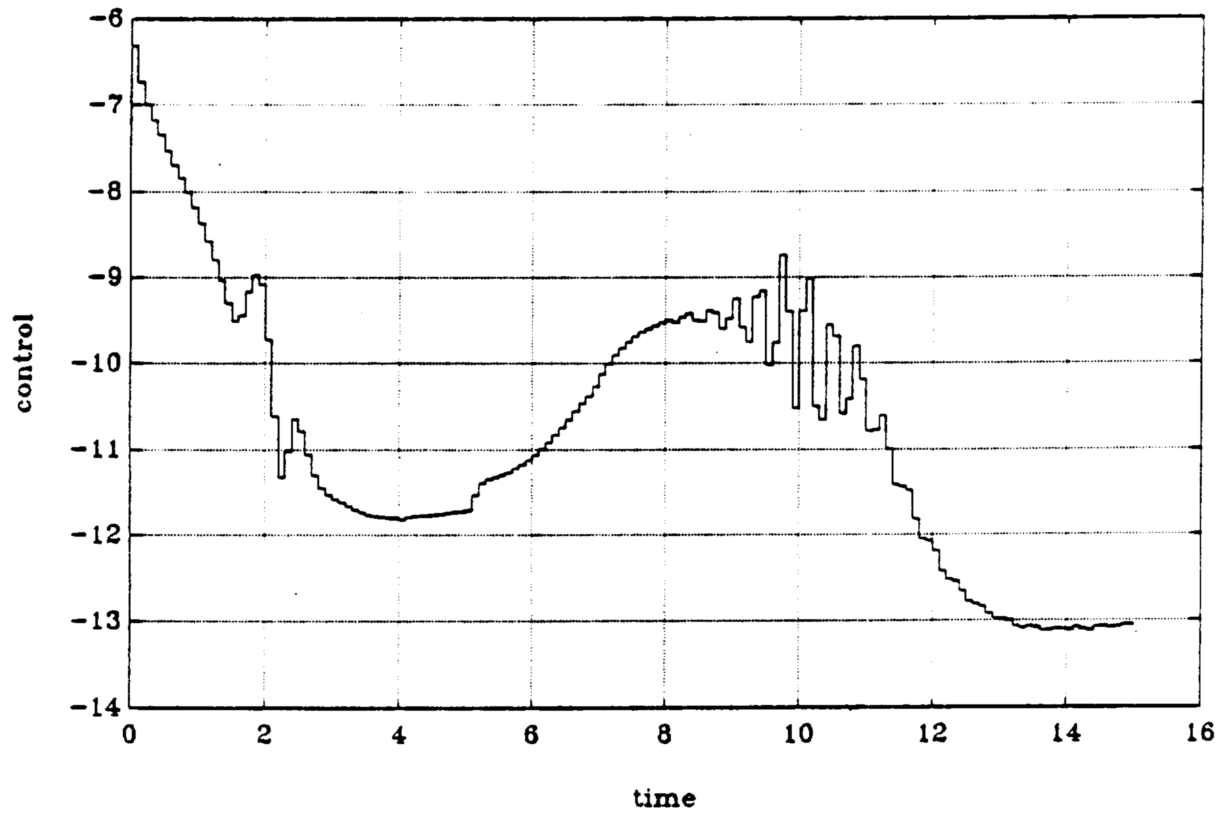


Figure 49. Linear Adaptive MAC

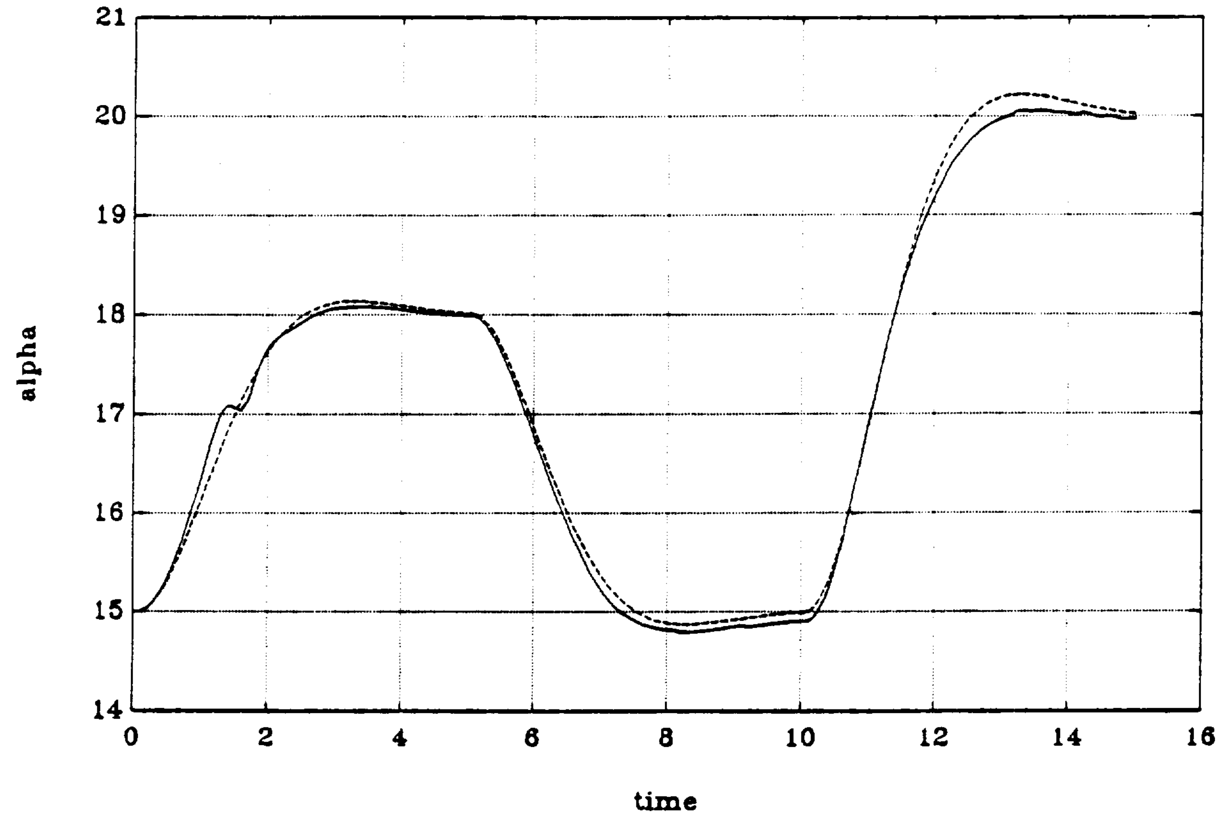


Figure 50. Linear Adaptive MAC

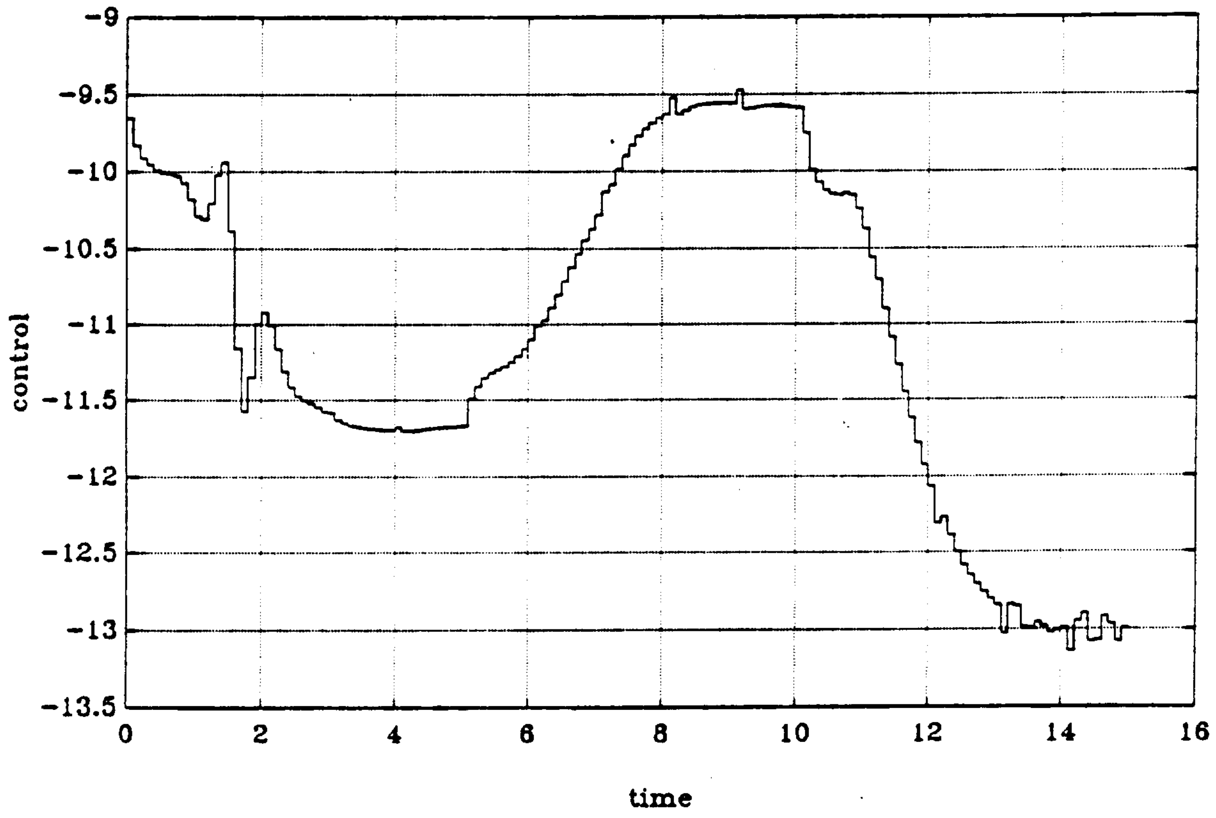


Figure 51. Linear Adaptive MAC (with Reference Trajectory)

obvious difficulties with fitting the parameters of a linear model to the behavior of the plant which is clearly nonlinear in this region (see Figure 47). As a result, the control starts oscillating for a while. Also, on the other plots it is seen that the nonlinear algorithm results in control plots that are more smooth, although they still contain one-pulse spikes. To eliminate these spikes weighting of the increments of control was introduced into the algorithm. The controller is designed to minimize the one step ahead cost function:

$$J = (y_{mod}(k+1) - y_r(k_1))^2 + \rho(u(k) - u(k-1))^2 \tag{3.18}$$

with y_{mod}, y_r as before. Minimization of (3.18) with respect to u(k) yields

$$u(k) = \frac{(\bar{y}_r - a)b + \rho u(k-1)}{b^2 + \rho} \tag{3.19}$$

where

$$a = p_{1\alpha}\alpha + p_{2\alpha}\alpha^2 + p_{3\alpha}\alpha^3 + p_{4\alpha}q + p_{5\alpha}q\alpha + p_{6\alpha}q\alpha^2 + p_{7\alpha}q\alpha^3 + p_{12\alpha}$$

$$b = p_{8\alpha} + p_{9\alpha}\alpha + p_{10\alpha}\alpha^2 + p_{11\alpha}\alpha^3$$

Obviously, for $\rho = 0$ (3.19) reduces to (3.12) while for $\rho = \infty$ we have u(k) = u(k-1) = const. Results of simulations of this algorithm with $\rho = 0.02$ and $\rho = 0.05$ are shown in Figures 52-53 and 54-55, respectively. The trade-off between the accuracy of tracking and control smoothness may be observed. For $\rho = 0.05$ the control contains no one-pulse spikes and, in fact, the accuracy of reference following deteriorates only slightly.

