Digital Keywords

Attitude Hold Mode
Autopilot
Axial Velocity Maneuver
Computer Simulation
Dot Product Jet Selection Method
EASY5
Euler's Theorem
Inter-Continental Missile (ICBM)
Lagless Filter
Large Attitude Maneuver
Least Squares Estimator
Monte Carlo Analysis
Phase Space Logic
Post Boost Vehicle (PBV)
Reaction Control Jets
Robert G. Borst - Boeing Defense & Space, Flight Controls Engineer
Terminal Condition Maneuver
Quaternion
Single-Axis-Rotation (SAR)
Z-Transform

Certificate of Appreciation

to

Robert G. Borst

Engineering Technology Employee of the Month - November, 1984

FOR PERFORMANCE EXCELLENCE:

For developing a superior simulation program capable of analyzing the performance of an ICBM's post boost vehicle's (PBV) attitude and velocity control system. This work has made Boeing's PBV velocity and attitude control analysis competitive with the industry.

Presented This 15th *Day of* FEBRUARY, 1985

[illegible]
W. L. Wilson

[illegible]
L. J. Mason

BOEING AEROSPACE COMPANY
A Division of The Boeing Company

To: T.H. Chase 8C-39

November 26, 1984
2-3633-SFFK-CS 191/84

cc:	T.R. Anderson	8K-80
	R.A. Brueske	17-04
	G.J. Burmeister	8C-39
	R.L. Dyrdahl	17-04
	J.W. Evans	8C-39
	R.D. Jones	8R-16
	J.F. Kenney	17-04
	M.T. Oshima	8F-17
	J.F. Toomey	13-10

Subject: Generalized PBV Simulation for IR & D task: Ballistic Missile Control EWA 90862

Reference: (1) Bergmann, E.V., "A New Spacecraft Autopilot," Charles Stark Draper Laboratory Report T-628, May 1976.

(2) Kellogg, M.L., "Precise Nulling of Attitude and Motion Errors of a Spacecraft Using a Phase Space Autopilot." Charles Stark Draper Laboratory Report T-661, Sept. 1978.

(3) Silver, S.R., "A Unified Spacecraft Autopilot for Controlling Velocity and Attitude in Several Types of Maneuvers," Charles Stark Draper Laboratory Report T-823, May 1983.

Summary

An EASY5 simulation has been developed to rapidly assess controllability of a generic postboost vehicle (PBV). A large attitude control law, axial velocity control law, terminal condition control law, and an attitude hold control law have been integrated into one control algorithm. This commonality minimized computer program complexity and memory requirements. The phase space technique was used to coordinate control for all six degrees of freedom of the vehicle and permit convergence on an end state of constant commanded attitude and constant commanded linear velocity. All IR & D objectives have been successfully completed and verified using several PBV configurations and representative system tolerances.

Problem

Computer tools for assessing PBV reaction control system performance are all very detailed and oriented toward specific program designs, such as the inertial upper stage (IUS). These tools, therefore, do not lend themselves to rapid assessment of system design requirements. A preliminary design tool is needed that will allow easy implementation of configuration changes.

Objective

The specific objective of this IR & D task is to develop a general six degree of freedom preliminary design computer simulation for PBV reaction control design analysis.

Approach

A general simulation using the EASY5 computer code will be developed to permit rapid accommodation of changes in thruster locations, thrust angle with respect to body axes, and thruster characteristics such as minimum impulse bit and thrust level. A single-axis-rotation (SAR) control law utilizing phase space logic will be implemented for assessing timelines, fuel consumption, and duty cycles.To the extent possible, the control law will,be generalized to accomodate various system configurations.

Progress

All planned activity described under Objective and Approach have been successfully completed. The remainder of this memorandum documents the algorithms developed and presents simulation results of a typical eight thruster configuration. Figure 1 depicts the control system which was implemented in the EASY5 simulation.

Measurement and Estimation

The measurements used by the autopilot shown in Figure 1 are assumed to come from an inertial measurement unit (IMU). The IMU measures the velocity contribution produced by the reaction control jet forces (total velocity - gravity component). The IMU measures the linear velocity at its location on the vehicle and measures the vehicle's attitude relative to the IMU inertial reference frame. These IMU measured quantitites are sampled every computer cycle for use in the flight computer error calculation routine. This routine builds an attitude increment vector by comparing the present IMU measured attitude with the attitude measured at the previous computer cycle. This routine also calculates the attitude error by making use of Euler's Theorem and will be discussed in the next section. Since angular rate is not measured directly by the IMU, an estimation method must be used. Two estimation routines have been developed for this control system. A "lagless filter" estimation routine is used when the jets are firing and a "least - squares" estimation routine is used during the coast period prior to the final nulling burn.

The derivation of the "lagless filter" as presented in Reference (3) is as follows:

$$\frac{W_i + W_{i-1}}{2} = \frac{\Delta\theta}{\Delta T} \qquad (1)$$

Where:

ΔT	= control cycle interval
$\Delta\theta$	= change in attitude during the control cycle
W_{i-1}	= the present angular velocity
W_i	= the angular velocity at the previous sample instant

The difference between the present angular velocity and the previous angular velocity is equal to the angular velocity gained (ΔW) during the previous control cycle

$$W_i - W_{i-1} = \Delta W \qquad (2)$$

Dividing Eq. (2) by 2 and adding it to Eq. (1) results in the following expression for the present angular velocity

$$W_i = \frac{\Delta\theta}{\Delta T} + \frac{\Delta W}{2} \qquad (3)$$

The change in attitude is calculated from IMU measurements ($\Delta\theta_m$), and the angular velocity gained can be approximated by the previous angular velocity-to-be-gained command (W_g). Therefore, the angular velocity can be estimated with the following equation

$$W_i = \frac{\Delta\theta_M}{\Delta T} + \frac{W_G}{2} \tag{4}$$

where w_i equals the estimate of the present rotational velocity.

The IMU measurements, and consequently $\Delta\theta_m$, contain high frequency noise, however, so a lagless filter is added to this equation to get a better estimate. To begin the derivation of the lagless filter,the z-transform representation of Eq. (4) is multiplied by a term that is a unity transfer function (and therefore lagless) to produce the expression

$$W(Z) = \left[\frac{\Delta\theta_M(Z)}{\Delta T} + \frac{W_G(Z)}{2}\right]\left[\frac{1}{\frac{\tau(1-Z^{-1})}{\Delta T}+1} + \frac{\frac{\tau(1-Z^{-1})}{\Delta T}}{\frac{\tau(1-Z^{-1})}{\Delta T}+1}\right] \tag{5}$$

Since $\frac{\Delta\theta_M(Z)}{\Delta T} + \frac{W_G(Z)}{2}$ is equivalent to the z-transform of the present angular velocity, this quantity multiplied by $(1 - z^{-1})$ is equal to the z-transform of the change in angular velocity during the last sample cycle, which as stated previously is assumed equal to the previous angular velocity-to-be-gained command W_g. So

$$W_G(Z) = \left[\frac{\Delta\theta_M(Z)}{\Delta T} + \frac{W_G(Z)}{2}\right](1-Z^{-1}) \tag{6}$$

Substituting this into Eq. (5) gives

$$W(Z) = \left[\frac{\Delta\theta_M(Z)}{\Delta T} + \frac{W_G(Z)}{2}\right]\left[\frac{1}{\frac{\tau(1-Z^{-1})}{\Delta T}+1}\right] + W_G(Z)\left[\frac{\frac{\tau}{\Delta T}}{\frac{\tau(1-Z^{-1})}{\Delta T}+1}\right] \tag{7}$$

Multiplying by the denominator yields

$$\left[W(Z)\ \frac{\tau(1-Z^{-1})}{\Delta T} + 1\right] = \frac{\Delta\theta_M(Z)}{\Delta T} + \frac{W_G(Z)}{2} + W_G(Z)\frac{\tau}{\Delta T} \tag{8}$$

which is equivalent to

$$\left(\frac{\tau}{\Delta T}+1\right)W(Z) - \left(\frac{\tau}{\Delta T}\right) Z^{-1}W(Z) = \frac{\Delta\theta_M(Z)}{\Delta T} + \left(\frac{1}{2} + \frac{\tau}{\Delta T}\right) W_G(Z) \tag{9}$$

Taking the inverse z-transform of Eq.(9) gives

$$\left(\frac{\tau}{\Delta T}+1\right)W_i - \left(\frac{\tau}{\Delta T}\right)W_{i-1} = \frac{\theta_M}{\Delta T} + \left(\frac{1}{2} + \frac{\tau}{\Delta T}\right)W_G \tag{10}$$

and rearranging yields

$$W_i = \frac{\Delta\theta_M}{(\frac{\tau}{\Delta T}+1)\Delta T} + W_G \frac{\frac{\tau}{\Delta T}+\frac{1}{2}}{(\frac{\tau}{\Delta T}+1)} + W_{i-1}\frac{(\frac{\tau}{\Delta T})}{(\frac{\tau}{\Delta T}+1)} \qquad (11)$$

or

$$W_i = \frac{\Delta\theta_M}{(\tau+\Delta T)} + W_G\frac{(\tau+\frac{\Delta T}{2})}{(\tau+\Delta T)} + W_{i-1}\frac{\tau}{(\tau+\Delta T)} \qquad (12)$$

which is the desired angular velocity estimation equation derived using the lagless filter approach. The time constant τ can be varied depending on the relative uncertainties in θ_m and W_g to obtain good estimation performance. The value of τ used in the computer simulation was chosen empirically by comparing the estimator's performances for several values of τ.

The derivation of the "least-squares" estimator as presented in Reference (3) is as follows:

The least-square method estimates the angular velocity of the space vehicle from N attitude measurements taken during a coast period of duration T_{coast}, where T_{coast} equals N - 1 sample intervals. A hypothetical plot of θ versus time for one axis is shown in Figure 2. The estimate is derived by finding the slope of the straight line that provides a "best fit" to the N attitude measurements. The "best fit" straight line can be defined mathematically as the line

$$\theta(t) = mt = b \qquad (13)$$

that minimizes the sum (S) of the squares of the errors between the line and the measured attitude at each of the N sample instants, where m is the slope of the line and b is the intercept. S is given by the expression

$$S = \sum_{i=1}^{N} (\theta(t_i) - \theta_i)^2 \qquad (14)$$

where t_i is the ith sample instant and θ_i is the ith value of θ. The desired angular velocity estimate is

$$W = \frac{d}{dt}\,\theta(t) = m \tag{15}$$

From the method of least-squares, the value of W is given by

$$W = m = \frac{N\sum_{i=1}^{N} t_i\theta_i - \left(\sum_{i=1}^{N} t_i\right)\left(\sum_{i=1}^{N} \theta_i\right)}{N\sum_{i=1}^{N} t_i^2 - \left(\sum_{i=1}^{N} t_i\right)^2} \tag{16}$$

From Figure 2, it can be seen that t_i can be expressed as

$$t_i = t_1 + (i-1)\,\Delta T \tag{17}$$

where t_1 is the initial sample instant and ΔT is the sample interval. It can be shown that W does not depend on the value of t_1, so t_1 was arbitrarily chosen to equal ΔT, reducing Eq. (17) to

$$t_i = i\,\Delta T \tag{18}$$

The following identities can be written

$$\sum_{i=1}^{N} i = \frac{N(N+1)}{2} \tag{19}$$

$$\sum_{i=1}^{N} i^2 = \frac{N(N+1)(2N+1)}{6} \tag{20}$$

Substituting Eq. (18) into Eq. (16) and using the above identities results in the expression

$$W = \frac{N\,\Delta T \sum_{i=1}^{N} i\,\theta_i - \frac{N(N+1)\,\Delta T}{2} \sum_{i=1}^{N} \theta_i}{\frac{N^2(N+1)(2N+1)\,\Delta T^2}{6} - \left(\frac{N(N+1)\,\Delta T}{2}\right)^2} \qquad (21)$$

which can be reduced to yield the expression

$$W = \frac{12 \sum_{i=1}^{N} \left(i - \frac{N+1}{2} \right) \theta_i}{(N^3-N)\Delta T} \qquad (22)$$

This is the desired least-squares coast angular velocity estimation equation. When this equation is implemented, it can be written as

$$W = \sum_{i=1}^{N} (K_1 i + K_2)\,\theta_i \qquad (23)$$

where:

$$K_1 = \frac{12}{(N^3-N)\,\Delta T} \qquad (24)$$

and

$$K_2 = \frac{-12\,\frac{N+1}{2}}{(N^3-N)\,\Delta T} \qquad (25)$$

Single-Axis-Rotation (SAR)

According to a Theorem of Euler, the attitude of a body can be changed from any given orientation to any other orientation by rotating the body about an axis that is fixed to the vehicle and stationary in inertial space. The attitude error between the space vehicle's present attitude and a desired attitude commanded by the guidance routine can be represented by a three-element vector based on such a rotation. The direction of the vector defines the single axis about which the vehicle must rotate to reach the desired attitude, and the magnitude of the vector represents the angle through which the vehicle must rotate about this axis. The quaternion is a compact form for representing this single fixed axis and angle referred to by Euler's Theorem. The advantage of the quaternion lies in its ability to define the rotational relationship between two coordinate systems using only four numbers as opposed to the nine elements of a direction cosine matrix. The principal impediment in using quaternions has been that the direction cosine matrix, rather than the quaternion, is the desired end product of computation. When the object of the computation is to obtain the control error it is not necessary to define the direction cosine matrix. Optimum control, in the least angle sense, is performed by defining the control error in terms of the single axis and angle of Euler's Theorem. The problem can be solved completely in terms of quaternions. The attitude error used in this control system is caluculated as follows:

The conjugate of the navigation quaternion is multiplied by the guidance quaternion to form the single-axis-rotation quaternion

$$\overline{Q}_{SAR} = \overline{Q}^{*}_{NAV} \quad \overline{Q}_{GUID} \tag{26}$$

such that

$$\overline{Q}_{SAR} = q_1 + q_2\,\overline{i} + q_3\overline{j} + q_4\overline{k} \tag{27}$$

corresponding to

$$\overline{Q}_{SAR} = (\cos\ (\tfrac{\alpha}{2}),\ \overline{R}\ \sin\ (\tfrac{\alpha}{2})) \tag{28}$$

where: $\overline{R}$= Euler axis unit vector

α= rotation angle about Euler axis unit vector

and calculating for $\overline{R}$ we get

$$R_i = q_2/\sin(\tfrac{\alpha}{2})$$
$$R_j = q_3/\sin(\tfrac{\alpha}{2}) \tag{29}$$
$$R_k = q_4/\sin(\tfrac{\alpha}{2})$$

The attitude error vector ($\overline{E}$) is then found to be

$$\overline{E} = 2\overline{R}\ \sin\ (\tfrac{\alpha}{2}) \tag{30}$$

It is this vector representation of the attitude error that is used by the control law.

Control Law

A common control logic is used to perform the four types of maneuvers: large attitude maneuver, axial velocity maneuver, terminal condition maneuver, and attitude hold mode. The basic equation of the control law can be derived for the idealized case of a large attitude maneuver whose duration is known at the beginning of the maneuver. Since the duration of the maneuver is known initially, the time remaining at any point in the maneuver can be easily computed. The control law is designed to compute a angular velocity command ($\overline{W}c$) each control instant, such that if this velocity is achieved at the end of the upcoming control cycle interval, and then linearly reduced to zero in the remaining maneuver time (T_{GO}), the attitude error will also be reduced to zero.

Consequently, the area under the angular velocity curve in Figure 3 is set equal to the estimated attitude error ($\overline{E}$), so the attitude error will be zero when the angular velocity goes to zero.

$$\text{AREA} = \frac{W + W_c}{2} \quad (\Delta T) + \frac{W_c}{2} \quad (T_{GO} - \Delta T) \tag{31}$$

so that in vector form

$$\overline{E} = \frac{\overline{W} + \overline{W}_c}{2} (\Delta T) + \frac{\overline{W}_c}{2} (T_{GO} - \Delta T) \tag{32}$$

Cancelling gives

$$\overline{E} = \frac{\overline{W}}{2} \quad (\Delta T) + \frac{\overline{W}_c}{2} (T_{GO}) \tag{33}$$

and solving for $\overline{W}c$ yields the equation for the commanded angular velocity

$$\overline{W}_c = \frac{2\overline{E} - \Delta T\overline{W}}{T_{GO}} \tag{34}$$

The equation for the angular velocity-to-be-gained is

$$\overline{W}_G = \overline{W}_c - \overline{W} \tag{35}$$

Since the duration of a maneuver is not known in advance, a method to determine the value of T_{GO} is redefined as a varying parameter related to but not always equal to the remaining maneuver time.

For the large attitude maneuver, T_{GO} is initialized to a small value (to account for angular velocity coupling) and incremented each control cycle by the control cycle interval (ΔT) until the magnitude of the attitude error has been reduced to half its original magnitude. The remainder of the maneuver is executed with T_{GO} being decremented each control cycle by ΔT until the end of the maneuver.

For the attitude hold mode, Eq. (34) used with a fixed value of T_{GO} will drive both the attitude error and angular velocity towards zero with the speed of response depending on the value of T_{GO}. However, in the attitude hold application, the jet firings must be inhibited when these states are near zero to prevent excessive fuel usage as the vehicle limit cycles. Therefore, if the magnitudes of the attitude error and the angular velocity are both less than specified deadbands, the angular velocity-to-be-gained command Eq. (35) is set equal to zero. If the attitude error is large, a small value of T_{GO} is used which makes the gains in Eq. (34) large, and causes the error to be quickly reduced. If the attitude error is small, a large value of T_{GO} is used, causing the angular velocity command to be small. A small angular velocity is desired so the attitude error will remain within its deadband for an appreciable time to minimize fuel consumption.

The terminal condition maneuver and the axial velocity maneuver also make use of a linear velocity-to-be-gained vector, $\overline{V}_G$.

$$\overline{V}_G = \overline{V}_c - (\overline{V}_{IMU} - \overline{W} \times \overline{R}_{CM-IMU}) \qquad (36)$$

where:	$\overline{V}_C$	= linear velocity command vector from guidance
	$\overline{V}_{IMU}$	= measured IMU linear velocity vector
	$\overline{W}$	= estimated angular velocity vector
	$\overline{R}_{CM-IMU}$	= position vector from center of mass to the IMU position

To combine the linear velocity-to-be-gained vector with the angular velocity-to-be-gained vector effectively requires that the magnitude of the linear velocity-to-be-gained command be limited to a value that can be nulled in one control cycle interval. This is a result of scaling done by the jet selection routine which would otherwise impair attitude error control. This will be demonstrated in the next section. In a method analogous to the derivation of Eq. (34), it can be shown that for terminal condition control the angular velocity command is

$$\overline{W}_C = \frac{2\overline{E} - \Delta T\overline{W}}{2(\Delta T + T_{COAST})} \tag{37}$$

This equation is identical to Eq. (34) with

$$T_{GO} = 2\ (\Delta T + T_{COAST}) \tag{38}$$

where: T_{coast} = coast period prior to final nulling burn

For terminal condition control and axial velocity maneuvers, both angular velocity-to-be-gained and limited linear velocity-to-be-gained are nulled by the jet selection routine.

Jet Selection Logic

Assuming a rigid vehicle with constant mass properties, the vehicle equations of motion may be expressed as

$$\overline{T} = \frac{d\overline{H}}{dT} + \overline{W} \times \overline{H} \tag{39}$$

and

$$\overline{F}_I = M\frac{d\overline{V}_I}{dt} \tag{40}$$

where:

$\overline{T}$ = 3-dimensional torque vector whose elements are the net jet torques about each of the three body axes

$\overline{F}_I$ = 3-dimensional force vector whose elements are the net jet forces along three orthogonal inertial references axes

$\overline{W}$ = 3-dimensional angular velocity vector whose elements are the angular velocities about the three body axes

$\overline{H}$ = 3-dimensional angular momentum vector

$\overline{H}$ = $I\overline{W}$, where I is the inertial tensor of the vehicle

$\overline{V}_I$ = 3-dimensional linear velocity vector whose elements are the linear velocity components along three inertial reference axes

M = vehicle mass

The above equations can be simplified as follows for the particular problem of achieving an end state of constant attitude and constant linear velocity.

First, the angular velocities comprising $\overline{W}$ can be assumed to be small enough that the term $\overline{W} \times \overline{H}$ can be neglected, resulting in

$$\overline{T} = I\frac{d\overline{W}}{dt} \tag{41}$$

(Here, it may be noted that the elements of $\overline{W} \times \overline{H}$ are called "angular velocity coupling terms").

Second, the effects of angular rotation on the direction of the linear acceleration resulting from the jet forces can be neglected, so the Eq. (40) can be rewritten in terms of the body-axis force vector, $\overline{F}$, and body-axis linear velocity, $\overline{V}$.

$$\overline{F} = M\,\frac{d\overline{V}}{dt} \tag{42}$$

(NOTE: These approximations are employed only in the autopilot implementation and not in the vehicle simulation.)

Both Eqs. (41) and (42) can be solved for the angular and linear acceleration vectors

$$\frac{d\overline{W}}{dt} = I^{-1}\,\overline{T} \tag{43}$$

$$\frac{d\overline{V}}{dt} = \frac{1}{M}\,\overline{F} \tag{44}$$

The jet selection routine of the phase space autopilot computes a set of firing times (t_{JET}) such that the elements of the change in $\overline{W}$, as obtained by integrating Eq. (43)

$$\Delta\overline{W} = I^{-1}\int_0^t \overline{T}\,dt \tag{45}$$

and the elements of the change in $\overline{V}$, as obtained by integrating Eq. (44)

$$\Delta\overline{V} = \frac{1}{M}\int_{0}^{t} \overline{F}\ dt \tag{46}$$

are equal to the corresponding elements of the velocity-to-be-gained vector $\overline{\Omega}$.

$$\overline{\Omega} = \begin{bmatrix} W_{G_X} \\ W_{G_Y} \\ W_{G_Z} \\ V_{G_X} \\ V_{G_Y} \\ V_{G_Z} \end{bmatrix} \tag{47}$$

The $\overline{W}_g$ and $\overline{V}_g$ terms are from Eq. (35) and (36) respectively. Thus, the jet selection problem consists of solving

$$\overline{\Omega} = \overline{A}_{AM}\ \overline{t}_{JET} \tag{48}$$

for $\overline{t}_{JET}$, where:

$$\overline{A}_{AM} = \begin{bmatrix} \overline{a}_1, \overline{a}_2 \ . \ . \ . \ \overline{a}_m \\ \overline{A}_1, \overline{A}_2 \ . \ . \ . \ \overline{A}_m \end{bmatrix} \tag{49}$$

$\overline{a}_i = I^{-1}\,\overline{R}_i \times \overline{F}_i$ (50)

$\overline{A}_i = \overline{F}_i/M$ (51)

I^{-1} = inverse inertia matrix

$\overline{F}_i$ = force vector of i^TH jet

$\overline{R}_i$ = position vector of iTH jet

M = vehicle mass

m = total number of jets

Flexibility in jet selection was required to account for changing vehicle mass properties. The "Dot Product Method", described in Reference (1), was chosen to solve Eq. (48). The key feature of this method is the computation of the dot products of the acceleration vector of each jet with the velocity-to-be-gained vector and the use of this dot product to determine the single jet firing time which will produce the largest beneficial change in the velocity-to-be-gained vector. This method is an iterative process and is summarized below:

The jet acceleration matrix from Eq. (49) is modified by C_1, a weighting factor equivalent to the lever arm effect from the PBV center of gravity to the control point.

$$\bar{A}_{MAM} = \begin{bmatrix} c_1\bar{a}_i, \; c_1\bar{a}_2 \; . \; . \; . \; c_1\bar{a}_m \\ \bar{A}_i, \; \bar{A}_2, \; . \; . \; . \; \bar{A}_m \end{bmatrix} \tag{52}$$

Calculating normalizing factors and normalize Eq. (52)

$$N_i = \frac{1}{|\bar{A}_{MAM}|} \tag{53}$$

$$\bar{A}_{NAM} = \begin{bmatrix} C_1\bar{a}_iN_i, \; C_1\bar{a}_2N_2 \; . \; . \; . \; C_1\bar{a}_mN_m \\ \bar{A}_iN_i, \; \bar{A}_2N_2 \; . \; . \; . \; \bar{A}_mN_m \end{bmatrix} \tag{54}$$

Modify the velocity-to-be-gained vector, Eq. (47) by C_1

$$\bar{X} = \begin{bmatrix} C_1\bar{W}_G \\ \bar{V}_G \end{bmatrix} \tag{55}$$

Initialize t_{JET} to zero, set $\bar{R} = \bar{X}$, and begin iteration.

Calculate the dot products of the modified jet acceleration vectors Eq. (54) with the modified velocity-to-be-gained vector Eq. (55)

$$D_i = \bar{A}_{NAM_i} \bullet \bar{R} \tag{56}$$

The i^{TH} jet with the largest positive dot product will provide maximum control authority. Calculate the time to fire this jet

$$\Delta t = D_i N_i \tag{57}$$

Update the total time to fire this jet

$$t_{JET_i} = t_{JET_i} + \Delta t \tag{58}$$

Calculate the residue velocity-to-be-gained vector

$$\overline{R} = \overline{X} - \overline{A}_{MAM} \overline{t}_{JET} \tag{59}$$

Iterate until $\overline{R}$ is sufficiently small and then scale $\overline{t}_{JET}$ to less than or equal to one control cycle.

$$\overline{t}_{JET} = \frac{\Delta T}{t_{MAX}} \overline{t}_{JET} \tag{60}$$

where: t_{MAX} = largest jet firing time
ΔT = control cycle

We now have time durations for firing the jets. However, the jets do not attain full thrust instantaneously, nor do they decay instantaneously. $\overline{t}_{JET}$ is adjusted for this effect by the following curve fit:

$$\overline{t}_{JET} = C_1 + C_2 \overline{t}_{JET} + C_3 \overline{t}^2_{JET} \tag{61}$$

If the jets all begin to fire at the start of each control cycle interval, the angular acceleration produced will not be constant during the full control cycle. This produces a small source of error since the control laws and estimation routine assume an attitude change each control cycle that is based on constant angular acceleration. This error has been eliminated by centering the jet firing times within the control cycle. With centering, the attitude change produced each control cycle will be the same as it would be with constant angular acceleration.

The dot product jet selection method provides more versatility than a table lookup routine and uses less software and memory than Simplex Linear programing methods. The dot product method can also account for jet failures and changing vehicle mass properties. However, a problem may develop for a system with a very large number of jets. Many iterations would be needed to find the best jet combination, possibly causing an unacceptable burden on the flight computer.

Computer Simulation

EASY5 was the computer code chosen for the simulation primary because it is widely used by Flight Control Technology staff and is supported on computer systems throughout the company. Flow charts of the simulation can be seen in Figures 4-9. Source code for the Model Generation File can be found in Appendix A. Source code for the Analysis File can be found in Appendix B. Source code for miscellaneous subroutines used in the simulation can be found in Appendix C. The analysis file provided in Appendix B was used to generate the terminal maneuver simulation results contained in the next section. There was a liberal use of comment cards and those people having EASY5 experience should have little difficulty in running the simulation. It should be noted that source listings have been incorporated directly into the manuscript from the original source files to assure accuracy in reproduction.

Simulation Results

The performance characteristics of the control laws presented herein were evaluated using an EASY5 computer simulation. The vehicle is simulated using a constant mass, rigid body model. The dimensions of the vehicle, the placement of the reaction control jets, and the locations of the center of gravity, control point and IMU are shown in Figure 10. To represent uncertainties in the knowledge of vehicle properties, the values of the vehicle's mass, moments and products of inertia, center of gravity, thruster magnitude and orientation were statistically varied. Vehicle properties and Monte Carlo Perturbations are shown in Figure 11 and Figure 12, respectively. The IMU linear velocity measurements were assumed to have a normally distributed random noise with a standard deviation of 0.00167 feet per second. The attitude measurements were assumed to have a normally distributed random noise with a standard deviation of 0.0167 degrees. The reaction control jets were modeled as shown in Figure 13. The jets were assumed to fire to a precision of 300 micro seconds with a minimum firing time of 0.01 seconds. The computer cycle and control cycle for all runs were 0.010 and 0.150 seconds respectively.

The large attitude maneuver results can be seen in Figures 14-20. The initial value of T_{GO} was 0.31 second and 20 degrees per second was used for the angular velocity limit. The final attitude error and angular velocity magnitudes were 0.144 degrees and 0.675 degrees per second, which are well within the range of values controllable by terminal condition control or attitude hold control.

The terminal condition maneuver results can be seen in Figures 21-29. The terminal condition simulation employed the following values of parameters:

1) an attitude error deadband of 0.5 degrees
2) a linear velocity limit of 0.04 feet per second
3) a linear velocity deadband of 0.03 feet per second
4) an angular velocity deadband of 0.5 degrees per second
5) a coast sample interval of 0.75 seconds
6) a dot product weighting factor of 1.0
7) a residue deadband of 0.004

These values were found to produce good performance for this configuration. The final attitude error, angular velocity and linear velocity magnitudes were 0.334 degrees, 0.129 degrees per second, and 0.0015 feet per second respectively.

The attitude hold mode results can be seen in Figures 30-36. The attitude hold simulation employed the following values of parameters:

1) a value of the attitude error at which T_{GO} is switched of 0.8 degrees
2) an attitude error deadband of 0.5 degrees
3) An angular velocity deadband of 1.0 degrees per second
4) a value of SMALL T_{GO} of 1.0
5) a value of LARGE T_{GO} of 10.0

It can be seen from the figures that both attitude and angular velocity magnitudes were held within the deadbands with a minimum usage of fuel.

The axial velocity maneuver results can be seen in Figures 37-45. The axial velocity simulation employed the following values of parameters:

1) an attitude error deadband of 1.0 degrees
2) a linear velocity limit of 0.60 feet per second
3) a dot product weighting factor of 1.0
4) a value of T_{GO} of 0.90
5) a residue dead band of 0.004

It can be seen from the figures that attitude control was well maintained and fuel usage was minimal during the axial velocity maneuver.

Conclusions and Recommendations

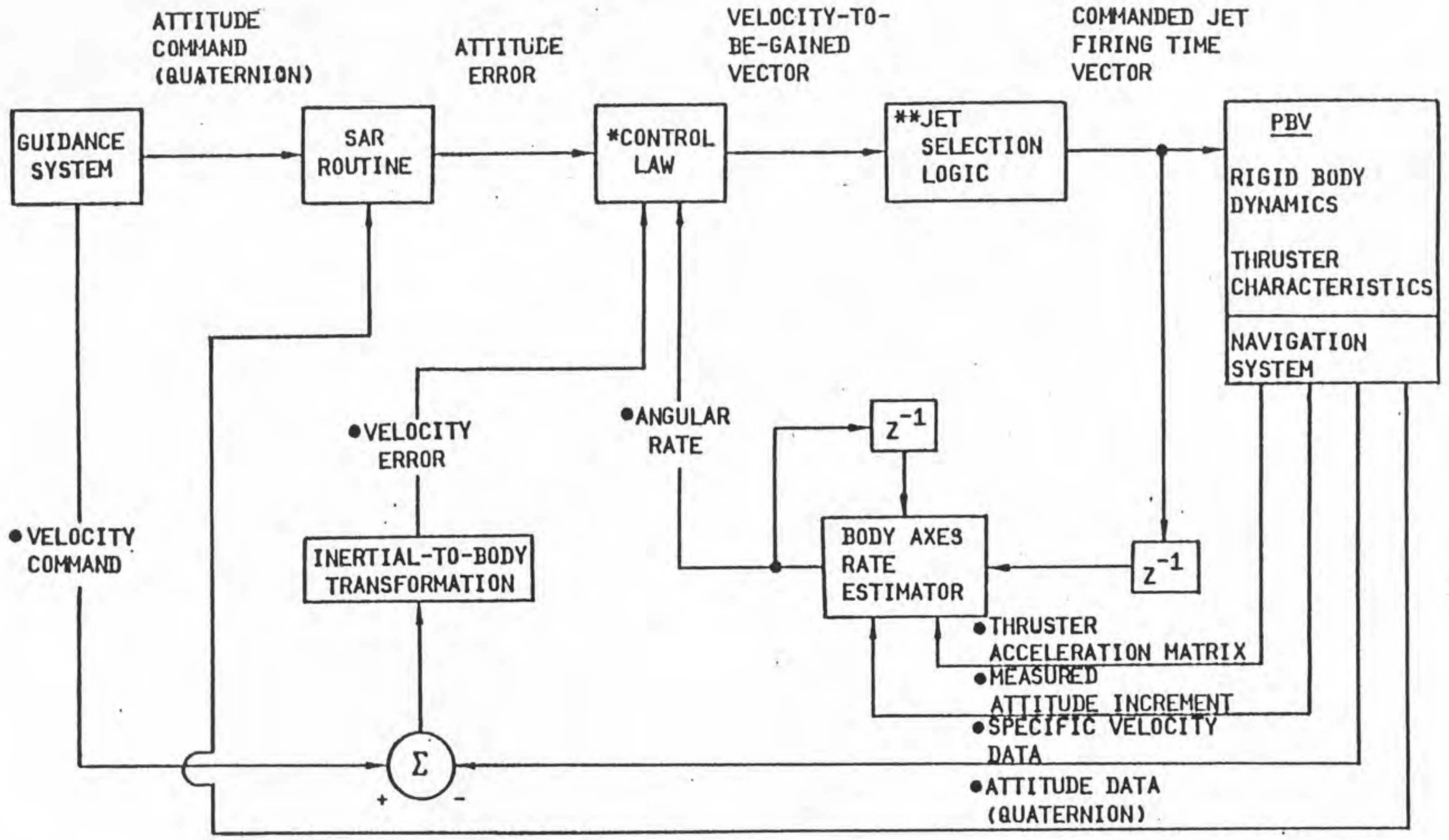
The simulation results show that the control laws perform the desired control tasks in the presence of uncertainties and disturbances. The terminal errors produced by the large attitude maneuver were shown to be within the initial error capabilities of the terminal condition control law and the attitude hold control law. The attitude hold and axial velocity control laws were shown to be fuel efficient. The Dot Product Jet Selection method was found to be powerful and computationally efficient. The phase space technique, which coordinates control for all six degrees of freedom, was found to rapidly converge on an end state of constant commanded attitude and constant commanded linear velocity. Reaction control system performance can be readily assessed by making minor parameter changes and performing Monte Carlo style analysis. In conclusion, a generalized six degrees of freedom PBV simulation has been developed, verified and is available for distribution.

It is recommended that the Gradient Jet Selection method be incorporated into the simulation to further reduce computer memory requirements. It is also recommended that a Kalman Rate Estimator be studied to determine if terminal condition control performance can be improved. It can not be over emphasized that this study is a multi-year task requiring updates as the technologies mature if we are to remain in a competitive posture.

Prepared By [illegible]

R. G. Borst
Flight Control Technology, 2-3634

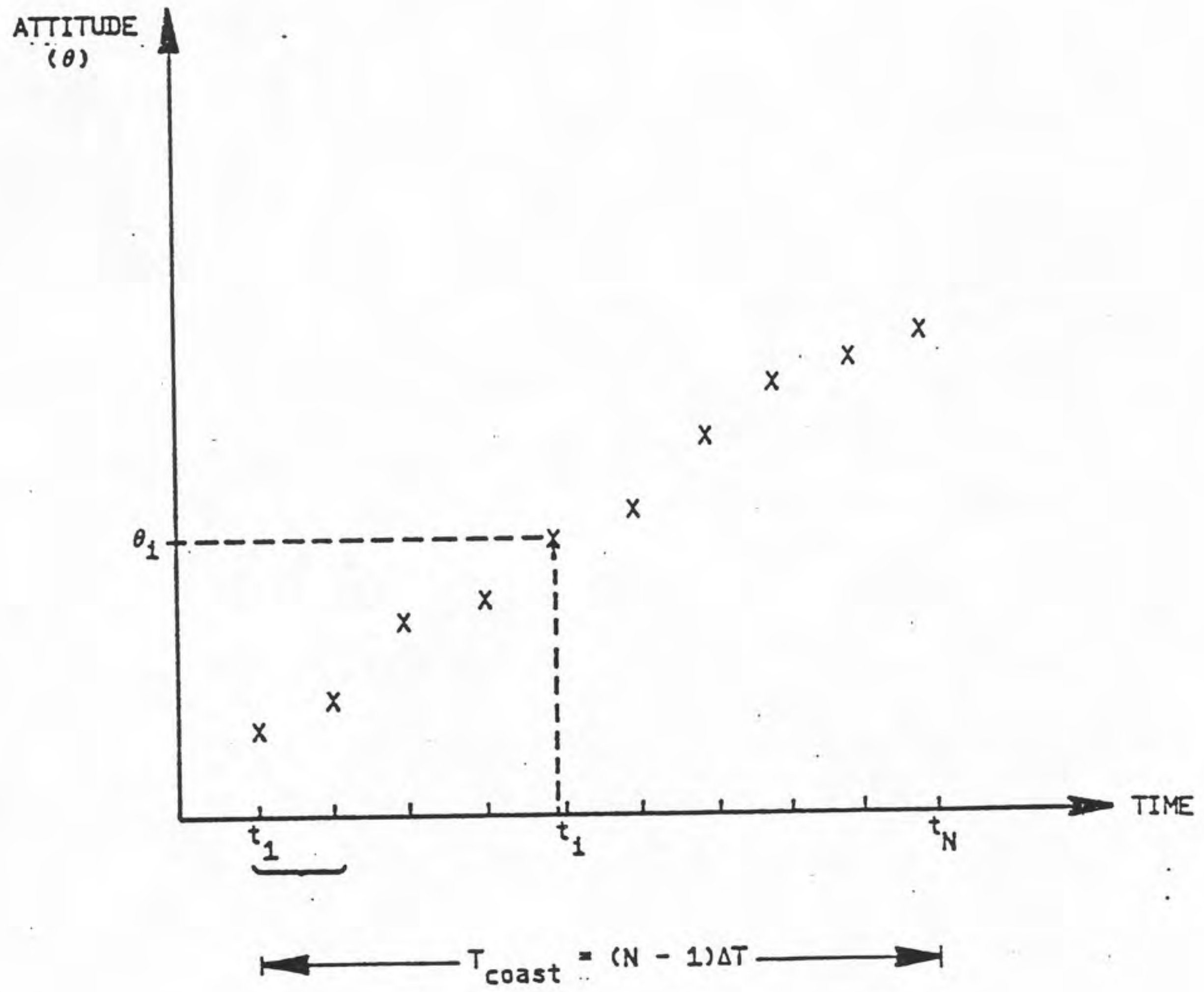
TABLES AND FIGURES



*CONTROL LAW BASED ON CSDLs PHASE SPACE UNIFIED AUTOPILOT

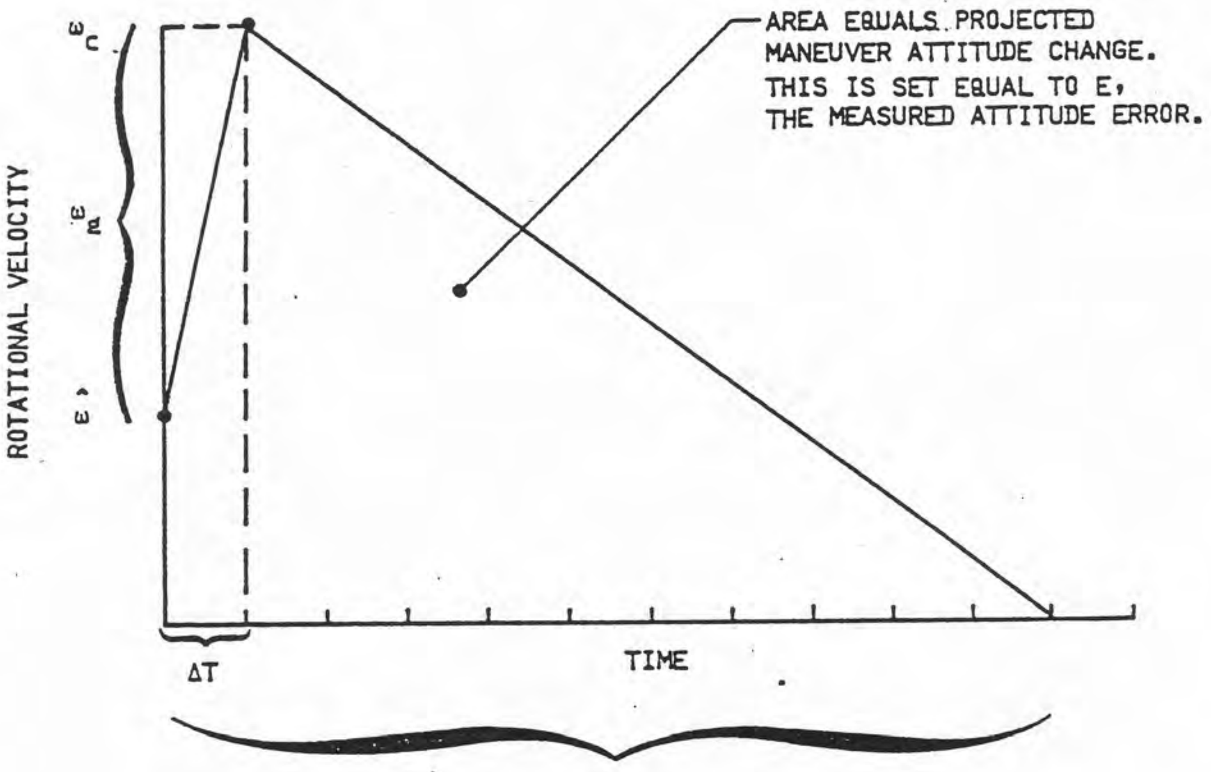
**JET SELECTION LOGIC BASED ON CSDLs DOT PRODUCT METHOD