Conclusions

Model algorithmic control based on an approximate discrete state space model works very well for the plant (2.35) with angle of attack as the output. Its adaptive version displays behavior slightly inferior to the nonadaptive case, which may be a result of inclusion of a probably unnecessary corrective term in (3.11). A control increment term in the cost function makes it possible to obtain more smooth control trajectories while retaining satisfactory performance. Previous research failed to find good global input-output nonlinear time-series approximation for the plant (2.35), so a state space model was used. It seems, however, that

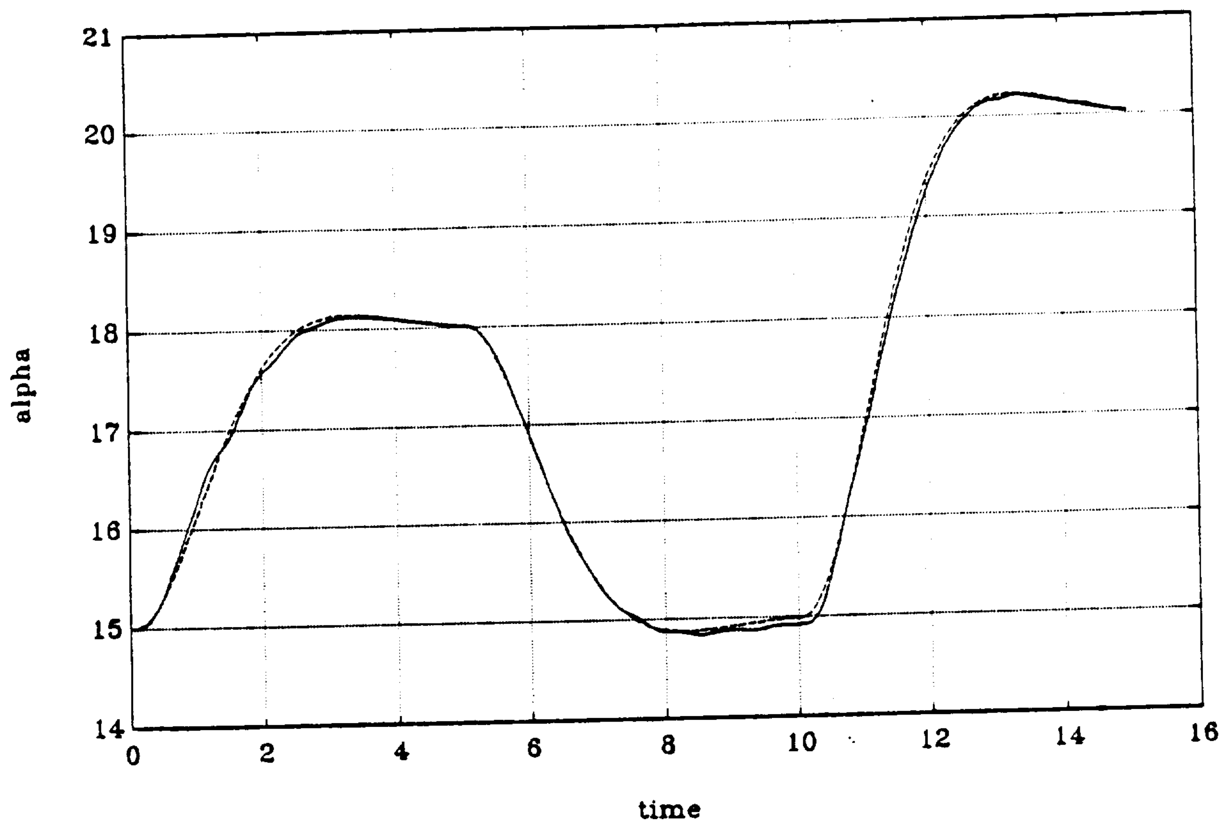


Figure 52. Nonlinear Adaptive MAC with Control Weighting ($\rho = 0.05$)

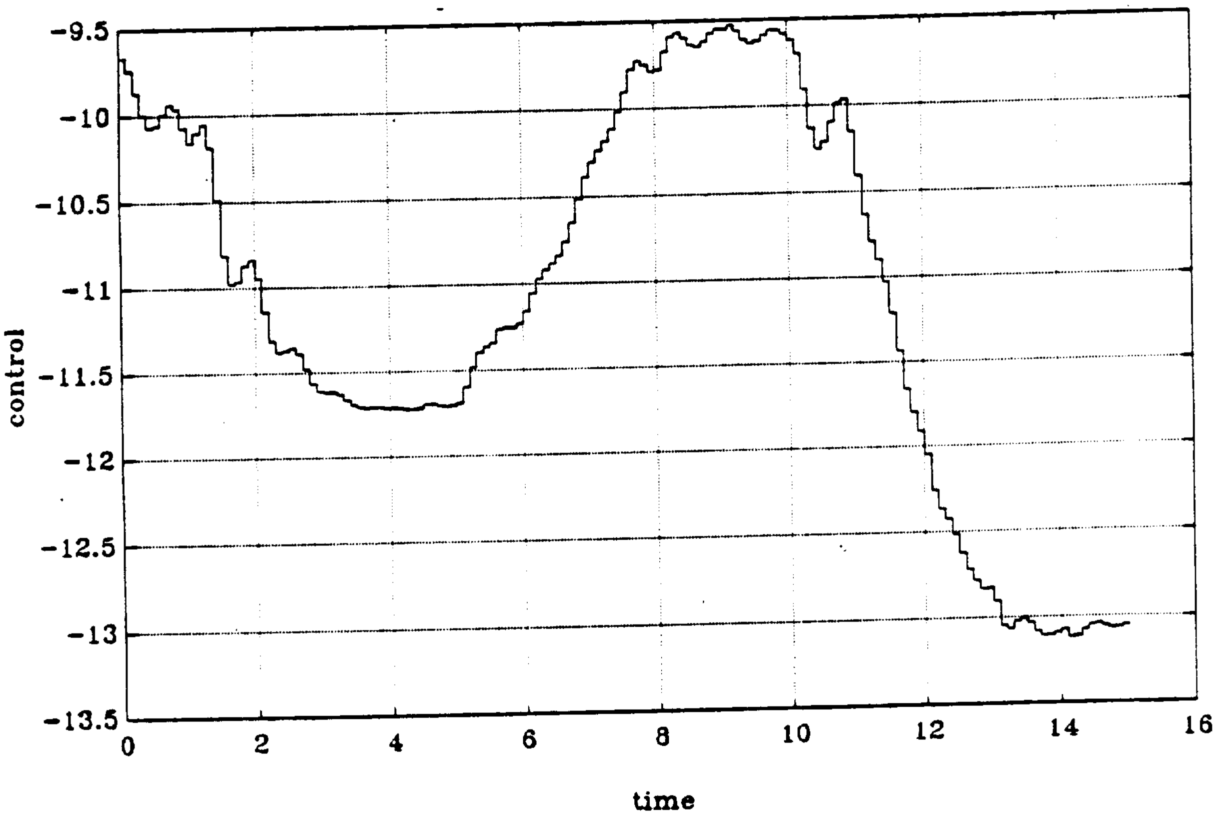


Figure 53. Nonlinear Adaptive MAC with Control Weighting ($\rho = 0.05$)

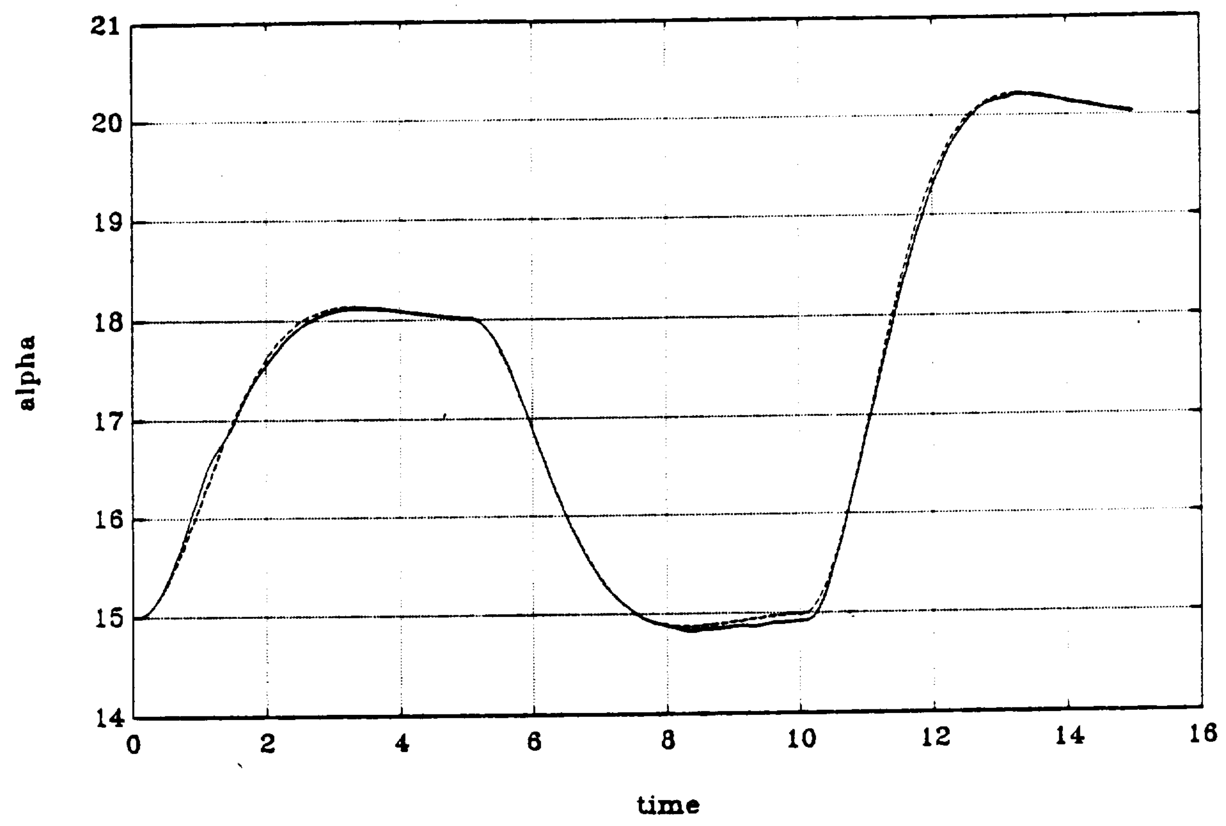


Figure 54. Nonlinear Adaptive MAC with Control Weighting (ρ = 0.02)