FIGURE 1. PBV CONTROL SYSTEM



t_i	= i^{TH} SAMPLE INSTANT
θ_i	= i^{TH} ATTITUDE MEASUREMENT
ΔT	= SAMPLE INTERVAL
N	= NUMBER OF MEASUREMENTS
T_{COAST}	= DURATION OF COAST PERIOD

FIGURE 2. HYPOTHETICAL PLOT OF THE MEASURED "ATTITUDE" VS. TIME



ω_c = COMMANDED ROTATIONAL VELOCITY

ω_g = ROTATIONAL VELOCITY-TO-BE-GAINED

ω = PRESENT ESTIMATED ROTATIONAL VELOCITY

ΔT = CONTROL CYCLE INTERVAL

T_{go} = REMAINING MANEUVER TIME

FIGURE 3. ROTATIONAL VELOCITY VS. TIME GRAPH FOR ONE AXIS SHOWING HOW ω_c AND ω_g ARE DETERMINED FOR A LARGE ATTITUDE MANEUVER.

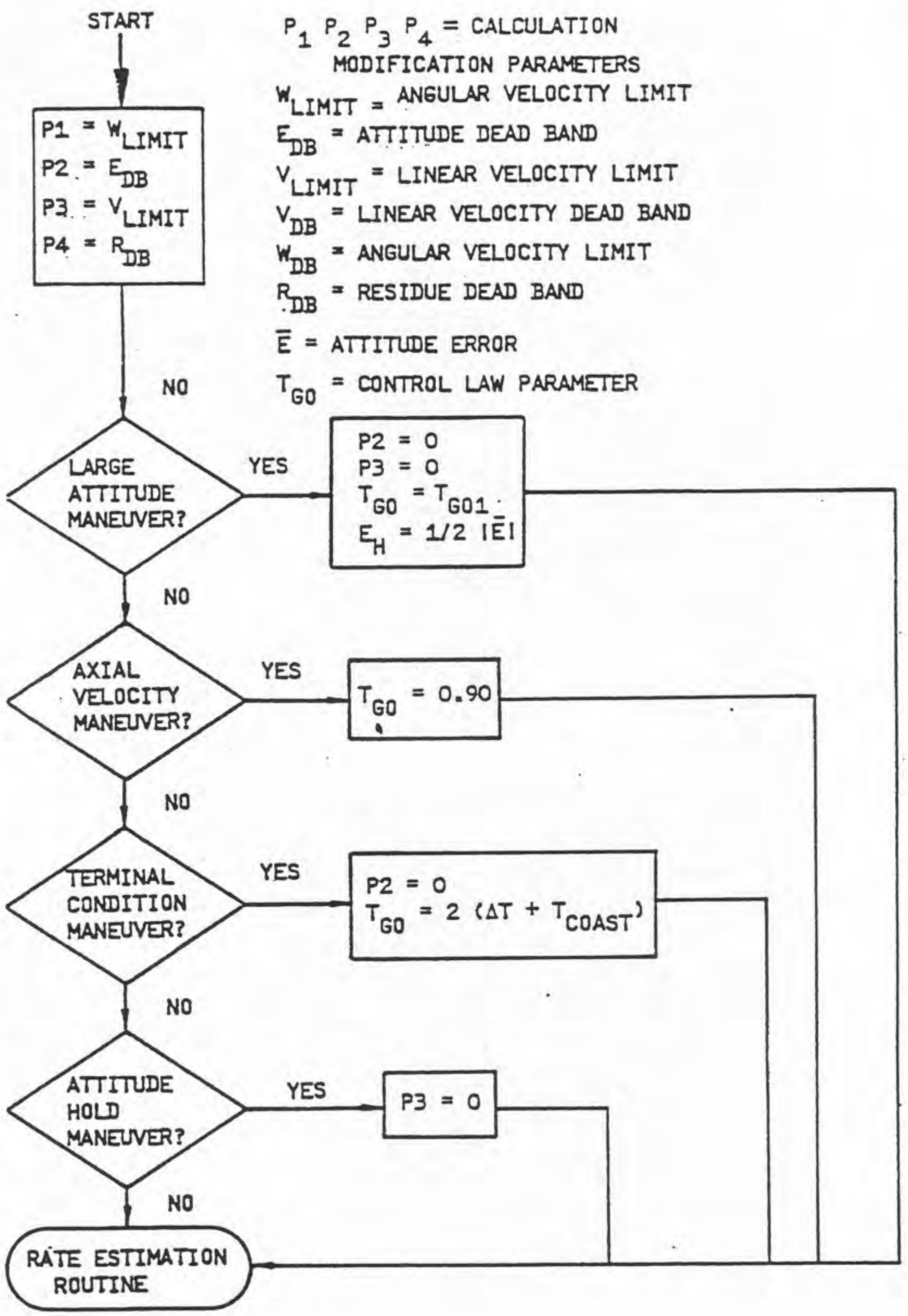
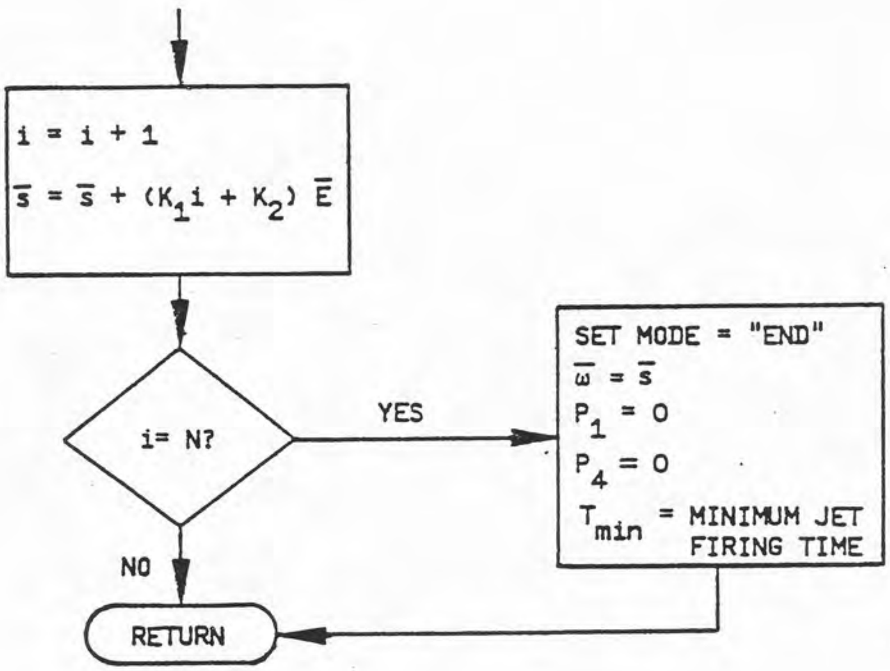

START
$P1 = W_{LIMIT}$
$P2 = E_{DB}$
$P3 = V_{LIMIT}$
$P4 = R_{DB}$
$P_1\ P_2\ P_3\ P_4$ = CALCULATION MODIFICATION PARAMETERS
W_{LIMIT} = ANGULAR VELOCITY LIMIT
E_{DB} = ATTITUDE DEAD BAND
V_{LIMIT} = LINEAR VELOCITY LIMIT
V_{DB} = LINEAR VELOCITY DEAD BAND
W_{DB} = ANGULAR VELOCITY LIMIT
R_{DB} = RESIDUE DEAD BAND
$\bar{E}$ = ATTITUDE ERROR
T_{GO} = CONTROL LAW PARAMETER
NO
LARGE ATTITUDE MANEUVER?
YES
$P2 = 0$
$P3 = 0$
$T_{GO} = T_{GO1}$
$E_H = 1/2\ |\bar{E}|$
NO
AXIAL VELOCITY MANEUVER?
YES
$T_{GO} = 0.90$
NO
TERMINAL CONDITION MANEUVER?
YES
$P2 = 0$
$T_{GO} = 2\ (\Delta T + T_{COAST})$
NO
ATTITUDE HOLD MANEUVER?
YES
$P3 = 0$
NO
RATE ESTIMATION ROUTINE

FIGURE 4. AUTOPILOT INITIALIZATION ROUTINE



$i = i + 1$
$\overline{s} = \overline{s} + (K_1 i + K_2)\,\overline{E}$
$i = N?$
YES
NO
SET MODE = "END"
$\overline{\omega} = \overline{s}$
$P_1 = 0$
$P_4 = 0$
T_{min} = MINIMUM JET FIRING TIME
RETURN

i	= SAMPLE COUNTER
$\overline{s}$	= SUM OF WEIGHTED ATTITUDE ERRORS
K_1, K_2	= ESTIMATION EQUATION CONSTANTS
$\overline{E}$	= ATTITUDE ERROR
N	= NUMBER OF SAMPLES IN COAST PERIOD
$\overline{\omega}$	= ANGULAR VELOCITY ESTIMATE
P_1, P_4	= CALCULATION MODIFICATION PARAMETER
T_{min}	= JET SELECTION PARAMETER

FIGURE 5. AUTOPILOT COAST ANGULAR VELOCITY ESTIMATION ROUTINE

$$\bar{\omega} = K_1 \, \Delta\bar{\theta}_m + K_2\bar{\omega}_g + K_3\bar{\omega}_p$$

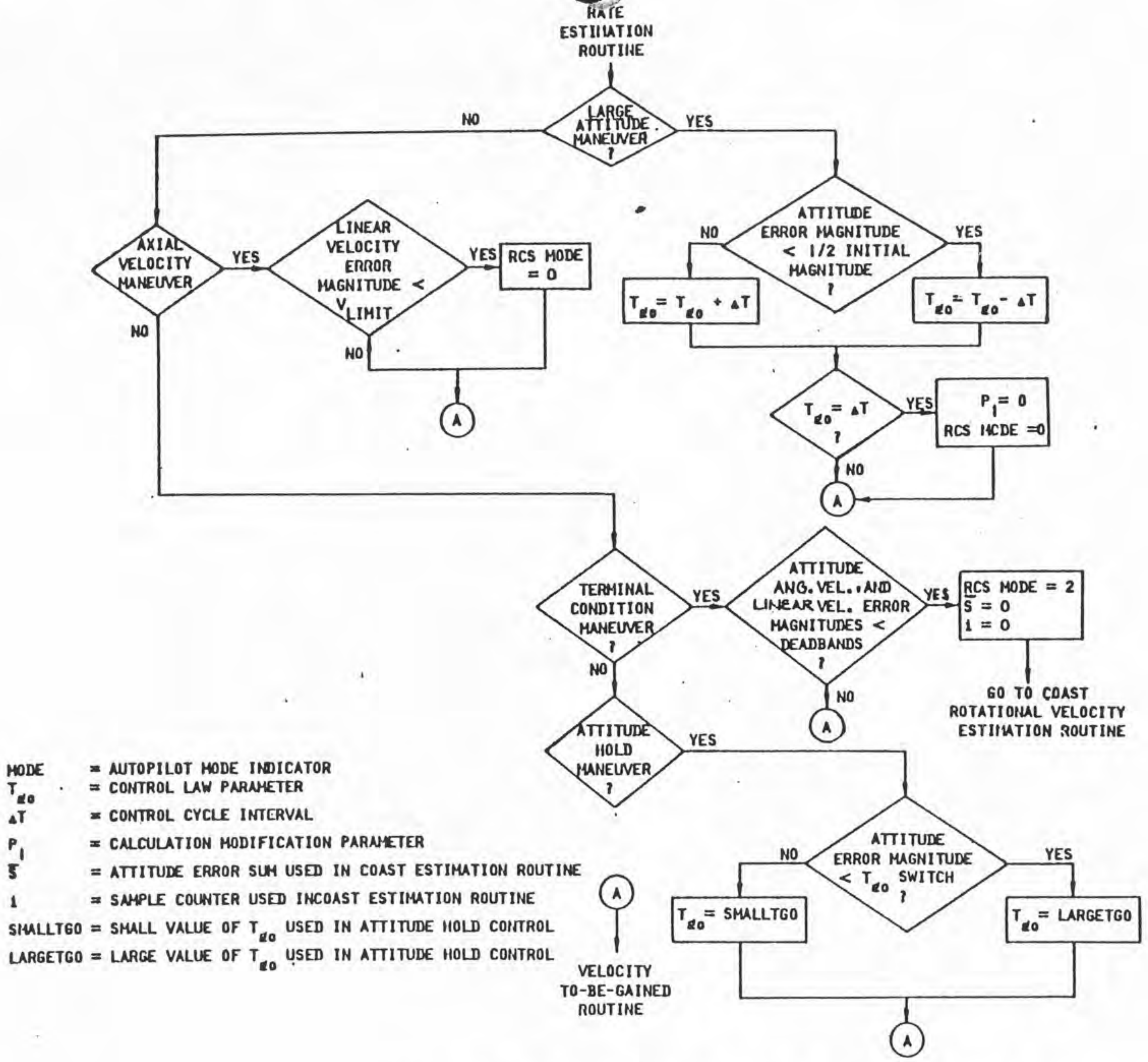
$$\bar{\omega}_p = \bar{\omega}$$

$$\bar{V} = \bar{V}_d - (\bar{V}_{imu} - (\bar{\omega} X r_{cm-imu}))$$

CONTROL DECISION ROUTINE

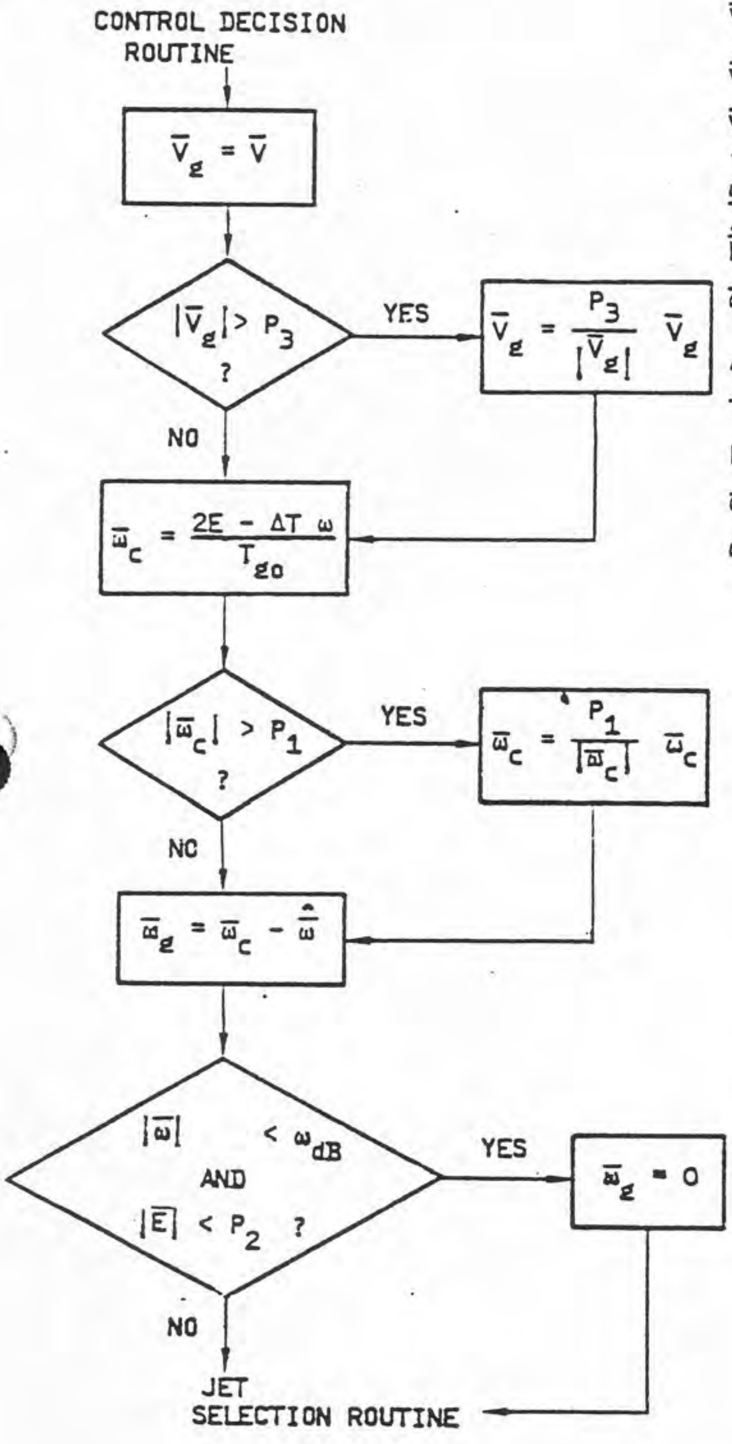
$\bar{\omega}$	= ANGULAR VELOCITY ESTIMATE
$K_1, K_2 K_3$	= ESTIMATION EQUATION CONSTANTS
$\Delta\bar{\theta}_m$	= MEASURED ATTITUDE INCREMENT
$\bar{\omega}_g$	= ANGULAR VELOCITY-GO-BE-GAINED COMMAND
$\bar{\omega}_p$	= PREVIOUS ANGULAR VELOCITY ESTIMATE
$\bar{V}$	= LINEAR VELOCITY ESTIMATE
$\bar{V}_d$	= DESIRED LINEAR VELOCITY (FROM GUIDANCE ROUTINE)
$\bar{V}_{imu}$	= LINEAR VELOCITY MEASURED BY IMU
$\bar{r}_{cm-imu}$	= VECTOR BETWEEN CENTER OF MASS POSITION AND IMU POSITION

FIGURE 6. AUTOPILOT RATE ESTIMATION ROUTINE


RATE ESTIMATION ROUTINE
LARGE ATTITUDE MANEUVER ?
NO
YES
AXIAL VELOCITY MANEUVER
YES
NO
LINEAR VELOCITY ERROR MAGNITUDE < V_{LIMIT}
YES
NO
RCS MODE = 0
A
ATTITUDE ERROR MAGNITUDE < 1/2 INITIAL MAGNITUDE ?
NO
YES
$T_{go} = T_{go} + \Delta T$
$T_{go} = T_{go} - \Delta T$
$T_{go} = \Delta T$?
YES
NO
$P_1 = 0$
RCS MODE =0
A
TERMINAL CONDITION MANEUVER ?
YES
NO
ATTITUDE ANG. VEL. AND LINEAR VEL. ERROR MAGNITUDES < DEADBANDS ?
YES
NO
A
RCS MODE = 2
$\bar{S} = 0$
$i = 0$
GO TO COAST ROTATIONAL VELOCITY ESTIMATION ROUTINE
ATTITUDE HOLD MANEUVER ?
YES
ATTITUDE ERROR MAGNITUDE < T_{go} SWITCH ?
NO
YES
T_{go} = SMALLTGO
T_{go} = LARGETGO
A
A
VELOCITY TO-BE-GAINED ROUTINE

MODE	= AUTOPILOT MODE INDICATOR
T_{go}	= CONTROL LAW PARAMETER
ΔT	= CONTROL CYCLE INTERVAL
P_1	= CALCULATION MODIFICATION PARAMETER
$\bar{S}$	= ATTITUDE ERROR SUM USED IN COAST ESTIMATION ROUTINE
i	= SAMPLE COUNTER USED INCOAST ESTIMATION ROUTINE
SMALLTGO	= SMALL VALUE OF T_{go} USED IN ATTITUDE HOLD CONTROL
LARGETGO	= LARGE VALUE OF T_{go} USED IN ATTITUDE HOLD CONTROL

FIGURE 7. AUTOPILOT CONTROL DECISION ROUTINE


CONTROL DECISION ROUTINE
$\bar{V}_g = \bar{V}$
$|\bar{V}_g| > P_3$?
YES
$\bar{V}_g = \frac{P_3}{|\bar{V}_g|} \bar{V}_g$
NO
$\bar{\omega}_c = \frac{2E - \Delta T\ \omega}{T_{go}}$
$|\bar{\omega}_c| > P_1$?
YES
$\bar{\omega}_c = \frac{P_1}{|\bar{\omega}_c|} \bar{\omega}_c$
NC
$\bar{\omega}_g = \bar{\omega}_c - \hat{\bar{\omega}}$
$|\bar{\omega}| < \omega_{dB}$ AND $|\bar{E}| < P_2$?
YES
$\bar{\omega}_g = 0$
NO
JET SELECTION ROUTINE

$\bar{V}_g$	= LINEAR VELOCITY-TO-BE-GAINED
$\bar{V}$	= LINEAR VELOCITY ESTIMATE
$\bar{V}_{limit}$	= LINEAR VELOCITY LIMIT
$\bar{\omega}_c$	= COMMANDED ANGULAR VELOCITY
$\hat{\bar{E}}$	= ATTITUDE ERROR
$\bar{\omega}$	= ANGULAR VELOCITY ESTIMATE
ΔT	= CONTROL CYCLE INTERVAL
T_{go}	= CONTROL LAW PARAMETER
P_1, P_2, P_3	= CALCULATION MODIFICATION PARAMETERS
$\bar{\omega}_g$	= ANGULAR VELOCITY-TO-BE-GAINED
ω_{dB}	= ANGULAR VELOCITY DEADBAND

FIGURE 8. AUTOPILOT VELOCITY-TO-BE-GAINED ROUTINE

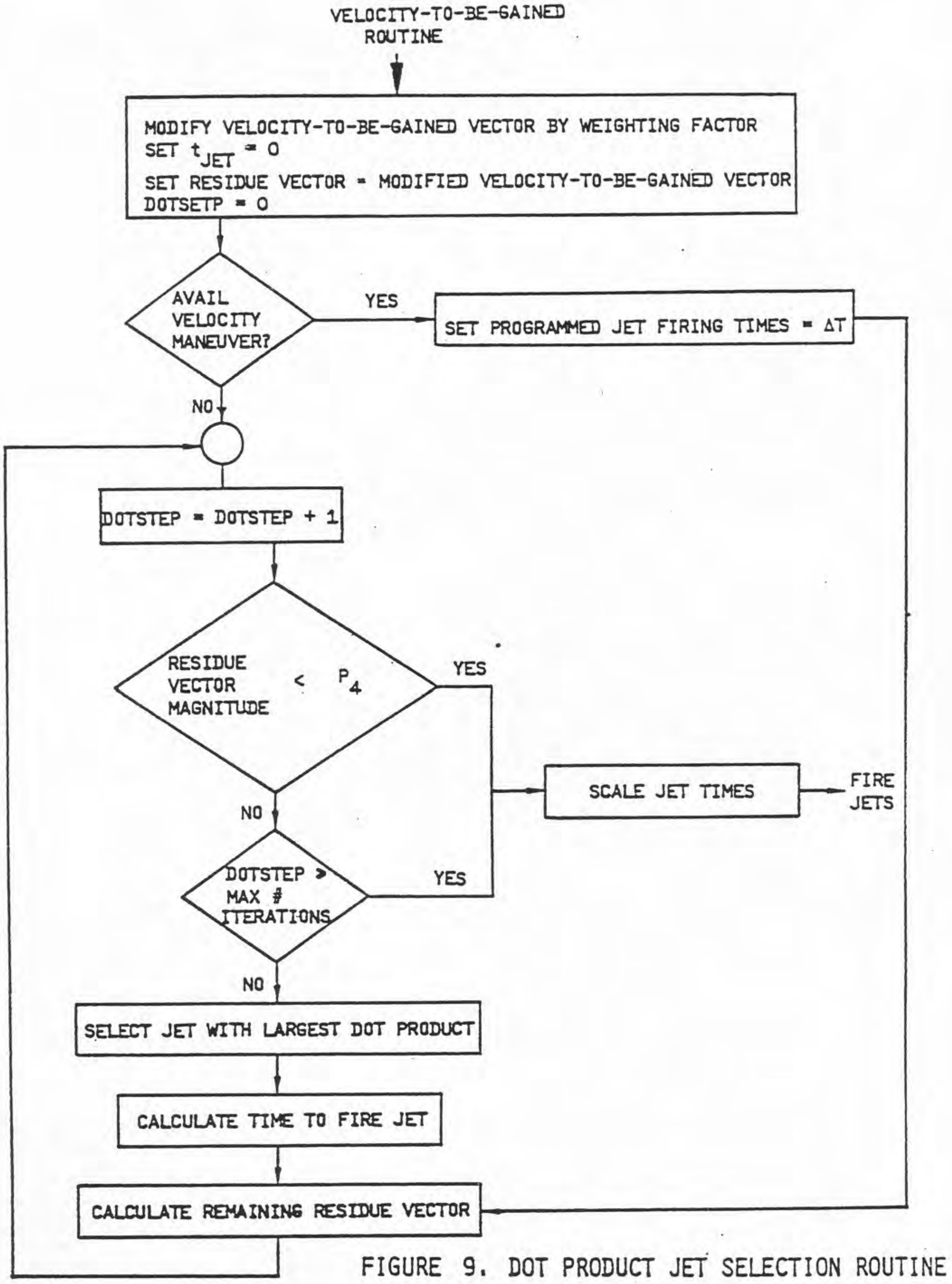
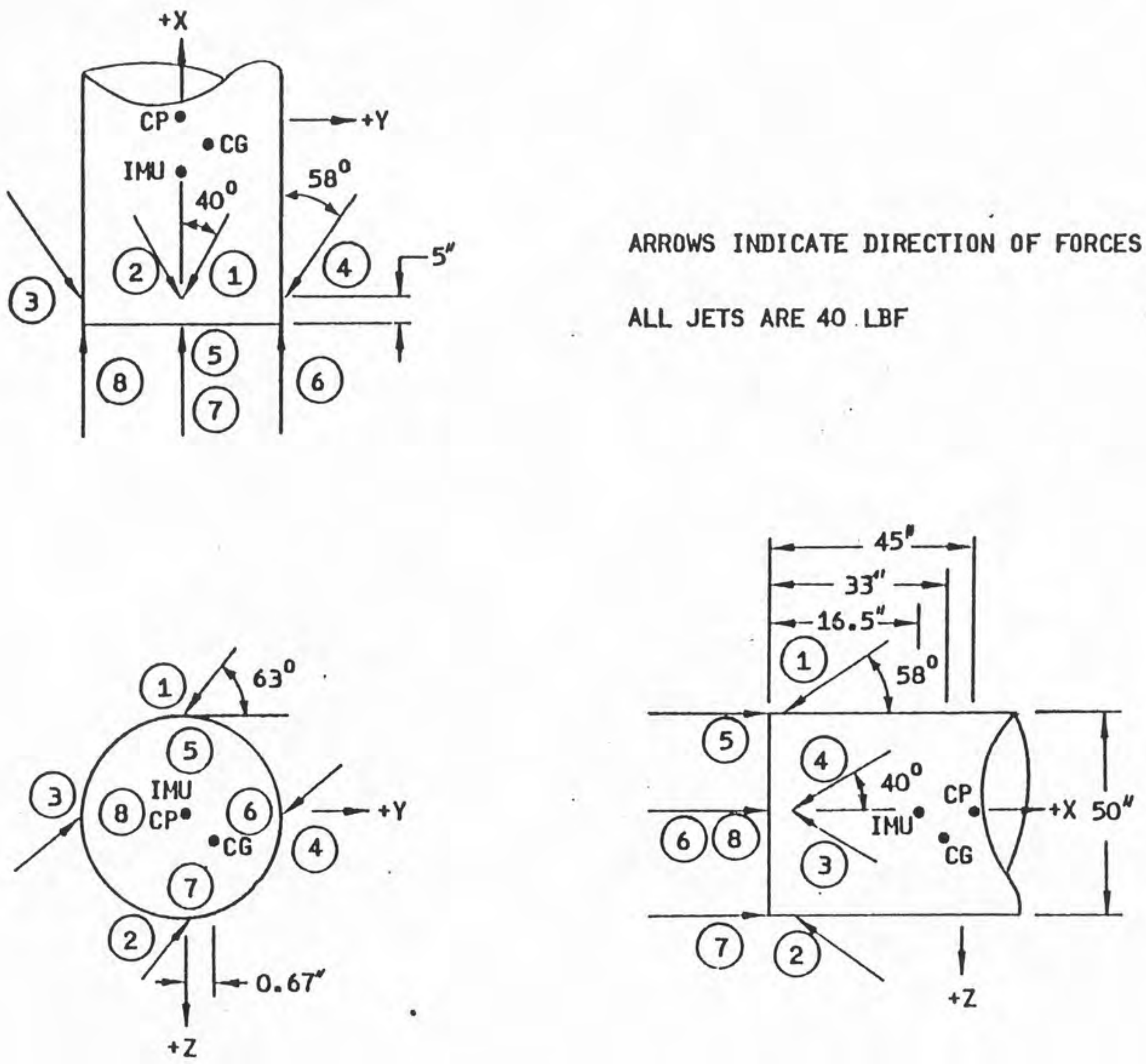

VELOCITY-TO-BE-GAINED ROUTINE
MODIFY VELOCITY-TO-BE-GAINED VECTOR BY WEIGHTING FACTOR
SET t_{JET} = 0
SET RESIDUE VECTOR = MODIFIED VELOCITY-TO-BE-GAINED VECTOR
DOTSETP = 0
AVAIL VELOCITY MANEUVER?
YES
SET PROGRAMMED JET FIRING TIMES = ΔT
NO
DOTSTEP = DOTSTEP + 1
RESIDUE VECTOR MAGNITUDE < P_4
YES
NO
SCALE JET TIMES
FIRE JETS
DOTSTEP > MAX # ITERATIONS
YES
NO
SELECT JET WITH LARGEST DOT PRODUCT
CALCULATE TIME TO FIRE JET
CALCULATE REMAINING RESIDUE VECTOR

FIGURE 9. DOT PRODUCT JET SELECTION ROUTINE



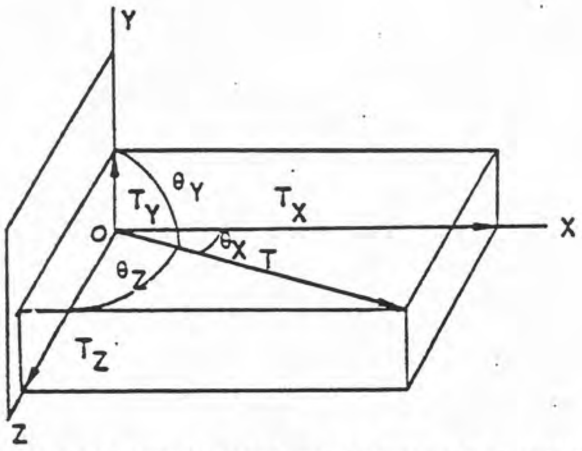
ARROWS INDICATE DIRECTION OF FORCES

ALL JETS ARE 40 LBF

FIGURE 10. PBV CONFIGURATION

SIMULATION MASS PROPERTIES

MASS (SLUGS)	30.0
I_{XX} (SLUG-FT^2)	34.4
I_{YY} " "	131.6
I_{ZZ} " "	139.1
I_{YY} " "	-1.49
I_{XZ} " "	-0.69
I_{YZ} " "	-0.02



DIRECTION ANGLES OF REACTION CONTROL JETS

JET NUMBER	θ_X (DEGREES)	θ_Y (DEGREES)	θ_Z (DEGREES)
1	118.66°	113.58°	38.67°
2	↓	66.42°	141.33°
3	↓	38.67°	113.58°
4	↓	141.33°	66.42°
5	0	90	90
6	↓	↓	↓
7	↓	↓	↓
8	↓	↓	↓

FIGURE 11

MONTE CARLO PERTURBATIONS

MASS DEVIATION	0.53%	(3 σ)
INERTIA DEVIATION	0.53%	$\downarrow$
CG LOCATION DEVIATION	0.25 in	$\downarrow$
JET ORIENTATION DEVIATION	1.0 DEG.	$\downarrow$
LINEAR VELOCITY MEASUREMENT NOISE	0.005 FPS	$\downarrow$
ATTITUDE MEASUREMENT NOISE	0.05%	$\downarrow$
JET TO JET THRUST DEVIATION	5.0%	$\downarrow$
PULSE TO PULSE THRUST DEVIATION	10.0%	$\downarrow$