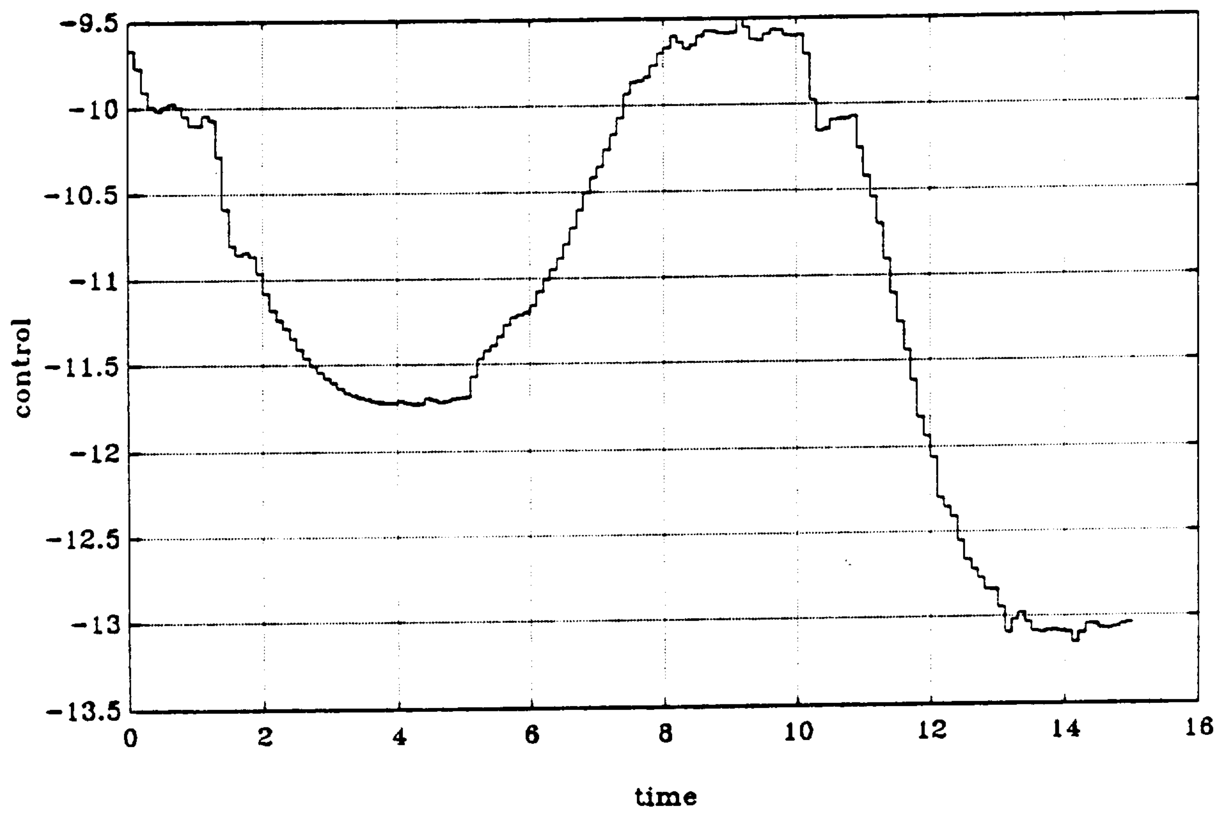


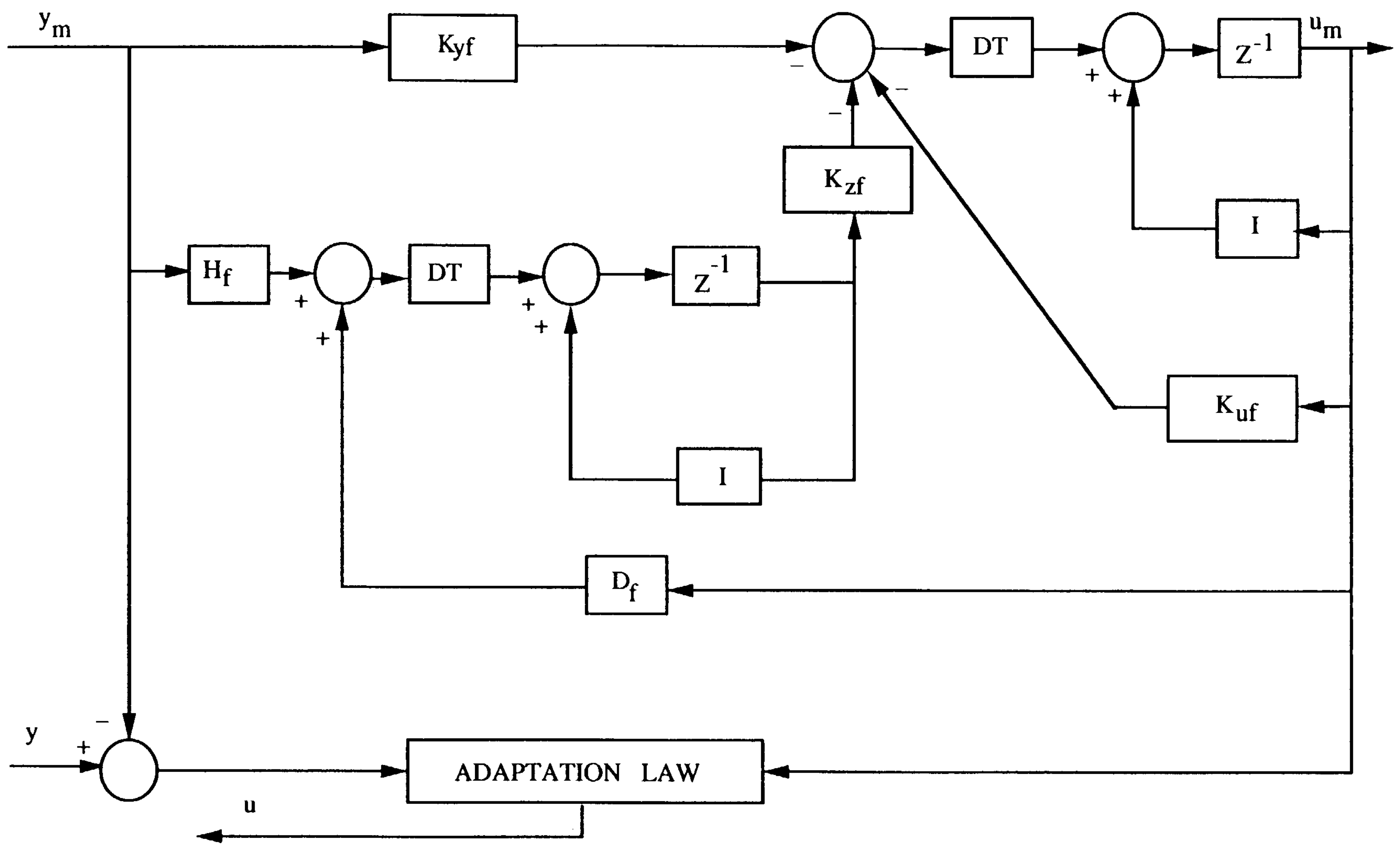
Figure 55. Nonlinear Adaptive MAC with Control Weighting (ρ = 0.02)

locally such approximations should be possible thus allowing for application of adaptive MAC algorithm based only on input-output data without using state measurements. Theoretical research as to robustness of the algorithm with respect to state or measurement noise also will be conducted and relative to more complex airframe simulations.

3.4 PIF Control

The proportional plus integral plus filter (PIF) control, which has been developed by Ostroff at NASA-Langley, has shown remarkable success in simulations as demonstrated in [4] and more recent work at NASA. Flight tests of a more refined PIF controller are planned for an experimental high-alpha aircraft.

Basically, the design incorporates linear optimal (quadratic performance index) control about multiple equilibrium conditions with controller gain scheduling in conjunction with a tracked command model. We are planning to study a simplified version of this controller for possible stability limitations from certain rapid, high-alpha maneuvers. We hope to develop a nonlinear adaptive PIF controller and compare its performance with the present NASA PIF controller. An example of this configuration is shown in Figure 56.



K_{yf}, K_{uf}, D_f, H_f, K_{zf} possibly functions of y_m

Figure 56. Nonlinear PIF Control Structure

4. CONCLUSIONS AND PLANS

Research in the first phase of a three-year program suggests that nonlinear adaptive control (such as those based on nonlinear time series or Volterra kernels) can provide improved stability over that of simple linear adaptive control. To make this conclusion more conclusive, however, future research will involve more complex airframe models and simulations. Also, more complex linear and nonlinear adaptive controllers, including PIF designs, will be studied for stability and performance limitations. Comparisons of the studied nonlinear control algorithms and the NASA-Langley PIF design (Ostroff) are planned.

REFERENCES

[1] R.R. Mohler, *Nonlinear Systems: V. 2 Applications to Bilinear Control*, Prentice-Hall, Englewood Cliffs, NJ, 1991.

[2] L. Chen, X. Yang, R.R. Mohler, "Stability of Bilinear Systems," *IEEE Trans. Autom. Control* AC36, 1991 (to appear).