FIGURE 12

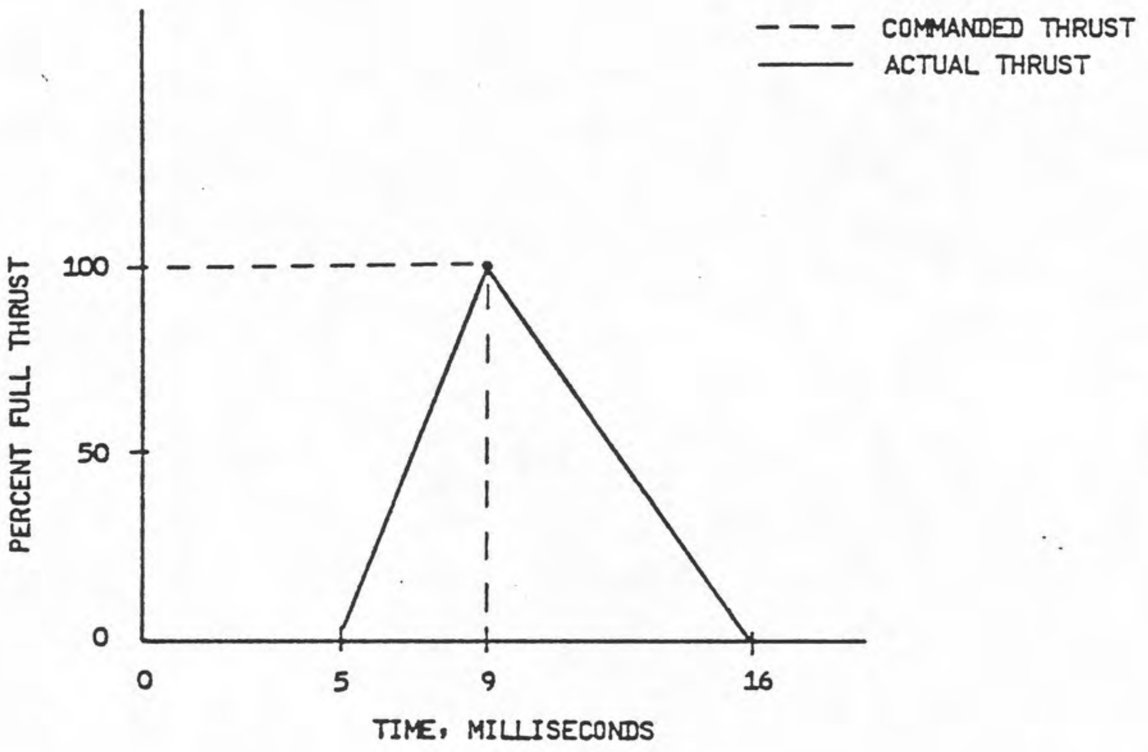


FIGURE 13. REACTION CONTROL JET TIME HISTORY

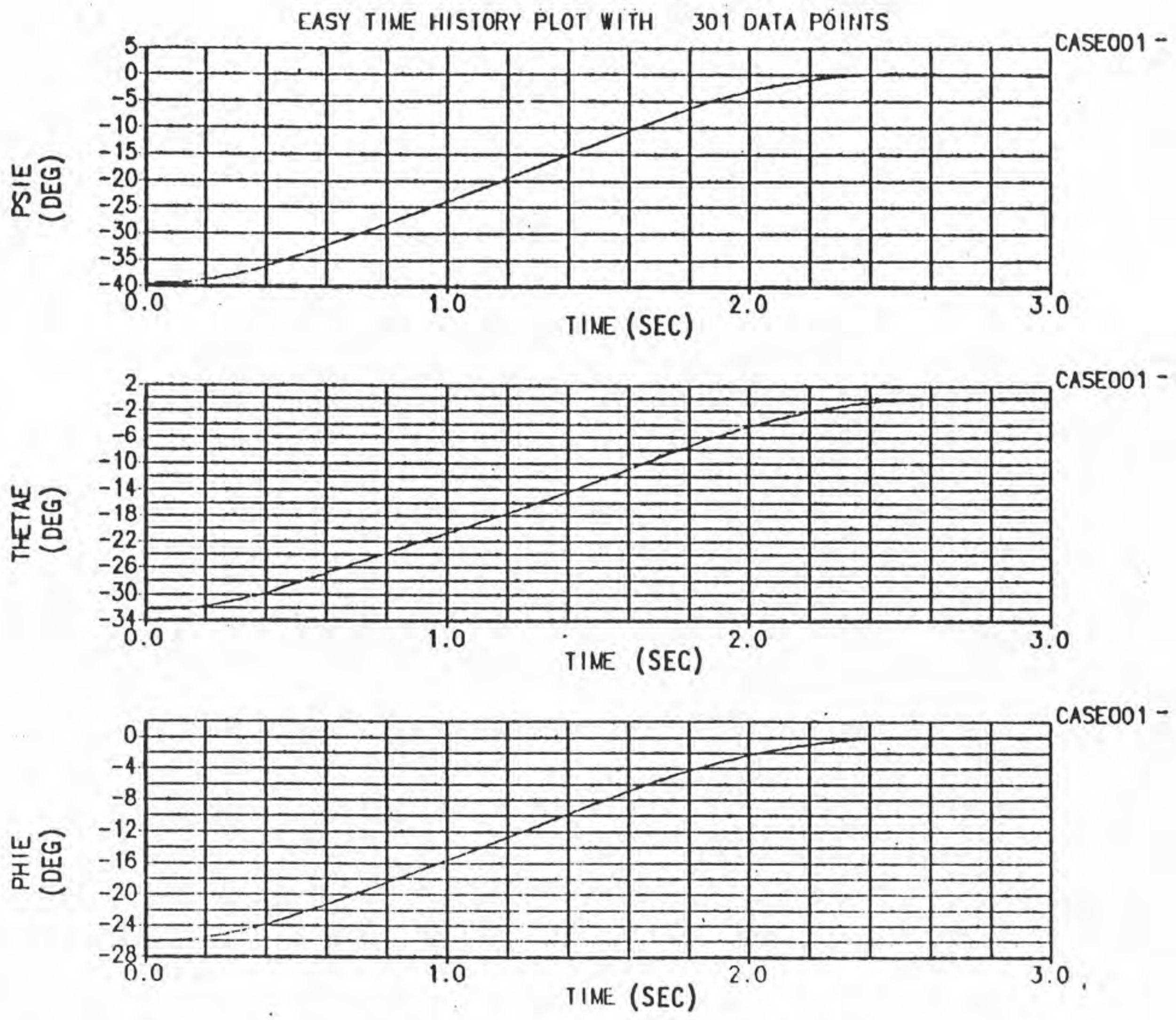


FIGURE 14 ATTITUDE ERROR FOR LARGE ATTITUDE MANEUVER

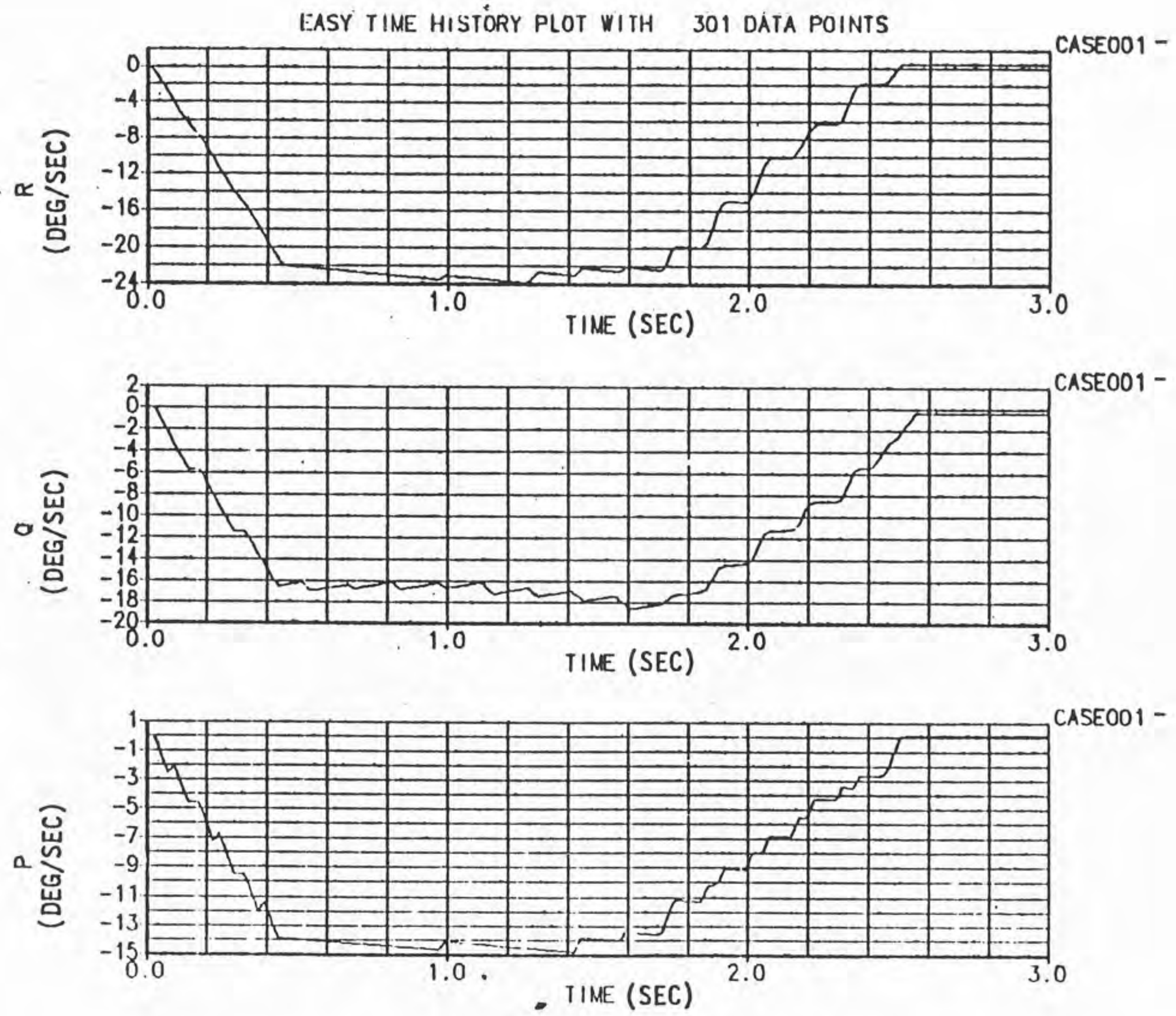


FIGURE 15 ANGULAR VELOCITY FOR LARGE ATTITUDE MANEUVER

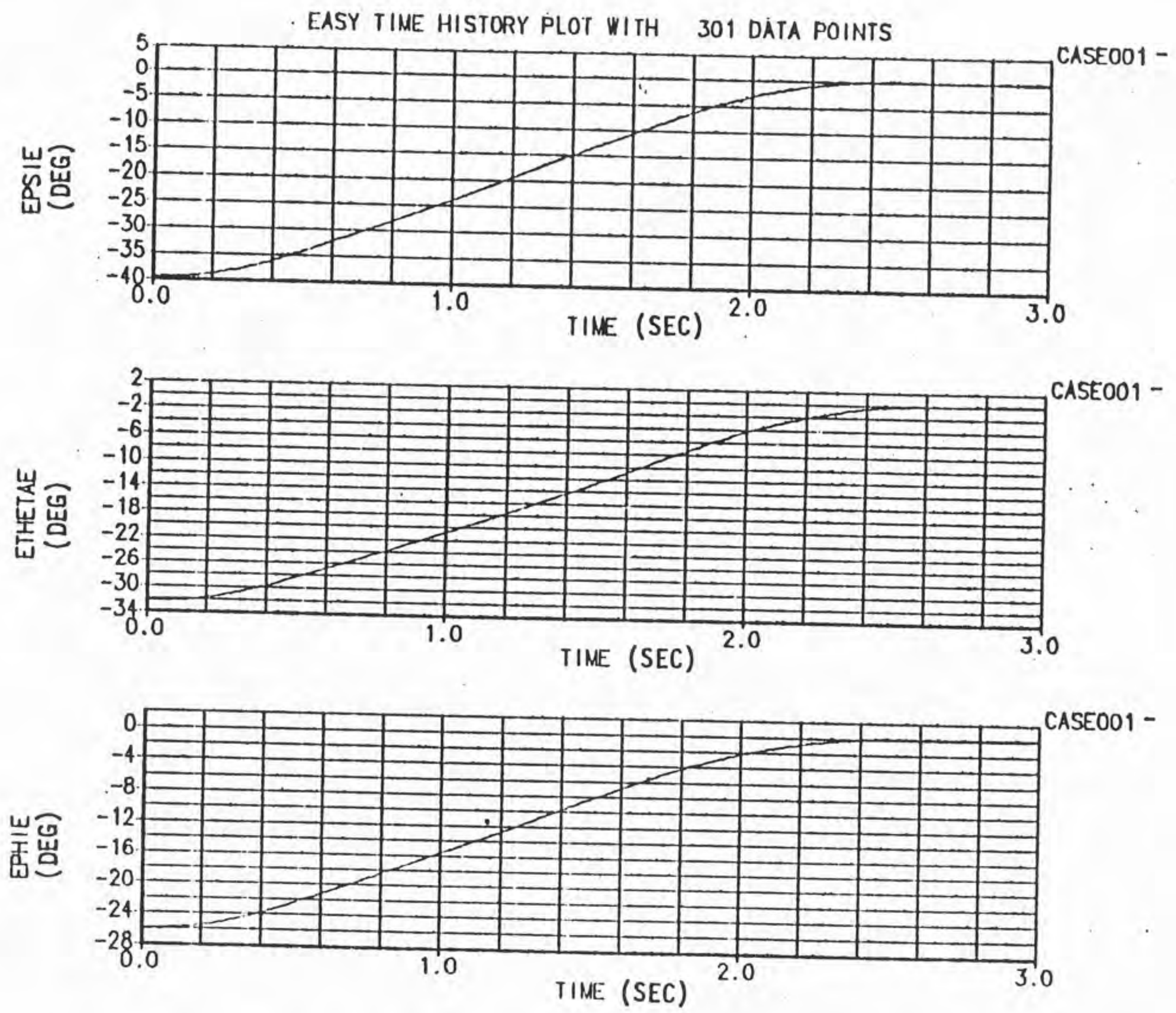


FIGURE 16 ESTIMATED ATTITUDE ERROR FOR LARGE ATTITUDE MANEUVER

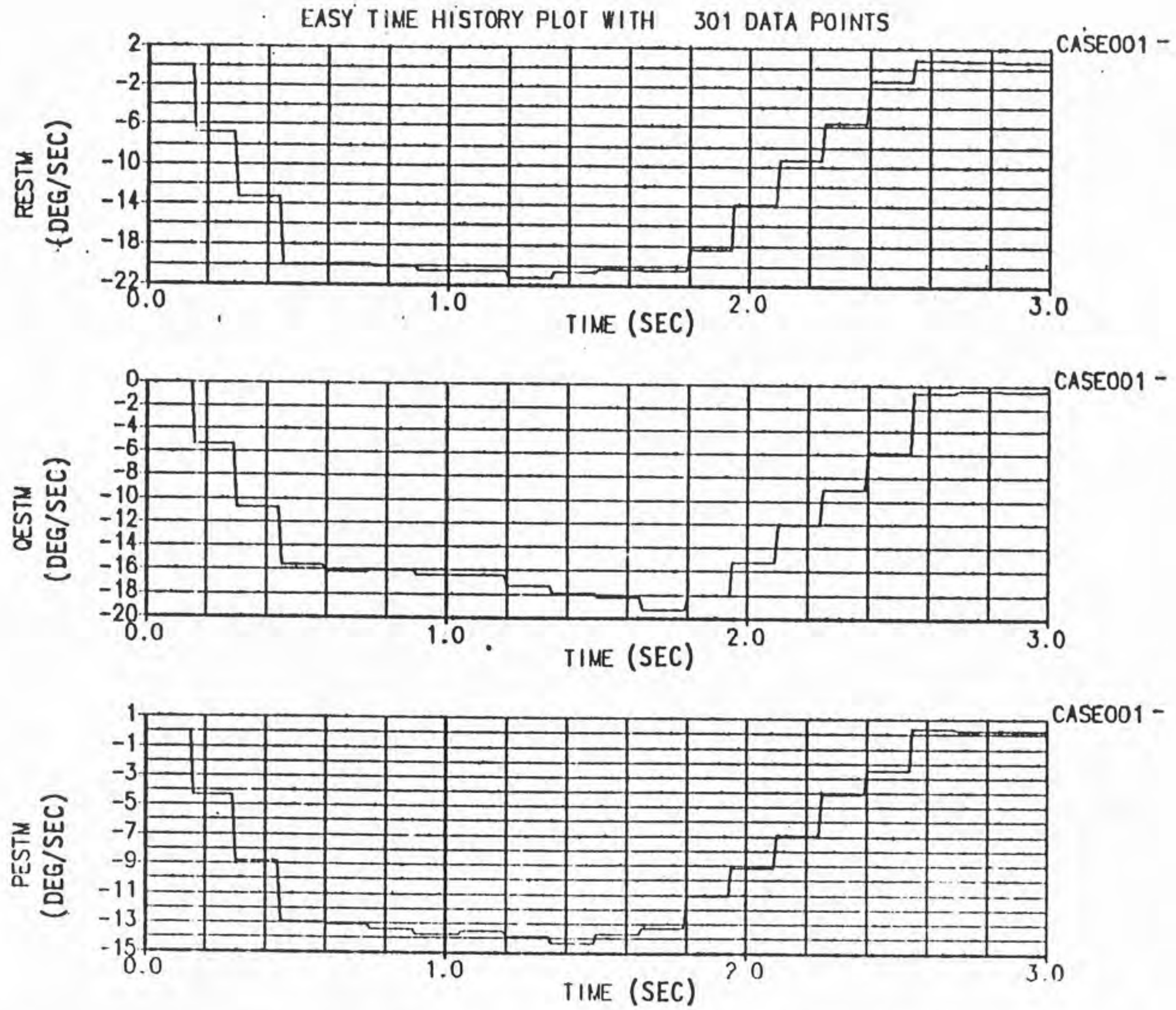


FIGURE 17 ESTIMATED ANGULAR VELOCITY FOR LARGE ATTITUDE MANEUVER

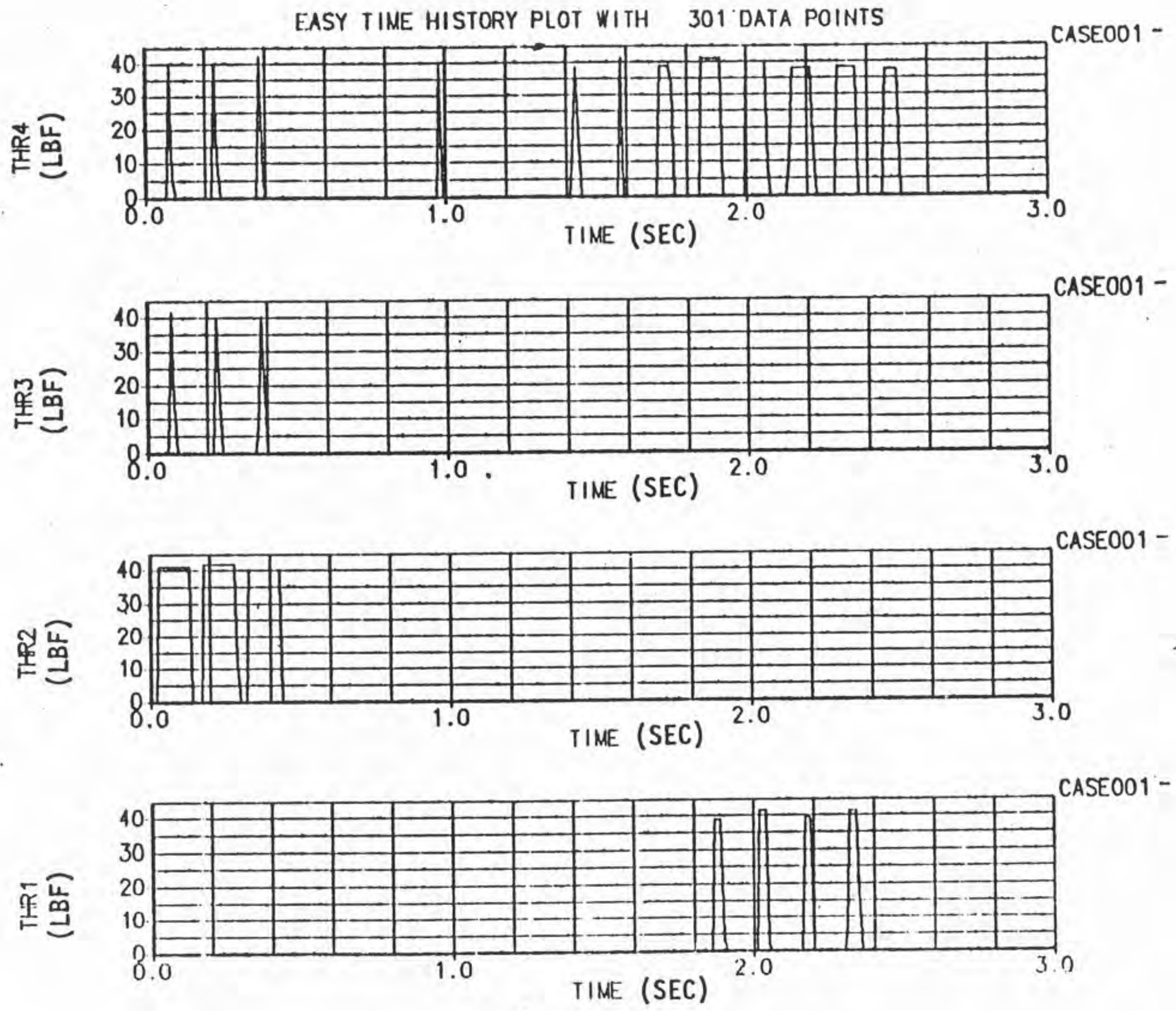


FIGURE 18 JET THRUST HISTORY FOR LARGE ATTITUDE MANEUVER

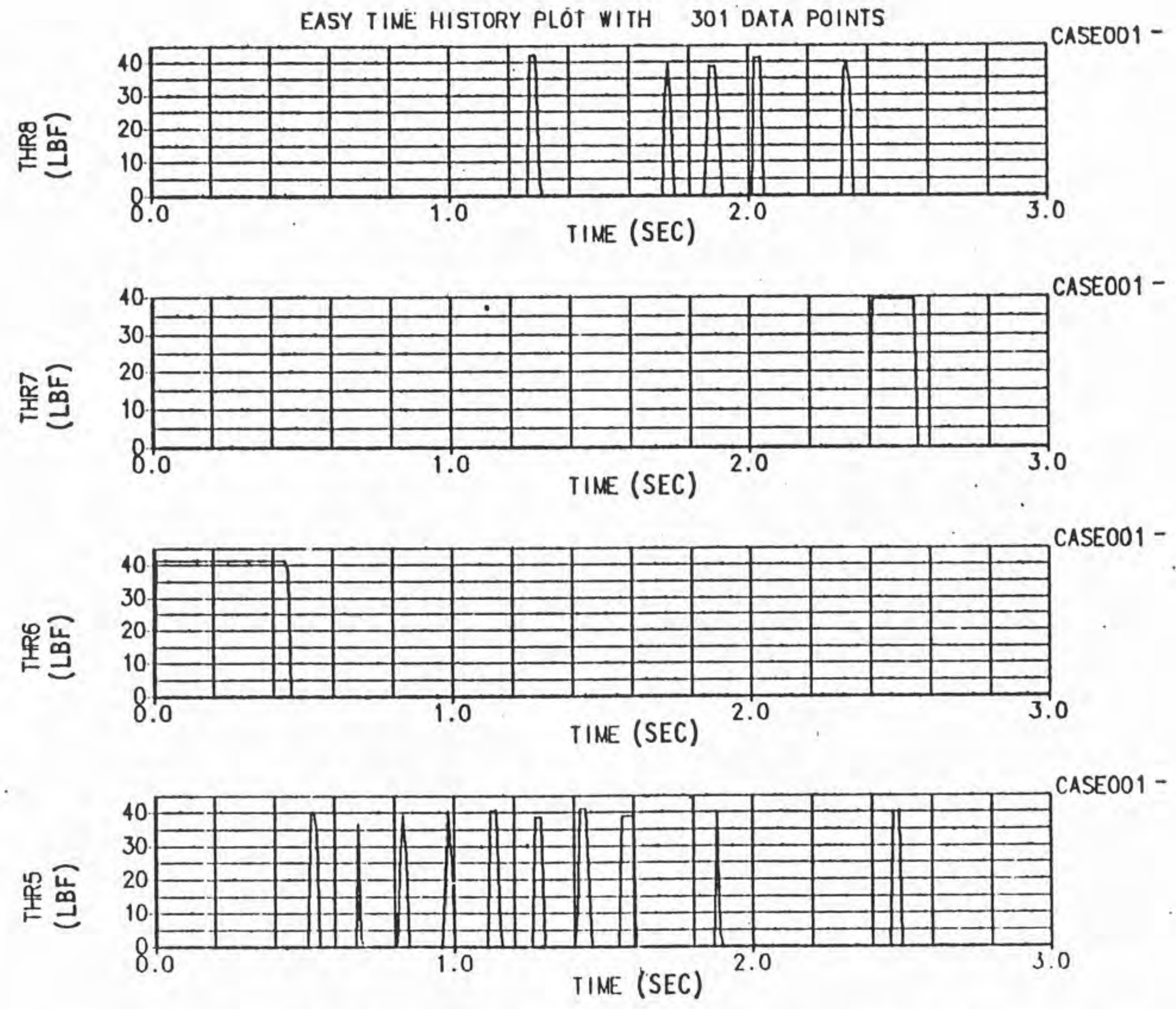


FIGURE 19 JET THRUST HISTORY FOR LARGE ATTITUDE MANEUVER

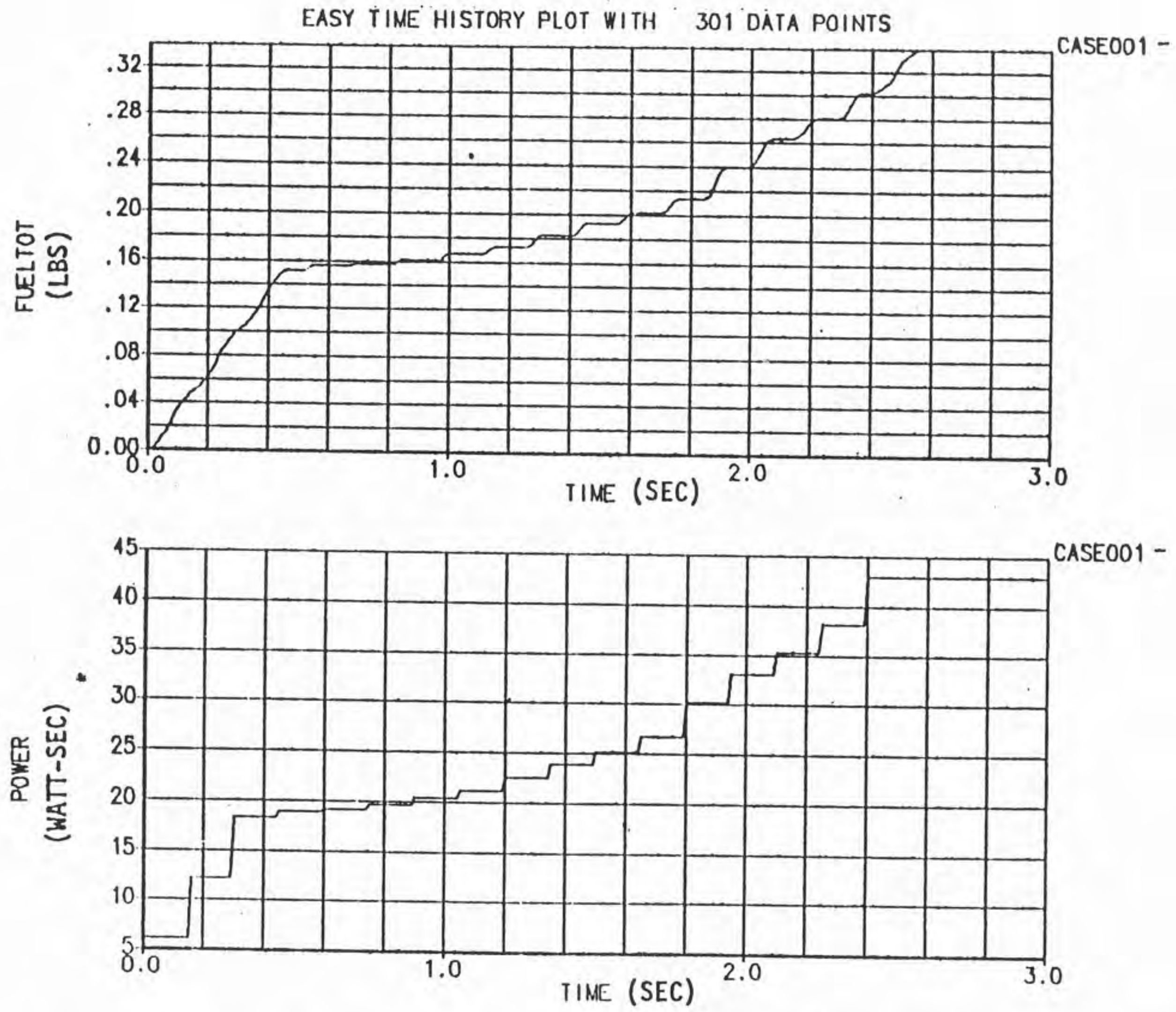


FIGURE 20 POWER AND FUEL CONSUMPTION FOR LARGE ATTITUDE MANEUVER

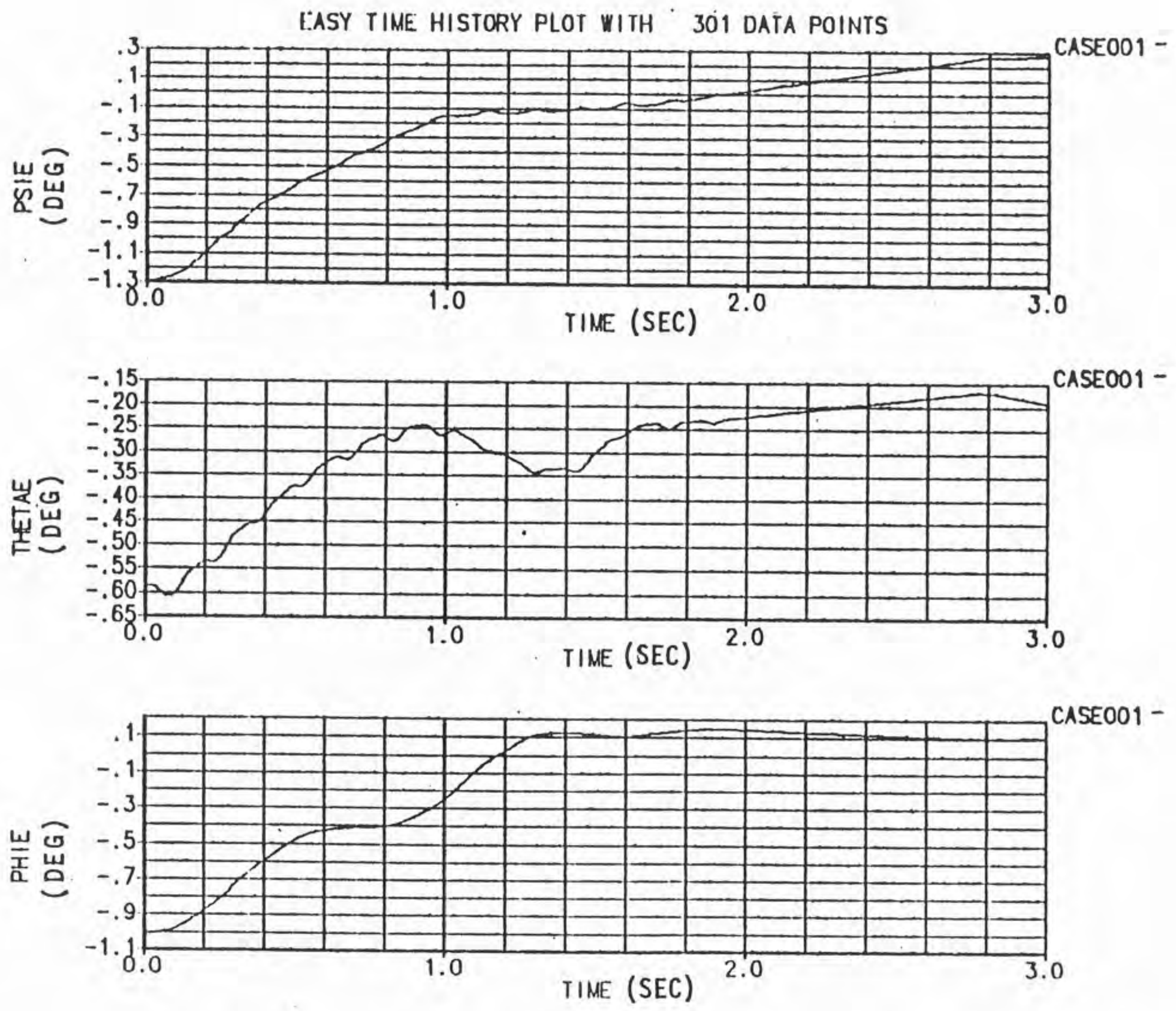


FIGURE 21 ATTITUDE ERROR FOR TERMINAL CONDITION MANEUVER

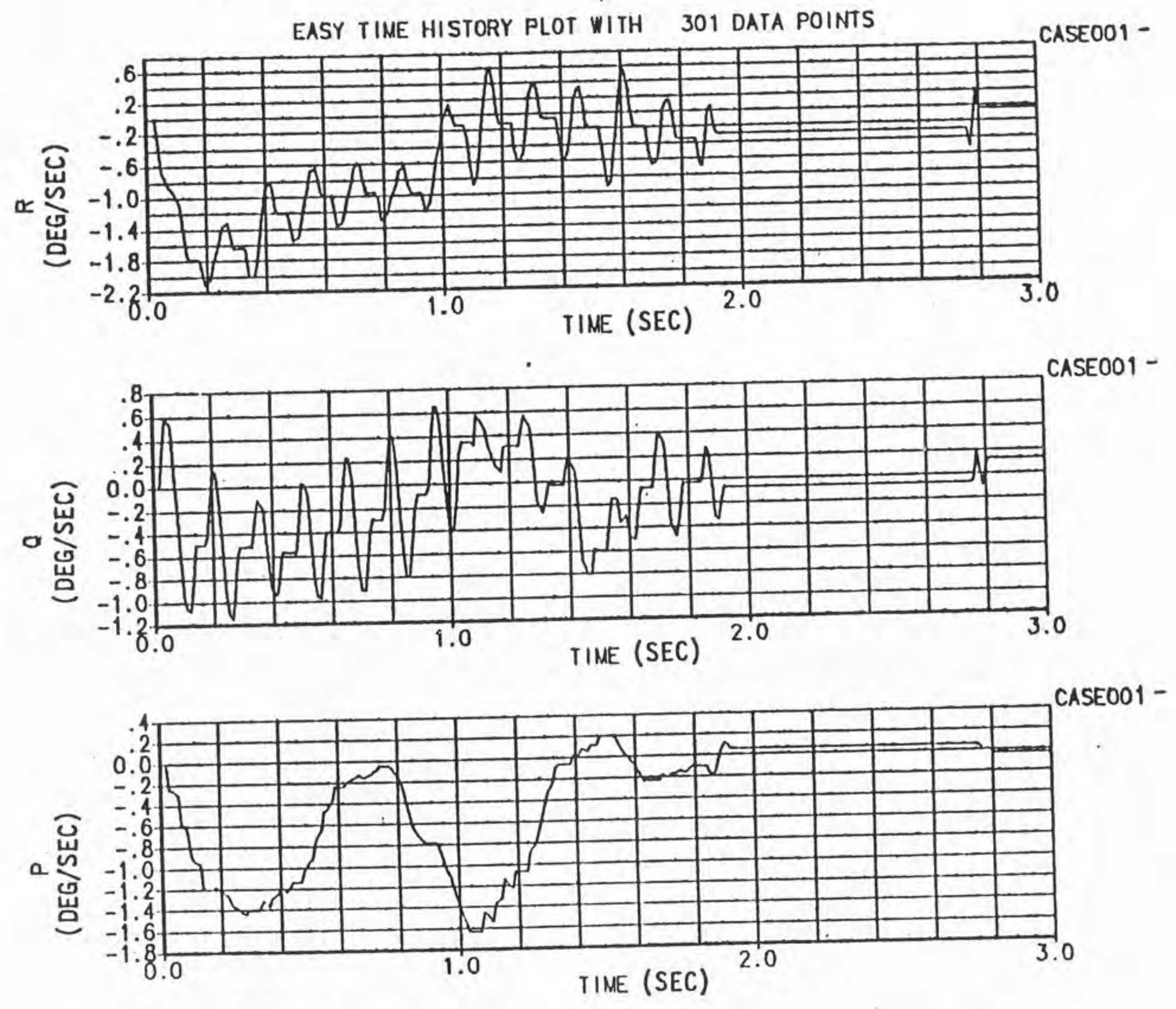


FIGURE 22 ANGULAR VELOCITY FOR TERMINAL CONDITION MANEUVER

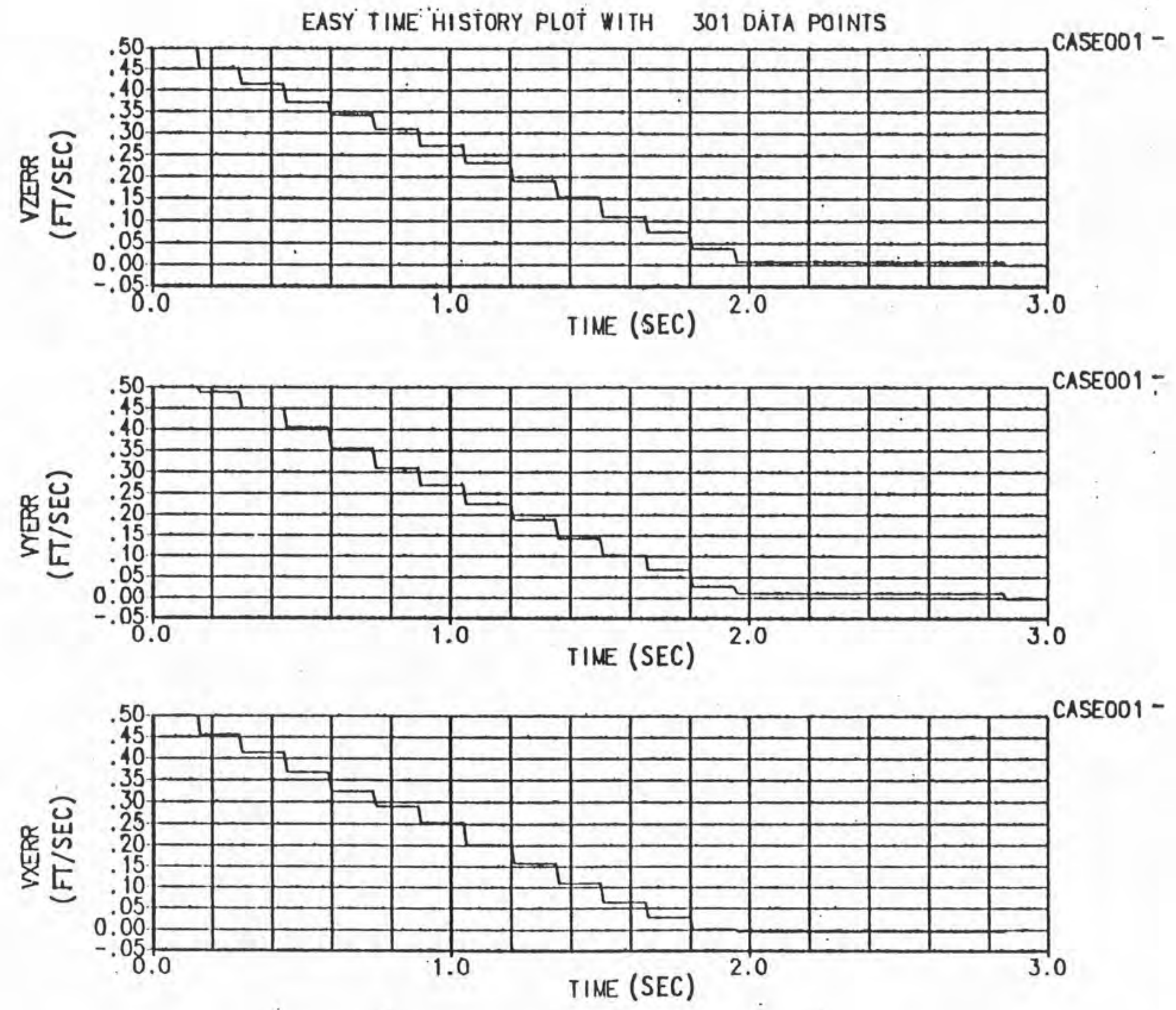


FIGURE 23 LINEAR VELOCITY ERROR FOR TERMINAL CONDITION MANEUVER