[3] R.R. Mohler, *Nonlinear Systems: V. 1 Dynamics and Control*, Prentice-Hall, Englewood Cliffs, NJ, 1991.

[4] A.J. Ostroff, "Application of Variable-Gain Output Feedback for High-Alpha Control," AIAA Guidance Nav. & Control Conf. (Paper No. 89-3576), Boston, 1989.

[5] H. Stalford, W.T. Baumann, F.E. Garrett, T.L. Herdman, "Accurate Modeling of Nonlinear Systems Using Volterra Series Submodels," *Proceedings*, 1987 American Control Conference, Minneapolis, 1987, Vol. 2, pp. 886-891.

[6] H. Wakamatsu, "Model Reference Nonlinear Adaptive Control System Using Nonlinear Autoregressive Moving Average Model Derived from Volterra Series and Its Application to Control of Respiration," Automatic Control, World Congress 1987, *Selected Papers from the IFAC 10th Triennial World Congress*, Munich, 1987, Vol. 10, pp. 191-196.

[7] R. Bars, R. Haber, "Long Range Predictive Control of Nonlinear Systems Given by Volterra Series," Identification and System Parameter Estimation, *Selected Papers from the Eighth IFAC/IFORS Symposium*, Beijing, 1988, Vol. 2, pp. 931-935.

[8] K. Harris, "Properties of Nonlinear Model Algorithmic Control," *Proceedings*, 24th Conference on Decision and Control, Ft. Lauderdale, 1985, Vol. 1, pp. 663-665.

[9] S.A. Al-Baiyat, "Nonlinear Feedback Synthesis: A Volterra Approach," Ph.D. dissertation, Dept. of Electrical Engineering, University of Notre Dame, 1986

[10] A.D. Modyaev, A.D. Averina, "Analysis and Synthesis of Discrete Control Systems based on Multidimensional z Transforms," in B. Naumov, *Philosophy of Nonlinear Control Systems*, Mir Publishers/CRC Press, 199.

[11] K. Harris, "Properties of Nonlinear Model Algorithmic Control," *Proceedings*, 24th Conference on Decision and Control, Ft. Lauderdale, 1985, Vol. 1, pp. 663-665

[12] G.C. Goodwin, K.S. Sin, *Adaptive filtering, Prediction and Control*, Prentice Hall, 1984

APPENDIX A

MATLAB and C Program Comparisons

NONLINEAR STABILITY AND CONTROL STUDY OF HIGHLY MANEUVERABLE HIGH-PERFORMANCE AIRCRAFT.

NASA Project Progress Report.

by
ZION HALEVY **
E.C.E, Oregon State University, Corvallis Or.

JULY 16, 1991

1. SUMMARY:

The object of this report is to summarize part of the programming work done on nonlinear adaptive control model of aircraft operating in highly nonlinear regimes with large values of angle of attack.

Two models (Cho[+] and Stalford[*] models) were simulated on PC-MATLAB and converted to "C". Comparison of the simulation time indicated that performance using the "C" algorithm was increased by two orders of magnitude. Thus enabling detailed parametric analysis in short time span. The fundamental structure of C programs can be also utilized extended to future simulator models.

** On sabbatical leave from ADA, Israel.

+ Refers to the full nonlinear longitudinal airframe model (Section 2.1 of Annual Report)

* Refers to approximate nonlinear model of Reference 3.

2. CONTENTS.

3. MATLAB SIMULATIONS.

3.1 Cho model [2]

An original MATLAB program using an Euler integration scheme (BLIN1.M ,COF.M -written by Cho 1991 based on Cao et al 1990) was analyzed and restructured in order to enable general purpose use and faster execution time.

The changes performed on the original programs are as follows:

a) Removal of the close-loop BARMA controller (Cho 1990) from the main program model BLIN1.M and writing of a general purpose structure controller sub program.

b) Modification and improvement of the structure of the program, in particular modification of the sub program COF.M (calculate seven aerodynamic coefficients) which was called at each iteration.

The final programs named ZBLIN1.M and ZCOF2.M are in the attached disk (see Appendix 3).

3.2 Stalford model. [3]

A model taken from Stalford 1989 was written in PC-MATLAB with the same general purpose structure noted above . This work was done jointly with J. Dory.

The final programs named ZRUN5.M ,ZIN5.M and ZMAIN5.M using ODE45.M (Runge-Kutta 4th and 5th order integration function for numerical solution of ordinary differential equations) are on the attached disk.

The results from the MATLAB simulation (Appendix 2) indicate that the numerical model is identical to that presented by Stalford 1989 .

4. <u>"C" PROGRAMS.</u>

4.1 <u>Cho model</u>

The final MATLAB programs (ZBLIN1.M ,ZCOF2.M) was converted to the "C" language (ZBL13.C ,ZCOF3.C and ZBL13.EXE -executable program).

The final programs are in the attached disk (see Appendix 3).

4.2 <u>Stalford model.</u>

The final MATLAB programs (ZRUN5.M ,ZIN5.M,ZMAIN5.M, ODE45.M) was converted to "C" language (ZRUN5.C, ZIN5.C, ZMAIN5.C, ODE45.C and ZRUN5.EXE -executable program) .

* xxx.M / xxx.C extensions denote MATLAB and "C" programs files respectively.

5. Comparison between MATLAB and "C" programs

Matlab is an interactive program that aid with fast performence scientific and engineering numerical calculation.

Matlab allows you to solve numerical and simulation problems in a fraction of the time it would take to write in other languages like "C" or Fortran.

Furthermore, problem solutions are expressed in almost exactly the same way as they are written mathematically.

It is a complete integrated system including graphics, programmable macros,an interpreter and analytical commands.

"C" and Turbo C++ has fast and efficient compiler that enables transfer application programs to other systems.

The MATLAB is an interpreter and hasn't a compiler like "C",so the "C" language is more efficient .

The MATLAB simulations included in this report were obtained using PC-MATLAB version 3.2 .

5.1 Cho model.

Comparison of performance are summarized in Table 1. Appendix 1 consists of examples plots from both MATLAB and "C" in the same conditions as follow:

Euler integration step= 0.05 [sec] (the desired accuracy).
tfinal= 10 [sec] (final value of t).
t0= 0 [sec] (initial value of t).

TABLE 1: Cho model.

PROGRAM	CONTROLLER INPUT dh	FLIGHT TIME [SEC]	SIMULATION TIME [SEC]
1. BLIN1.M ,COF.M (original prog.)	VARIABLE (Cho)	10.0	1068
2. BLIN1.M,ZCOF2.M (Improve COF.M)	VARIABLE (Cho)	10.0	580
3.ZBLIN1.M,COF.M (Remove controller from main prog, original COF.M)	CONSTANT -1.0 [deg]	10.0	555
4.ZBLIN1.M,ZCOF2.M (Remove controller from main prog, improve COF.M)	CONSTANT -1.0 [deg]	10.0	364
5. C PROGRAMS ZBL13.C ,ZCOF3.C (Remove controller from main prog. improve COF.M)	CONSTANT -1.0 [deg]	10.0	6

5.2 Stalford model.

Comparison of performance are summarized in Table 2. Appendics 2 consists of examples plots from both MATLAB and "C" in the same conditions as follow:

ODE45 step integration= 0.1 [sec].
ODE45 tolerance= 1.e-4 [sec] (the desired accuracy).
tfinal=T_FINAL=13.6565 [sec] (final value of t).
t0= 0.45*T_FINAL [sec] (initial value of t).

TABLE 2 :Stalford model.

PROGRAM	CONTROLLER INPUT dh	FLIGHT TIME [SEC]	SIMULATION TIME [SEC]
1.MATLAB programs: ZRUN5.M ,ZMAIN5.M ZIN5.M .	VARIABLE STEP (ZIN5.M)	7.5	1866
2. C PROGRAMS: ZRUN5.C ,ZMAIN5.C ZIN5.C , (and ZRUN5.EXE)	VARIABLE STEP (ZIN5.M)	7.5	6

6. CONCLUSIONS AND RECOMMENDATIONS FOR FUTURE WORK.

6.1 Conclusions

Camprison of MATLAB and "C" simulation reveal that performance was increased by two orders of magnitude by the "C" simulation .

MATLAB is a "package" for development stage but for detailed parametric analysis the "C" performance better.

6.2 recommendations.

a) Using of PC-MATLAB version 3.5 (for 386 computer) for first step developing of the simulation.

We used in PC-MATLAB version 3.2 for our simulation but the new version 3.5 (for 386 computer) is faster at least four times (it hasn't limits imposed by 16 bit nature of 80286/8086) .
We used also PC-386/20 Epson computer with 80287 numeric coprocessor chip.

b) Using of MATLAB MEX-files enables to combine "C" and MATLAB programs .

MEX-file produce from compiled "C" linked into .EXE files and renames to .MEX extension.
So it is possible to call your own "C" programs from MATLAB as if they were built-in MATLAB function.
Speed improvement of up to a factor of 25 are possible in this way .

c) Using of "C" (turbo C++ Ver 1.0) simulation for parametric analysis.

d) Using at Cho model in ODE45.M Runge-Kutta integration function instead of Euler integration scheme to get better accuracy of results.

7. ACKNOWLEDGEMENT.

I would like to thank Prof. R.R. Mohler,Principal Investigator and R.R Zakrzewski for the assistance during this work.

8. REFERENCES.

1. J. Cao H. stalford "analytical aerodynamic model of a high alpha research vehicle wind-tunnel model" NASA NAG1-959 1990

2. Cho model- MATLAB programs: BLIN1.M ,COF.M 1991 (written by Cho based on [1]) .

3. H. Stalford E. Hoffman "Maximum principle solutions for time-optimal half-loop maneuvers of high alpha fighter aircraft" ACC ,1989 (p. 2453-2458).