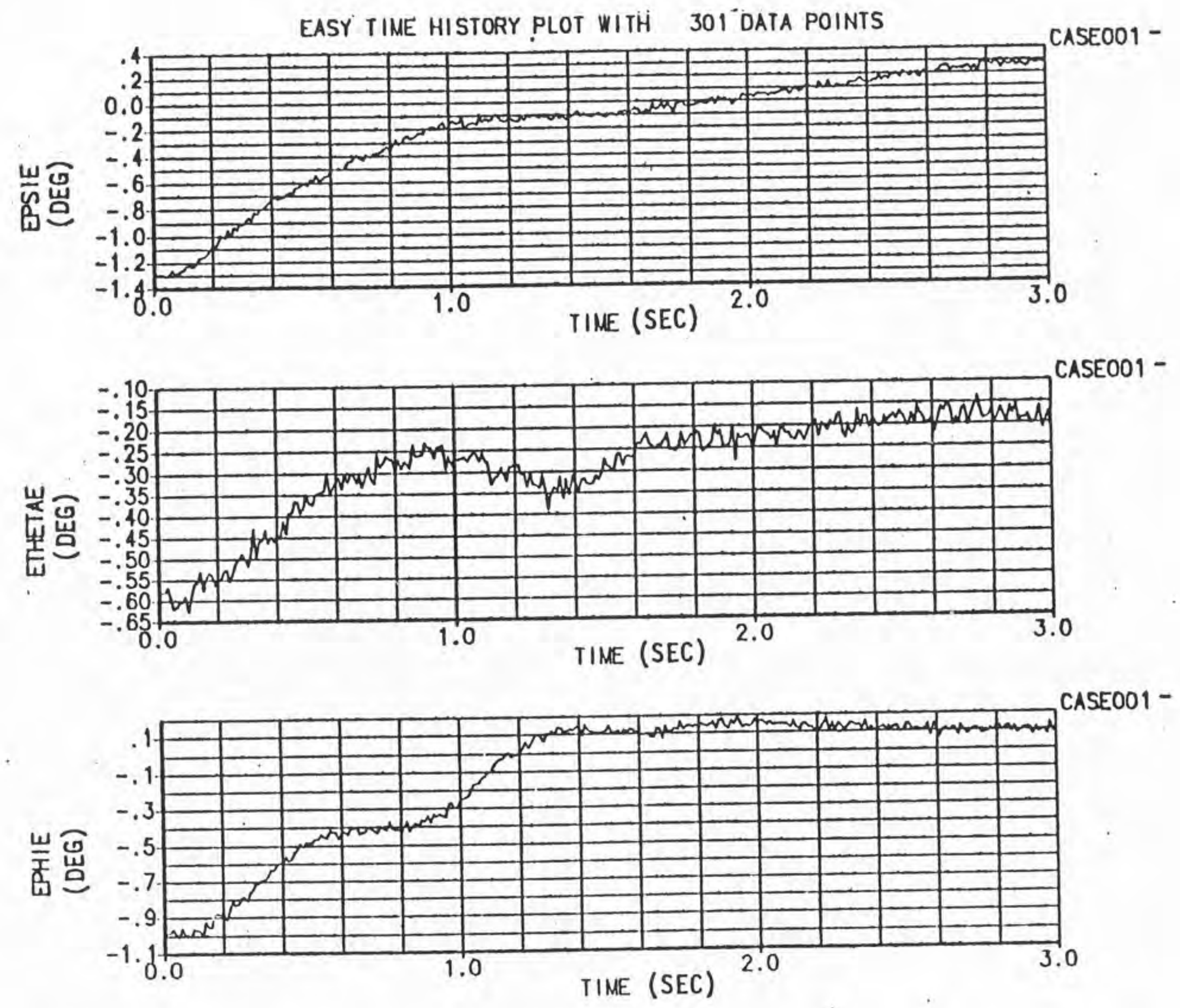


FIGURE 24 ESTIMATED ATTITUDE ERROR FOR TERMINAL CONDITION MANEUVER

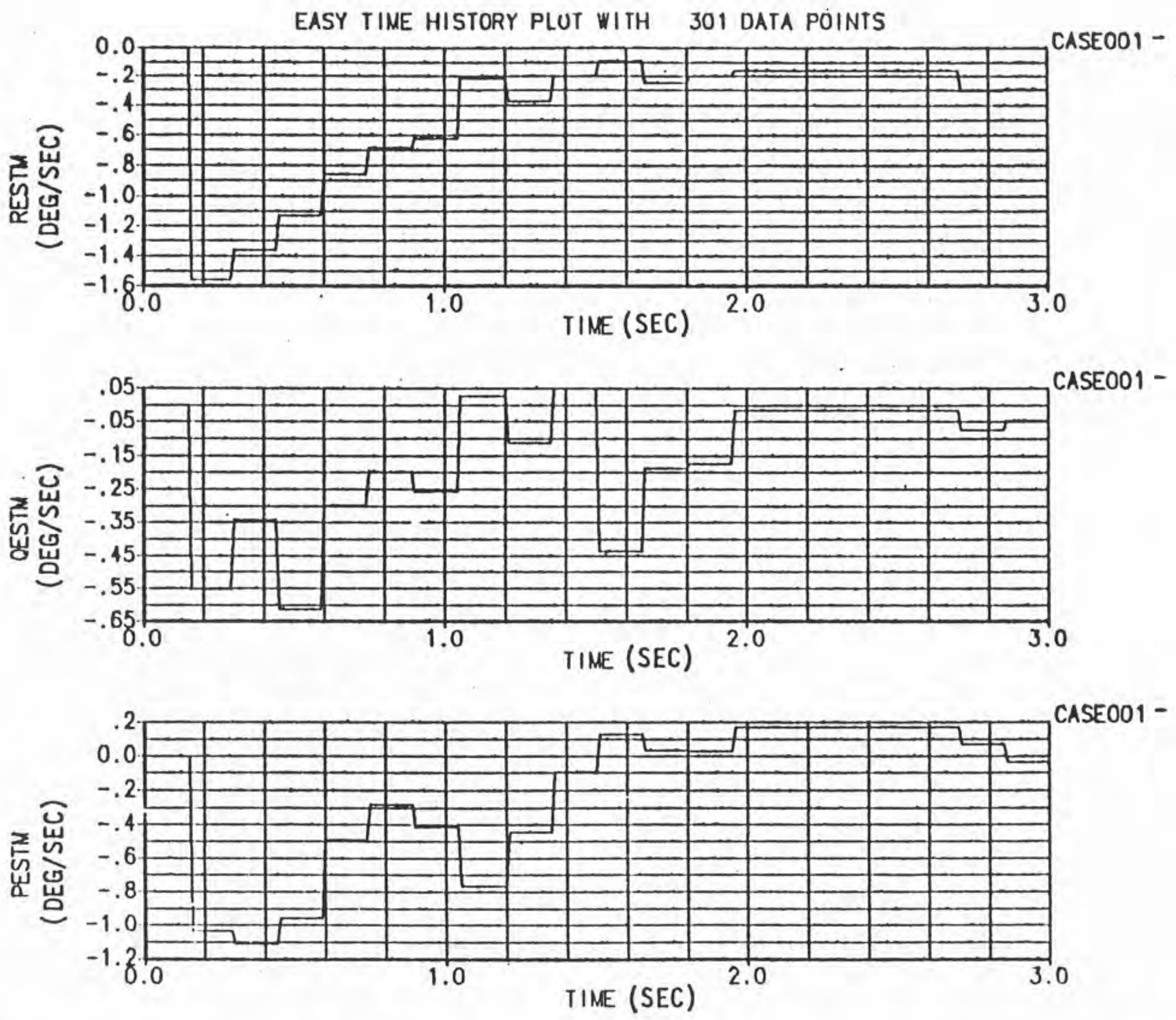


FIGURE 25 ESTIMATED ANGULAR VELOCITY FOR TERMINAL CONDITION MANEUVER

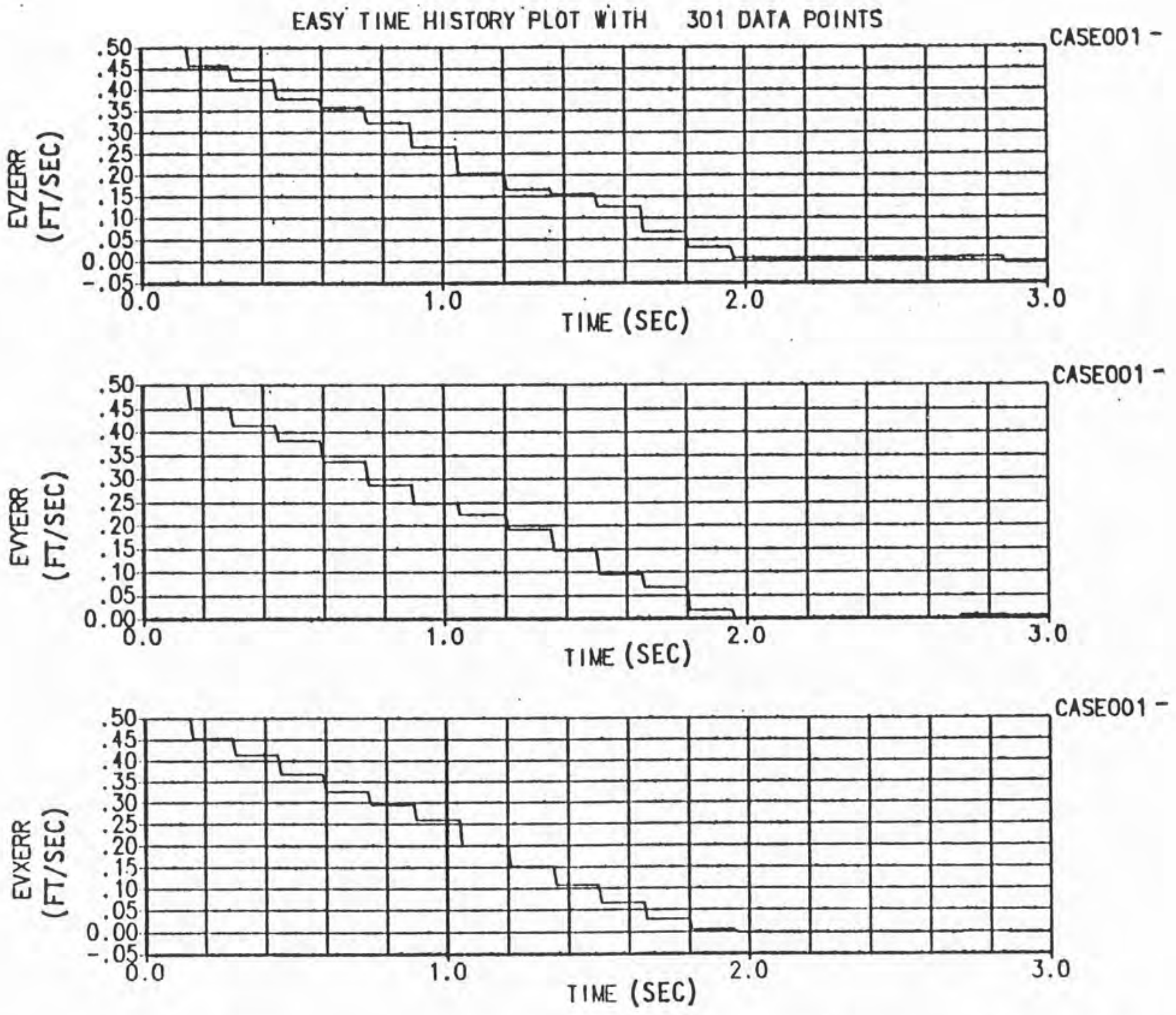


FIGURE 26 ESTIMATED LINEAR VELOCITY ERROR FOR TERMINAL CONDITION MANEUVER

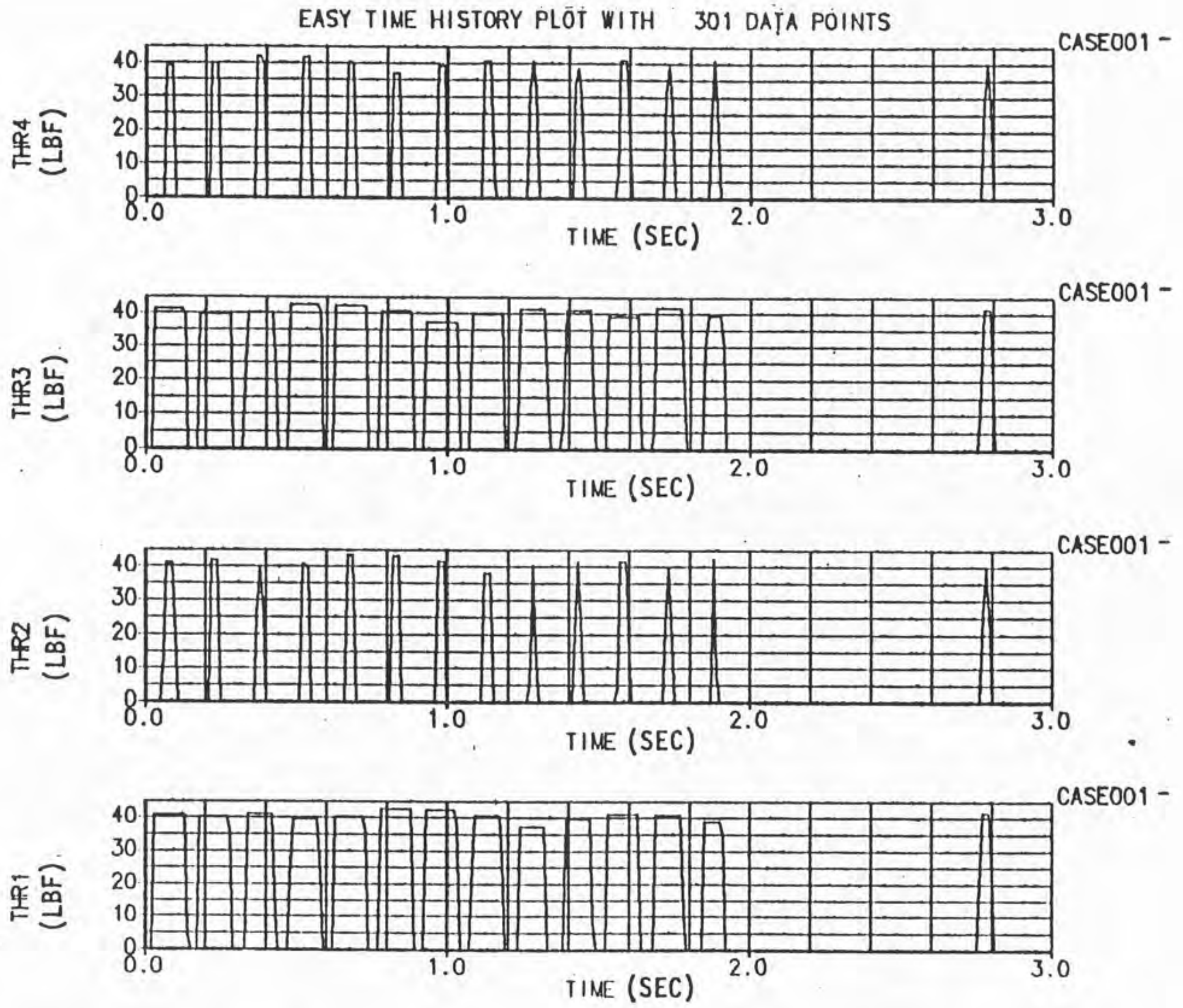


FIGURE 27 JET THRUST HISTORY FOR TERMINAL CONDITION MANEUVER

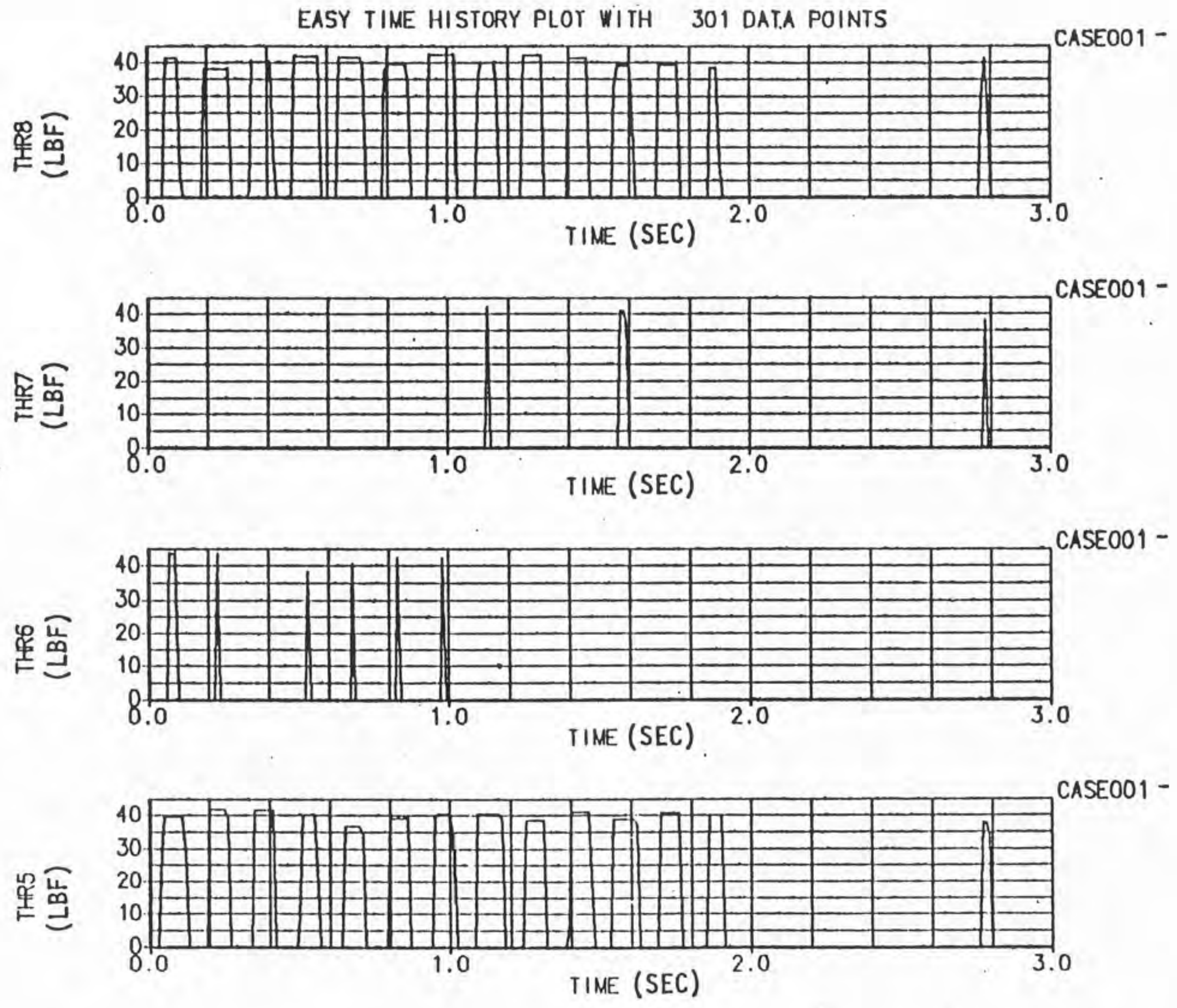


FIGURE 28 JET THRUST HISTORY FOR TERMINAL CONDITION MANEUVER

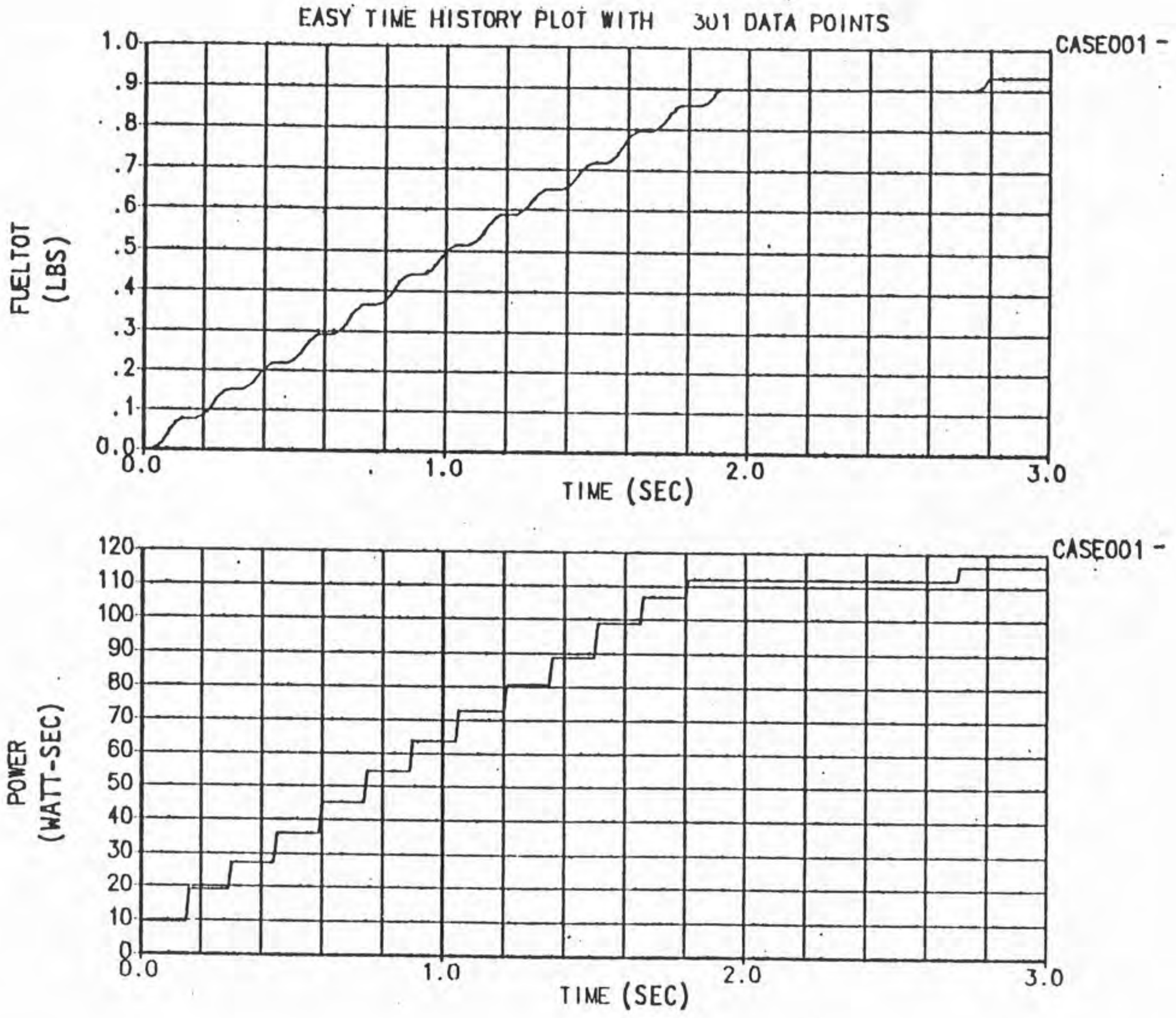