4. C. Moler "PC-MATLAB for MS-DOS personal computers" User's Guide ver. 3.2 1987

5. B.W Kernighan D.M Ritchie "The C programming language"

6. Turbo C++ manual
   User's Guide of Borland

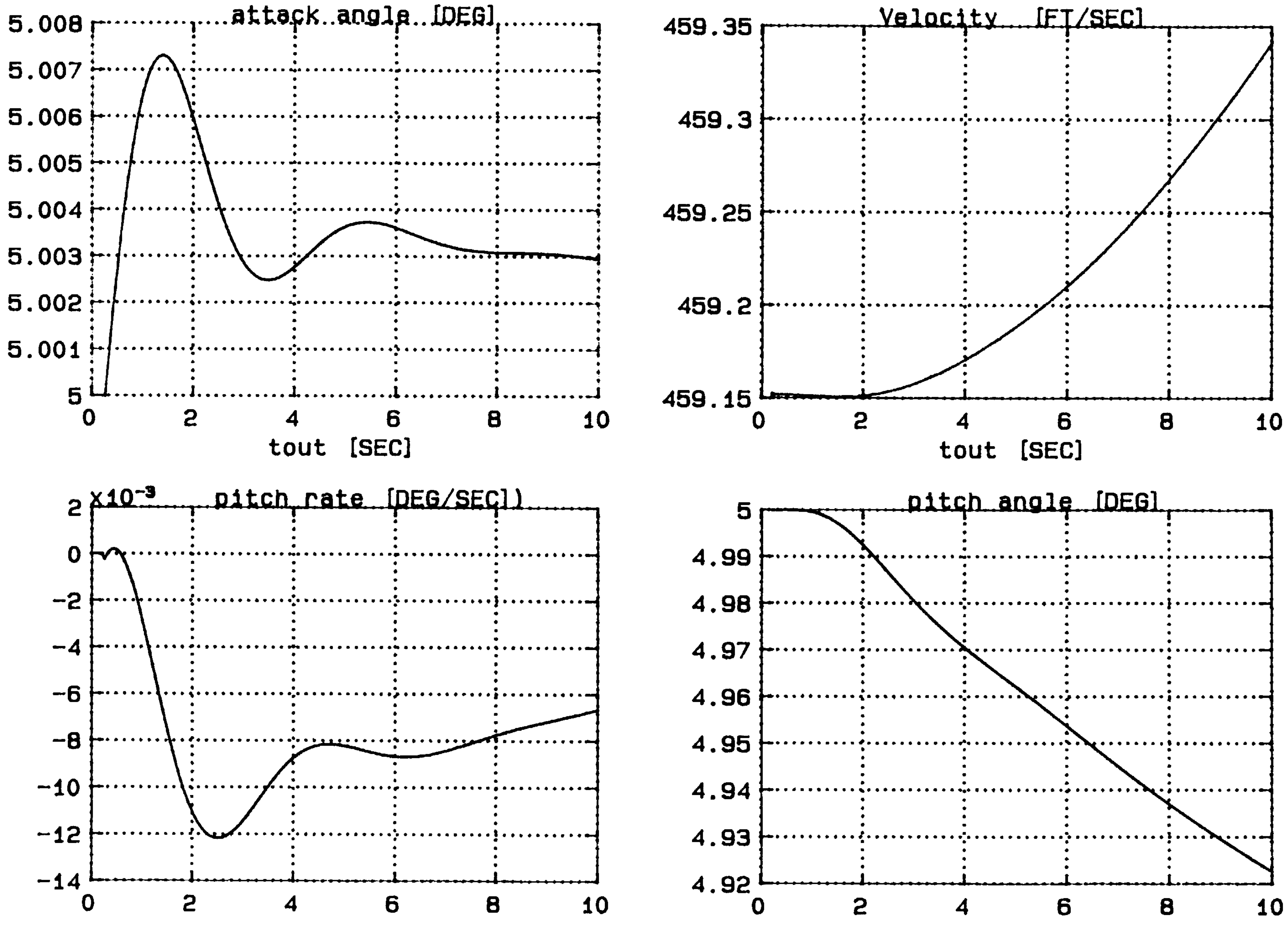
9. APPENDICES

1. Cho model : examples plot (Fig 1 - Fig 6).

2. Stalford model : examples plot (Fig 7 - Fig 10).

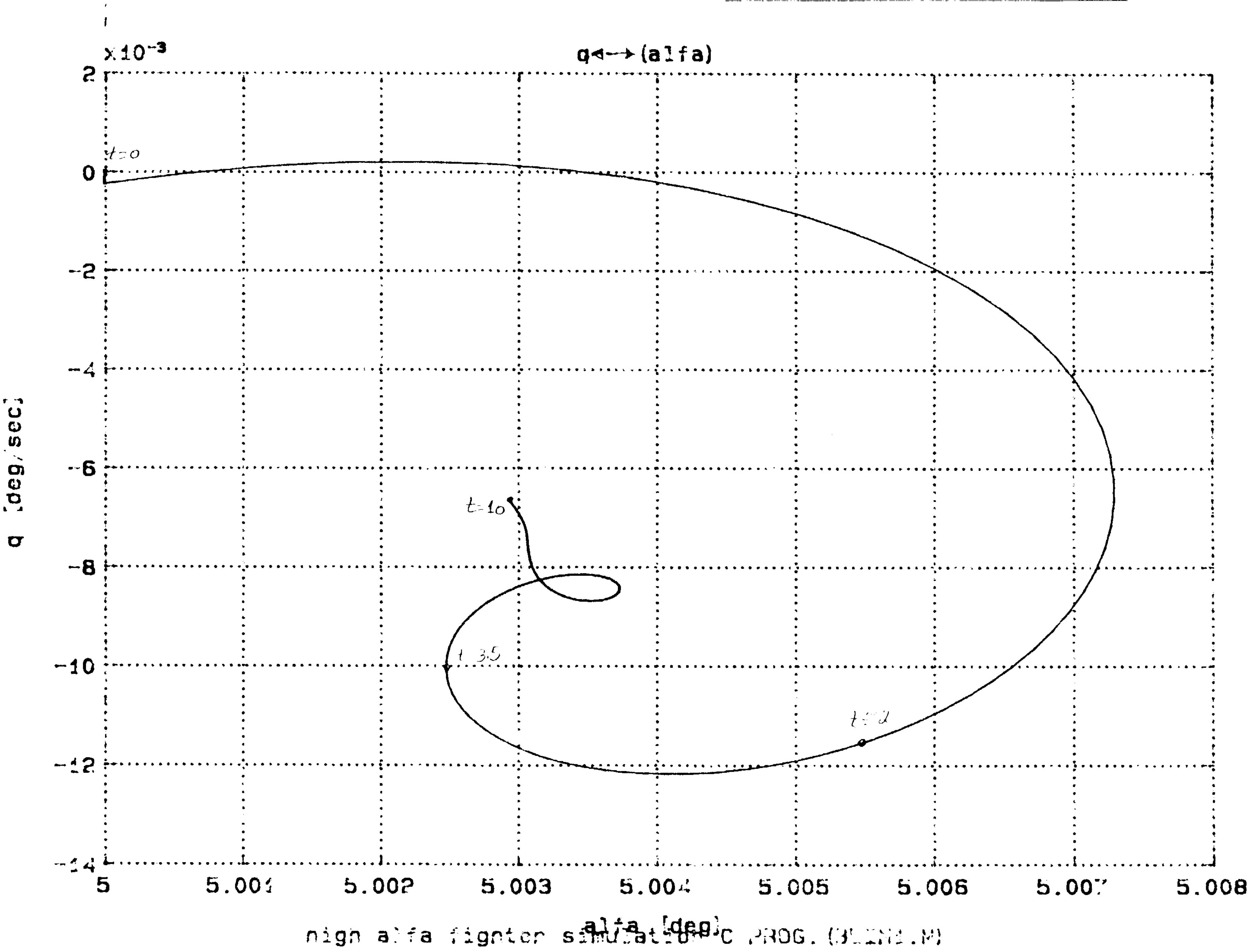
3. Programs description of the attached disk.

FIG 1 [illegible]

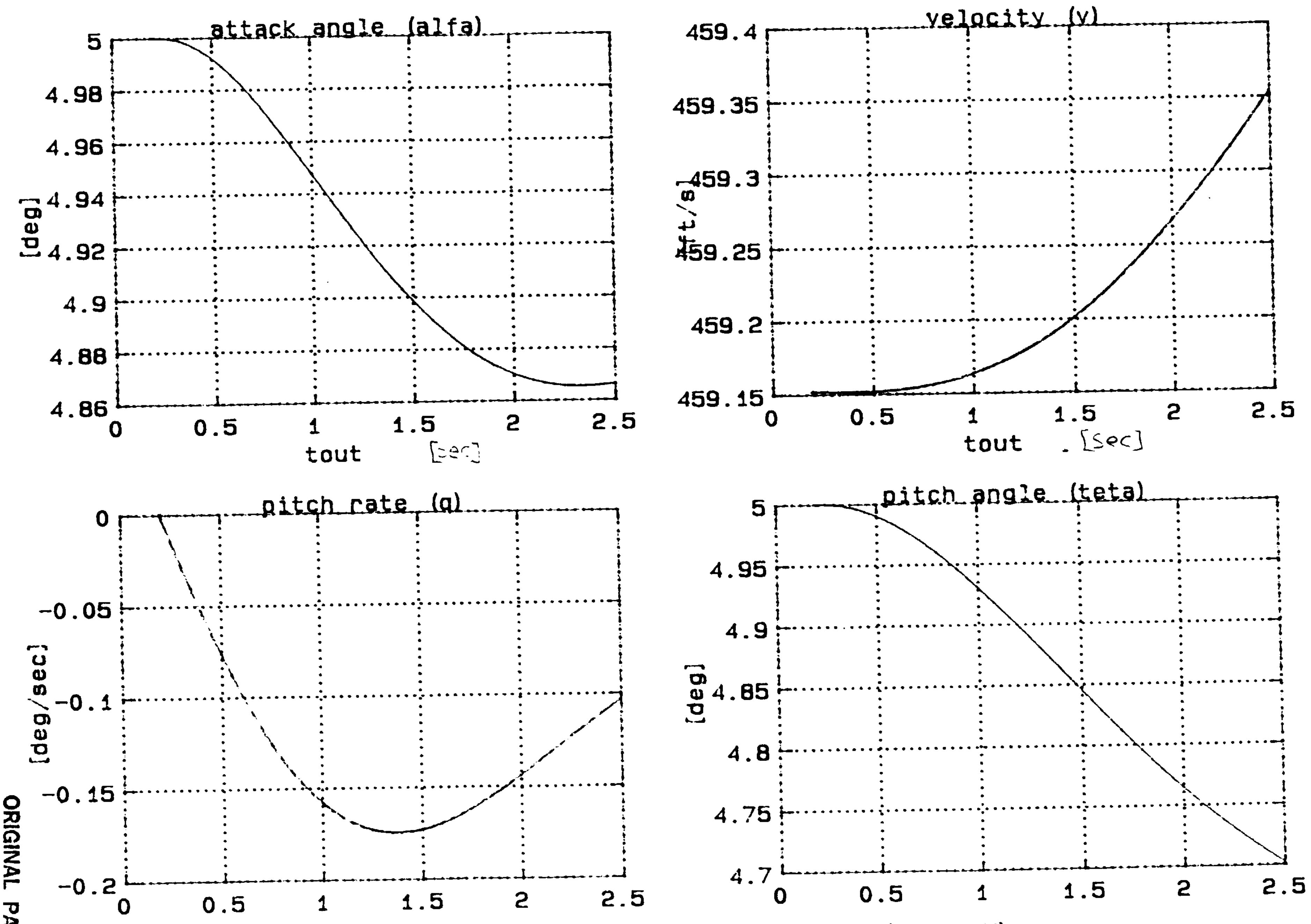


high alfa fighter simulation C PROG.(BLIN1.M)

FIG 2 : $dh = -1.052^{\circ}$ (const)

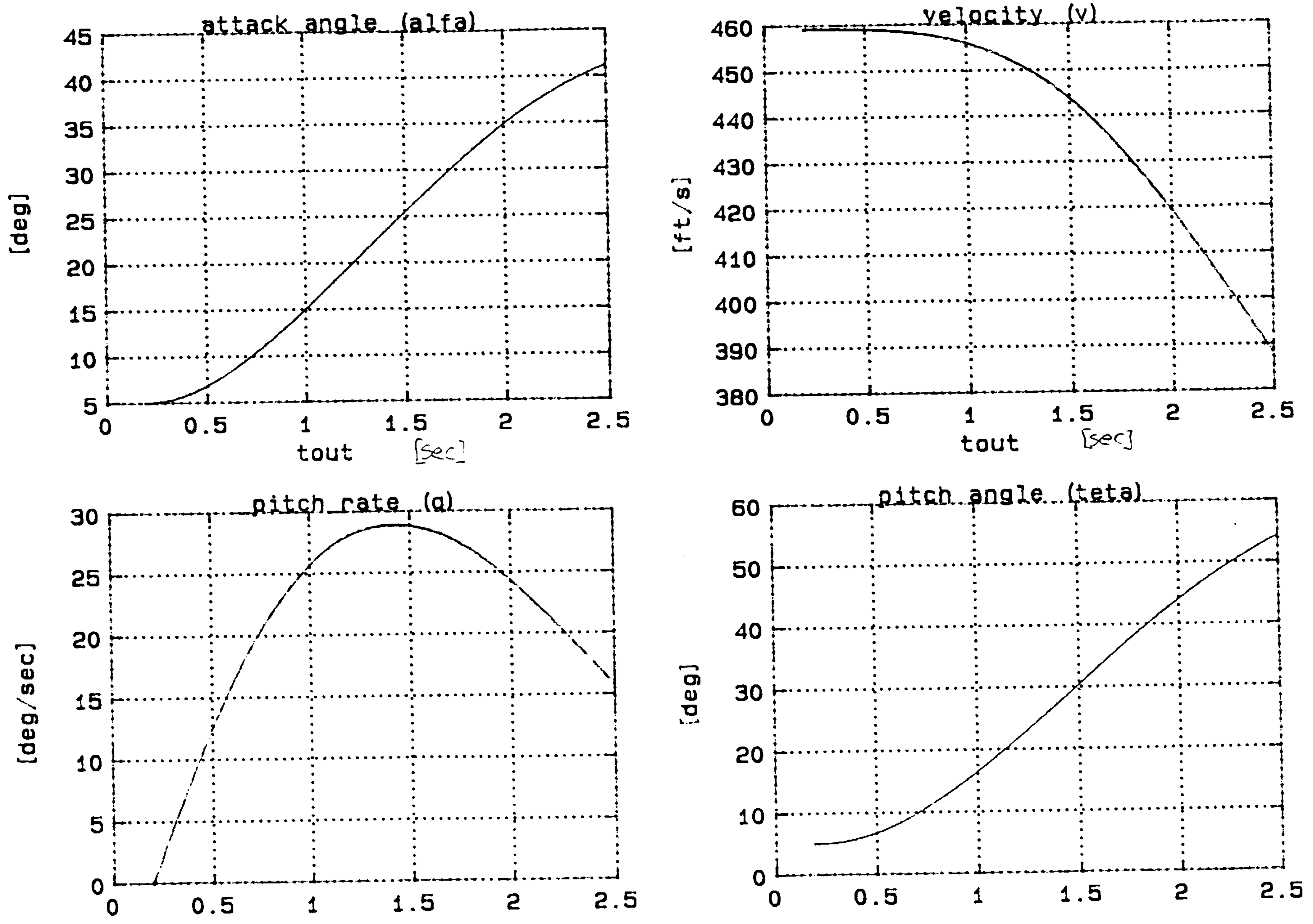


ORIGINAL PAGE IS OF POOR QUALITY

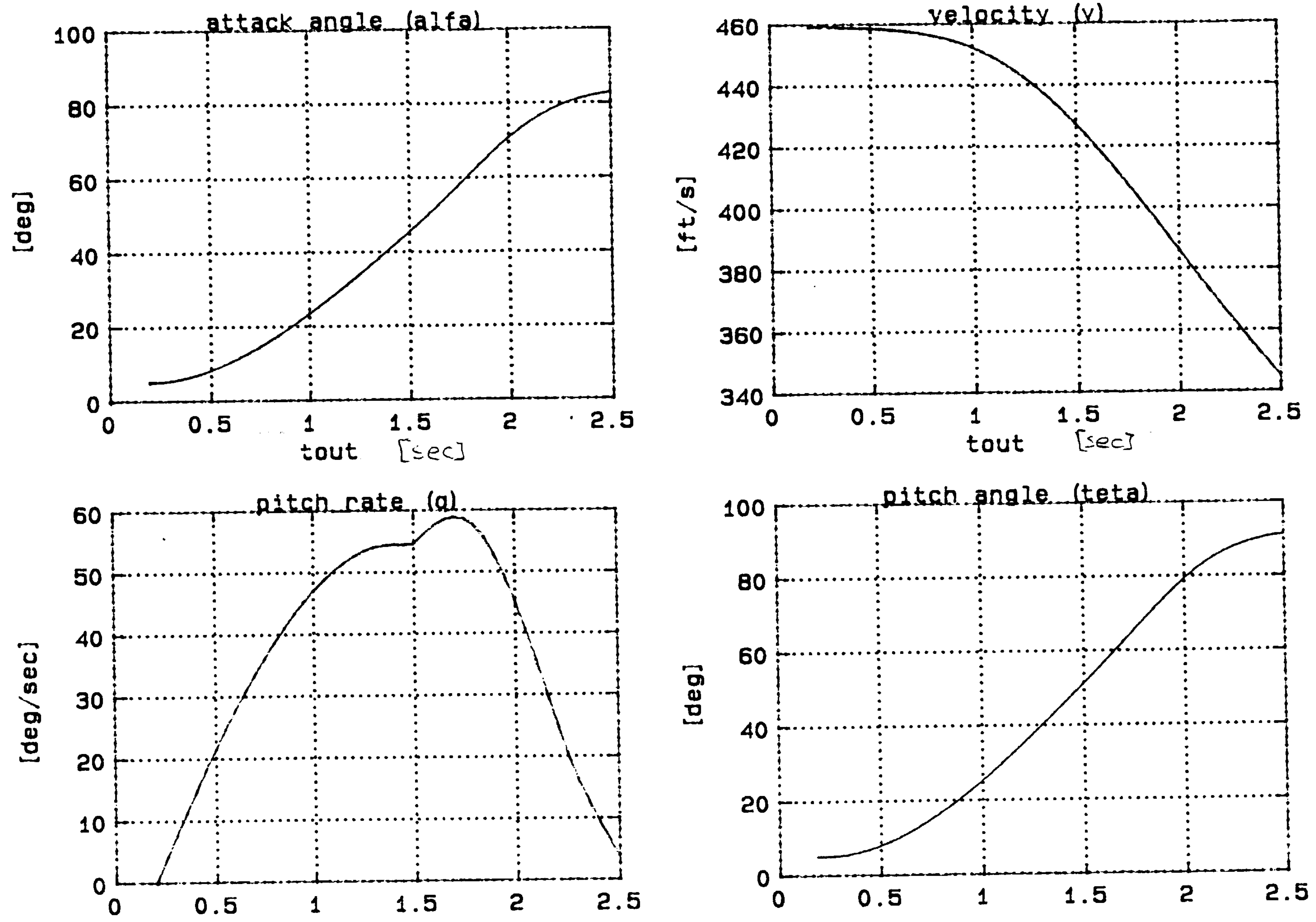


high alfa fighter simulation C PROG. (BLIN1.M)

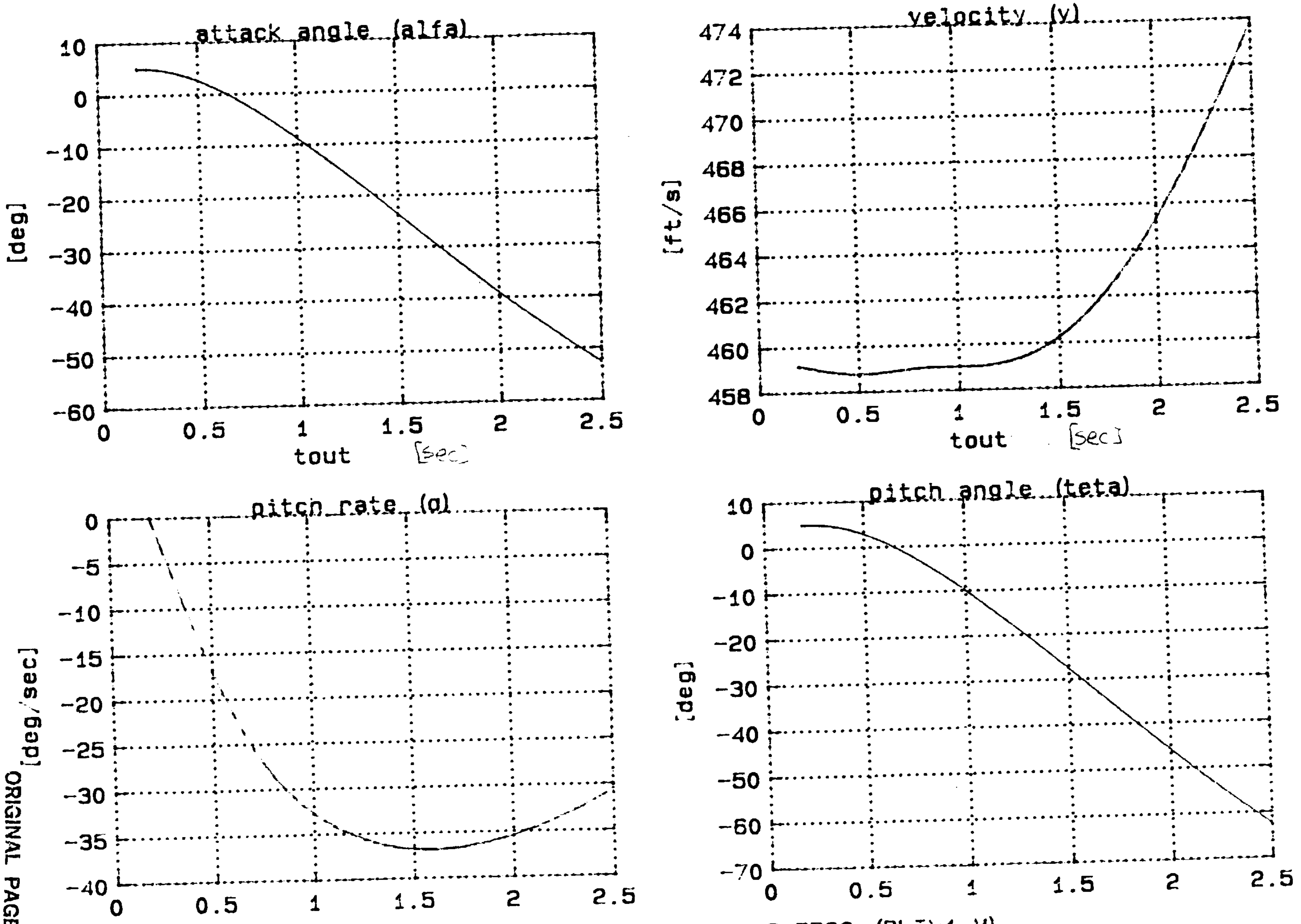
FIG- 7 [illegible]



high alfa fighter simulation C PROG. (BLIN1.M)

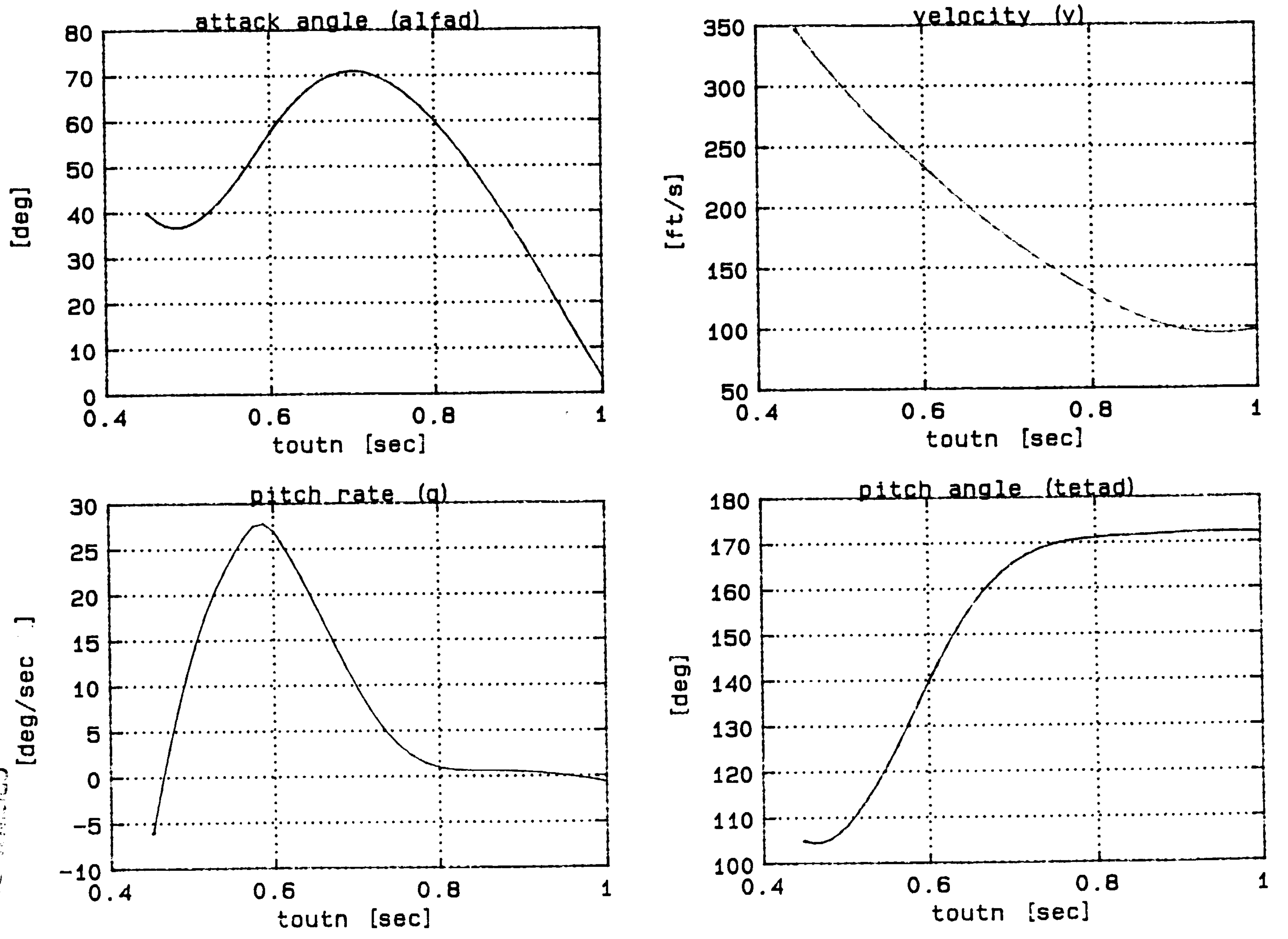


high alfa fighter simulation C PROG. (BLIN1.M)



high alfa fighter simulation C PROG. (BLIM1.M)

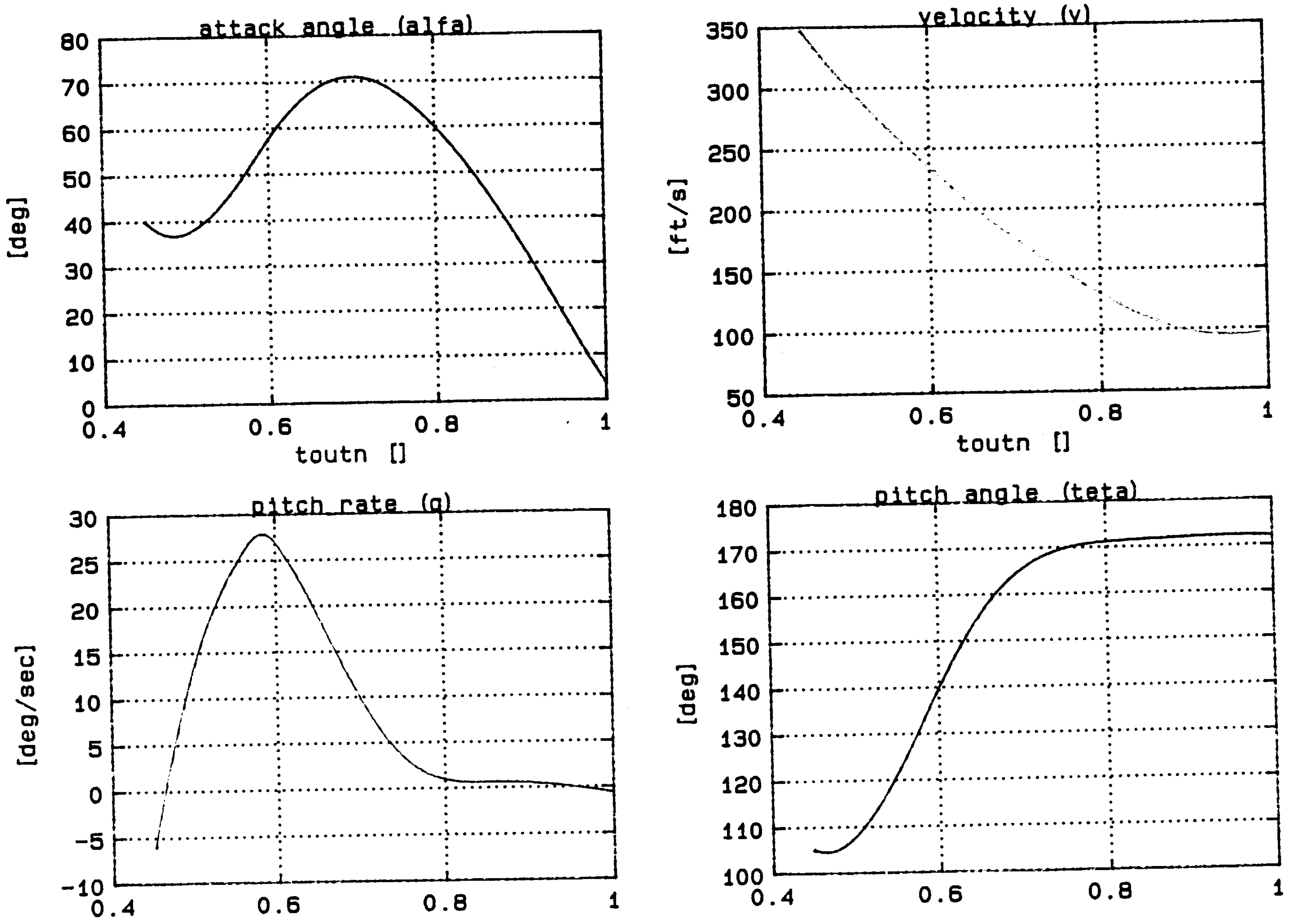
FIG-7 : MATLAB SIMULATION PLOT (T_FINAL: 1365 Sec)



($\Delta T_3 = 1866$ sec)

ORIGINAL PAGE IS
OF POOR QUALITY

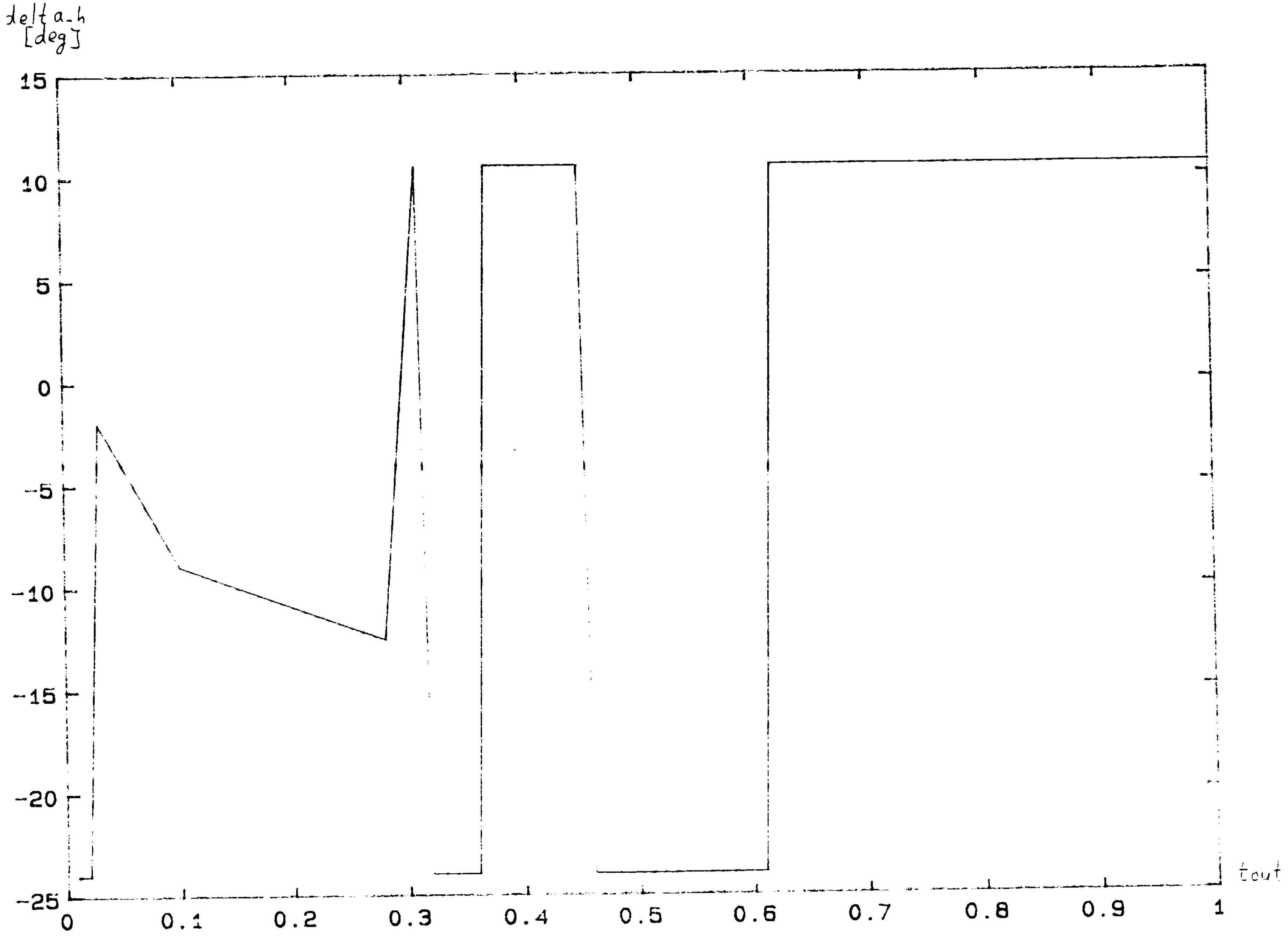
FIG- 8 : "C" SIMULATION PLOT(T_FINAL= 15.65)



high alfa fighter simulation C PROG.

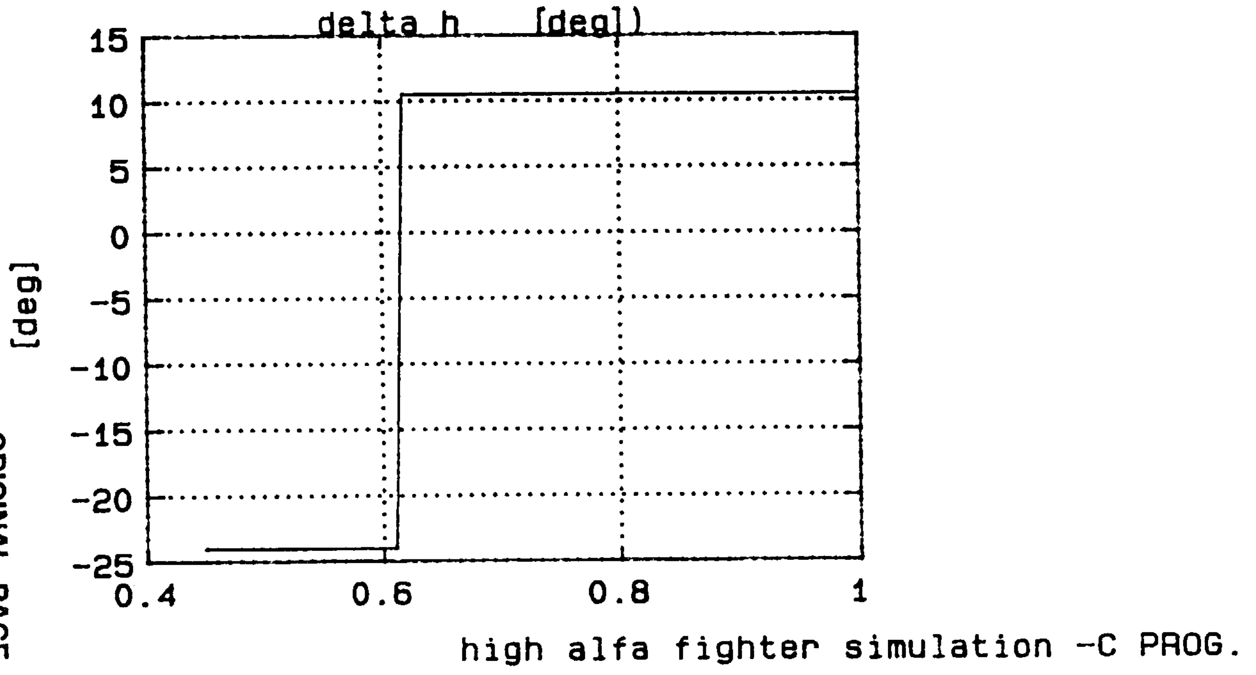
$\Delta T_j = 6$ sec

FIG- [illegible] : INPUT CONTROL [illegible]



ORIGINAL PAGE IS OF POOR QUALITY

FIG- 10 : INPUT CONTROL [illegible]



APPENDIX 3

The programs description in the attached disk :

1. Cho model [2].

a) MATLAB programs.

BLIN1.M-	Original main program model (including the Cho close loop controller) [2].
COF.M -	Original function called by main program (BLIN1.M) to calculate seven aerodynamic coefficients [2]
ZBLIN1.M-	The final main program after the changes performed on BLIN1.M (see 3.1).
ZCOF2.M-	The final function after the changes performed on COF.M (see 3.1).

b) "C" programs .

ZBL13.C-	The final MATLAB program ZBLIN1.M converted to "C"
ZCOF3.C-	The final MATLAB program ZCOF2.M converted to "C".
ZBL13.PRJ-	"C" project (link) file for ZBL13.C .
ZCOF3.H-	"C" header file for ZCOF3.C (prototype).
ZBL13.EXE-	The final executable program for Cho model.
RES.RES-	The output results file (alpa,V,q,teta,dh,t).
ZLC1.M -	The program used for graphic program (at MATLAB). to load RES.RES.
ZPCC5.M -	The graphic program called by ZLC1.M.
ZPC51.M -	The graphic program called by ZLC1.M.

2. Stalford model [3].

a) MATLAB programs.

ZRUN5.M-	The macro running program that give initial values to the model function (ZMAIN5.M) and also graphic program (zp5.m).

ZMAIN5.M-The function describes nonlinear aircraft equations in the form to be used by ODE45.M (integrating routine) called by ZRUN5.M.

ZIN5.M -The function describes input controller called by ZRUN5.M , the output of the function is the value of the control (the elevetor angle).

ODE45.M -Integration of a system of ordinary differential equations called by ZRUN5.M .The function is using 4th and 5th order- Runge-Kutta formulae .

ZP5.M -The graphic program called by ZRUN5.M.

b) "C" programs .

ZRUN5.C- The final MATLAB program ZRUN5.M converted to "C"

ZMAIN5.C-The final MATLAB program ZMAIN5.M converted to "C".

ZIN5.C- The final MATLAB program ZIN5.M converted to "C"

ODE45.C- The final MATLAB program ODE45.M converted to "C"

ZRUN5.PRJ-"C" project (link) file for ZRUN5.C .

ZMAIN5.H- "C" header file for ZMAIN5.C (prototype).

ZMAIN51.H- "C" header file for ZMAIN5.C (decleration).

ZIN5.H- "C" header file for ZIN5.C (prototype).

ODE45.H- "C" header file for ODE45.C (prototype).

UTIL.C-Utility program for ODE45.C .

UTIL.H- "C" header file for UTIL.C (prototype).

ZRUN5.EXE-The final executable program for Cho model.

RES.RES- The output results file(alfa, V, q, teta, delta_h, tn).

ZLC.M -The program used for graphic program (at MATLAB) to load RES.RES.

ZP5.M -The graphic program called by ZLC.M.

ZP51.M -The graphic program called by ZLC.M.

APPENDIX B

Project Publications

PROJECT PUBLICATIONS

(Supported Wholly or in Part by NASA Grant)

1. R.R. Zakrzewski, R.R. Mohler, "On Nonlinear Model Algorithm Controller Design," *Proceedings*, IFIP Conf. Sys. Modeling and Optimiz., Zurich, 1991 (to appear).

2. R.R. Mohler, V. Rajkumar, R.R. Zakrzewski, "Nonlinear Time-Series Based Adaptive Control Applications," *Proceedings*, IEEE Conf. Decision & Control, Brighton, 1991 (to appear).

3. R.R. Mohler, *Nonlinear Systems: Vol. 2 Applications to Bilinear Control*, Prentice Hall, Englewood Cliffs, NJ, 1991.

4. R.R. Mohler, V. Rajkumar, R.R. Zakrzewski, "On Discrete Nonlinear Self-Tuning Control," *Proceedings*, Korean Control Conf., Seoul, 1991 (to appear).