FIGURE 29 POWER AND FUEL CONSUMPTION FOR TERMINAL CONDITION MANEUVER

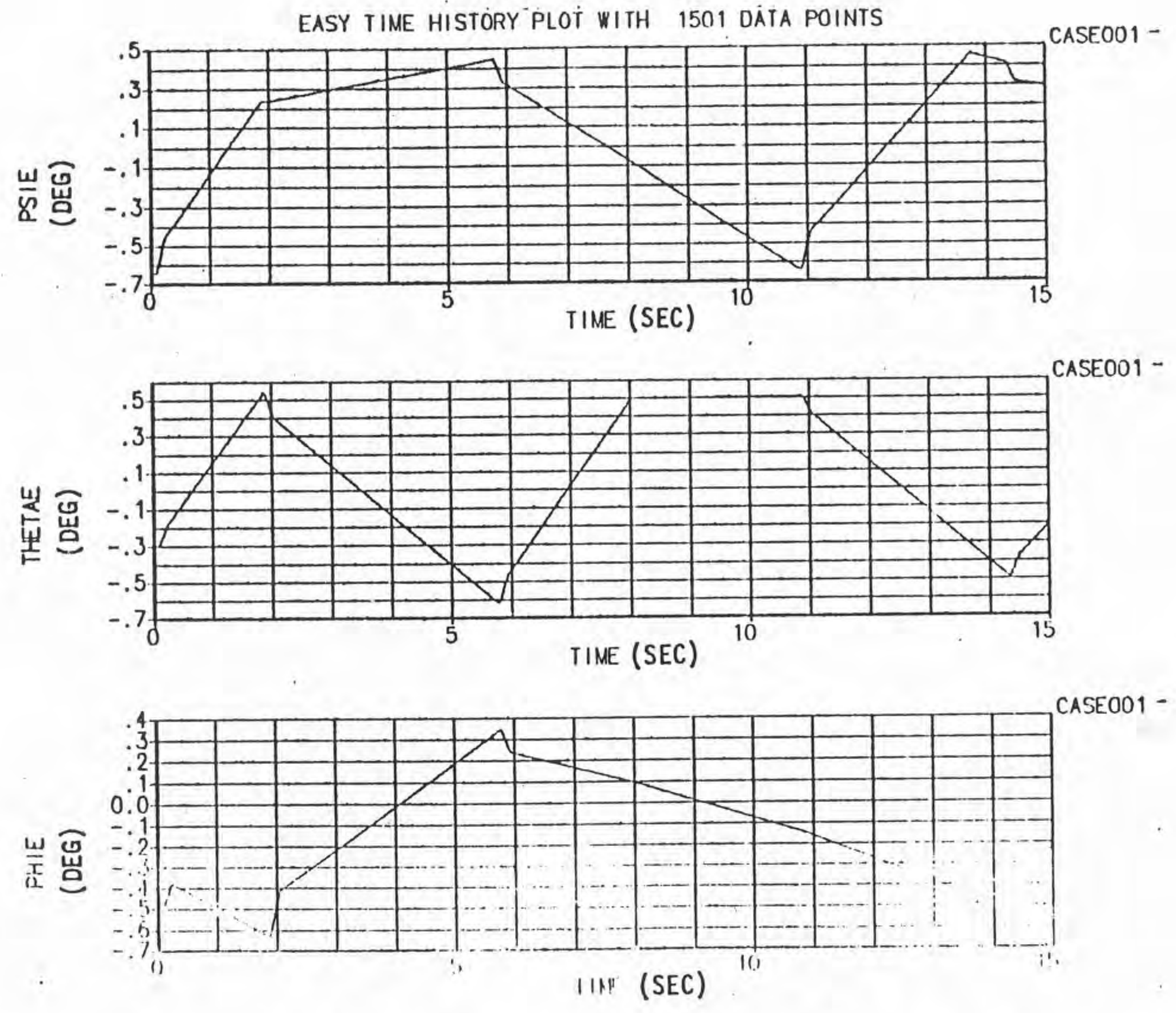


FIGURE 30 ATTITUDE ERROR FOR ATTITUDE HOLD MANEUVER

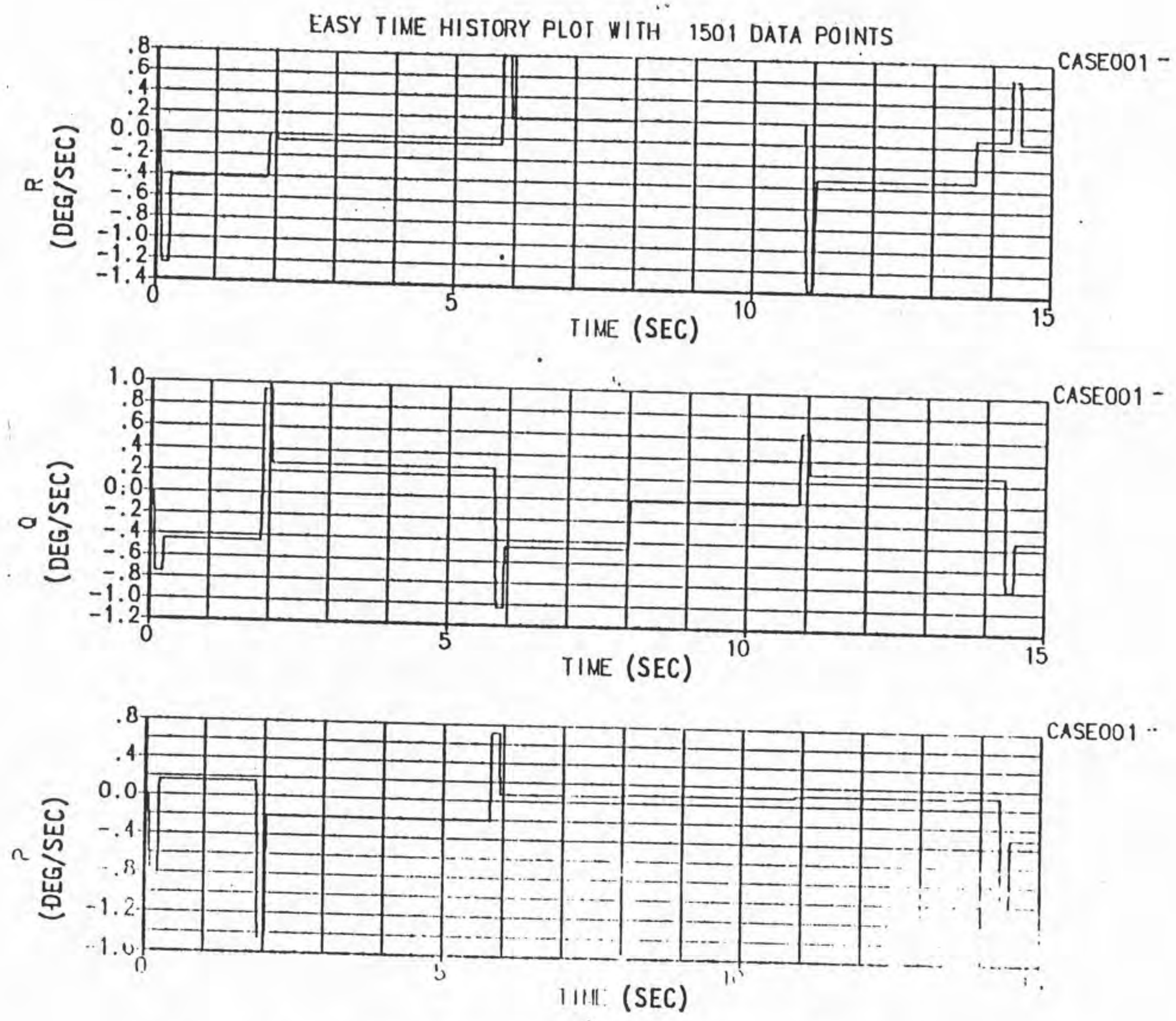


FIGURE 31 ANGULAR VELOCITY FOR ATTITUDE HOLD MANEUVER

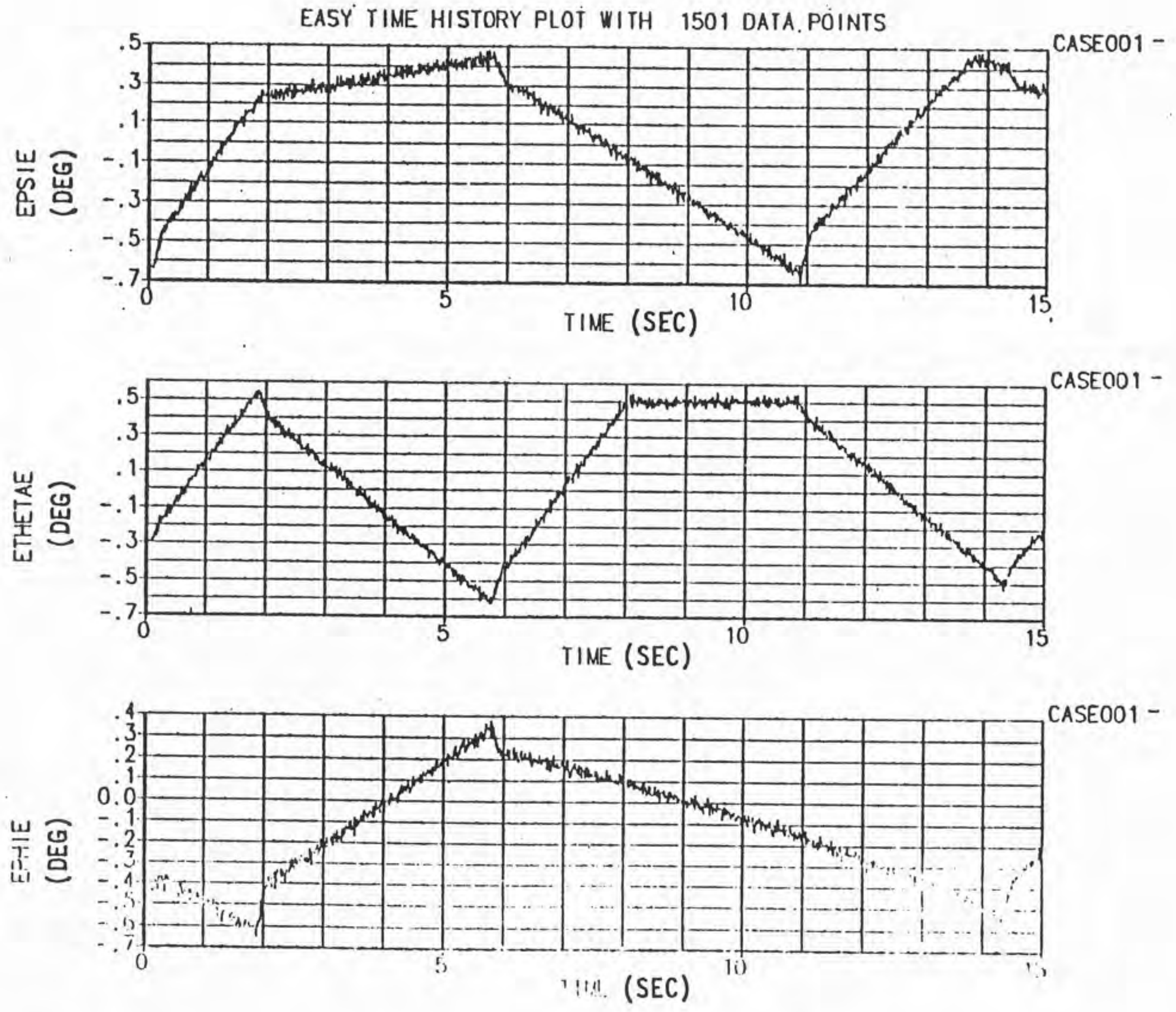


FIGURE 32 ESTIMATED ATTITUDE ERROR FOR ATTITUDE HOLD MANEUVER

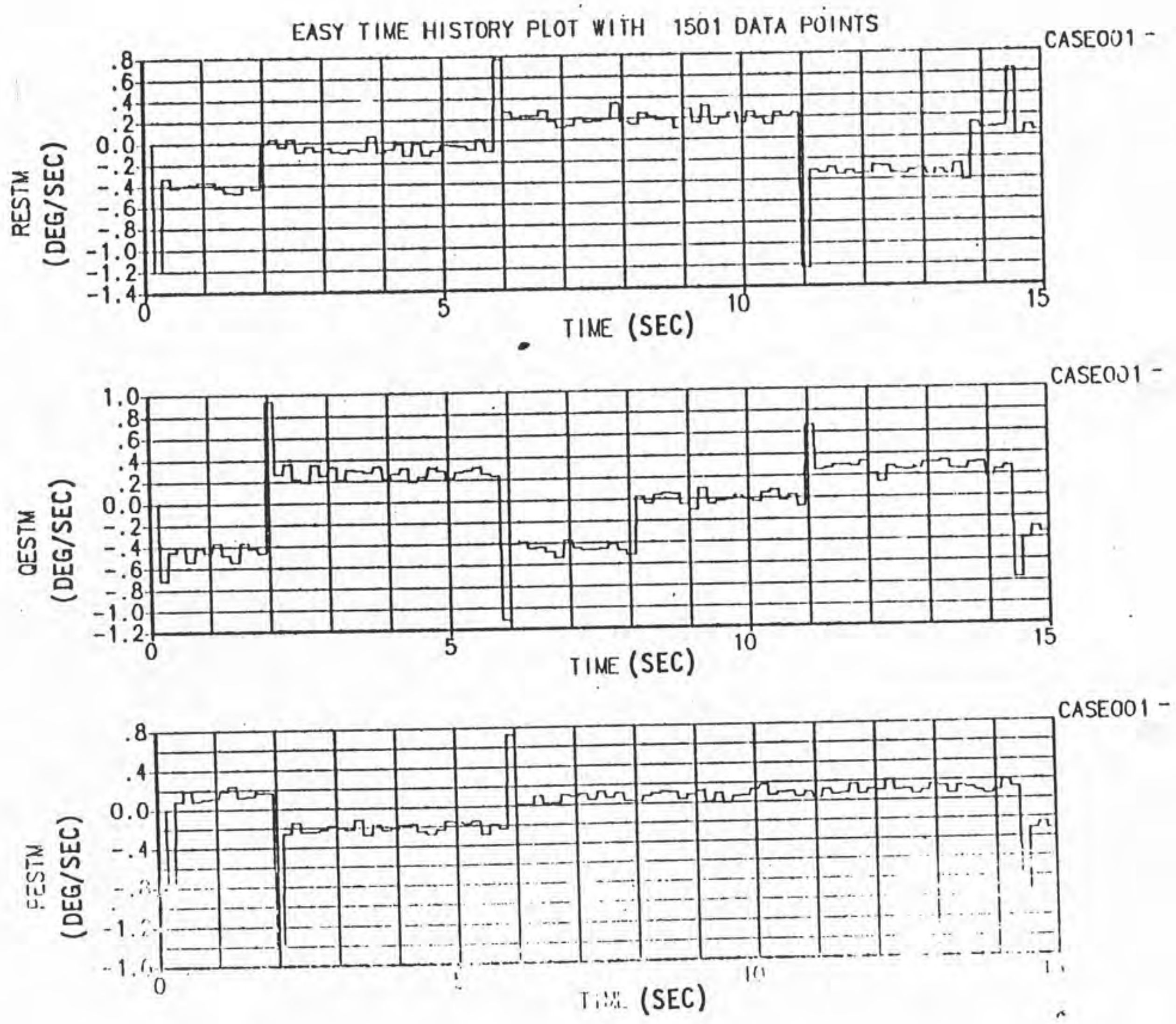


FIGURE 33 ESTIMATED ANGULAR VELOCITY FOR ATTITUDE HOLD MANEUVER

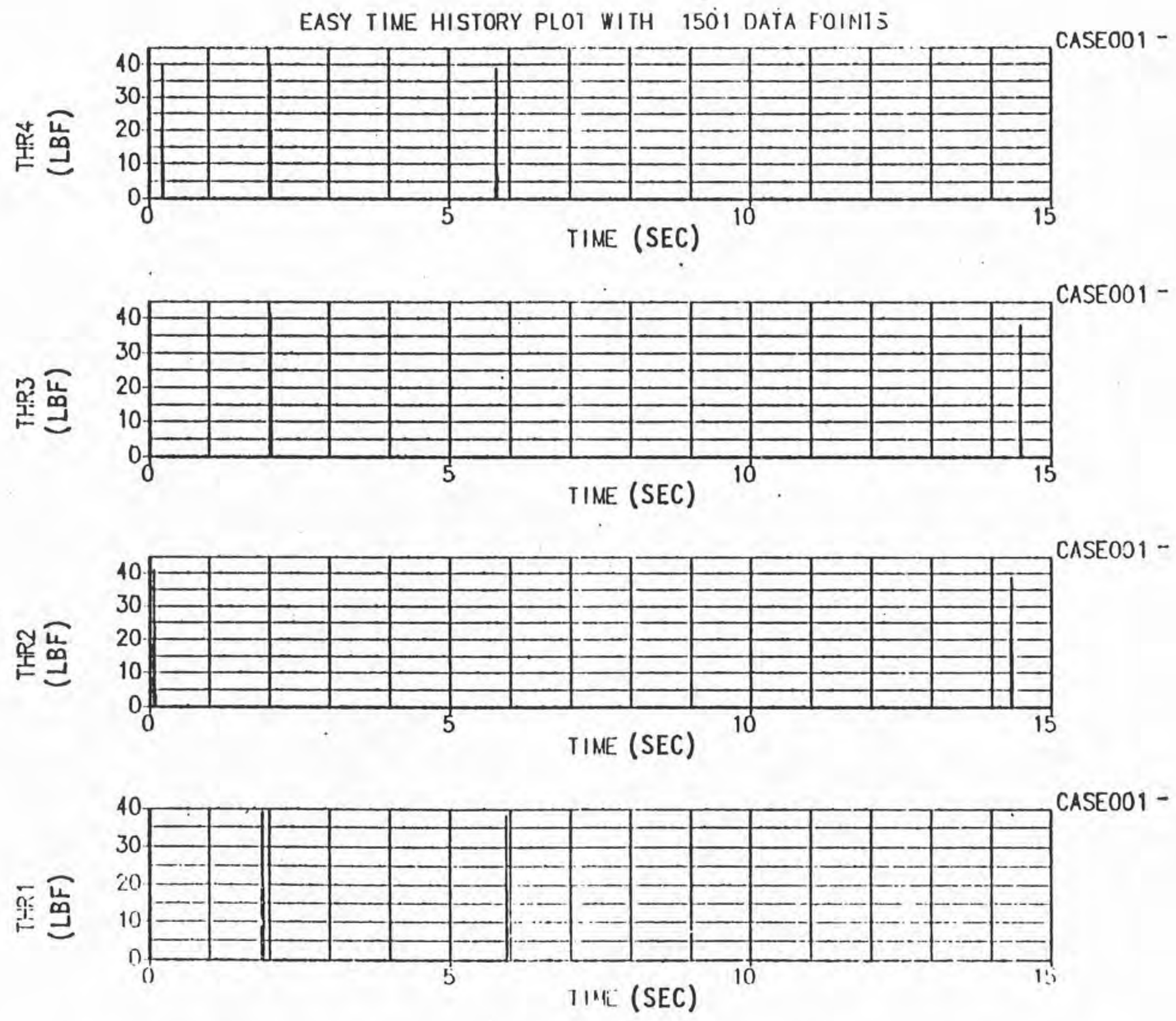


FIGURE 34 JET THRUST HISTORY FOR ATTITUDE HOLD MANEUVER

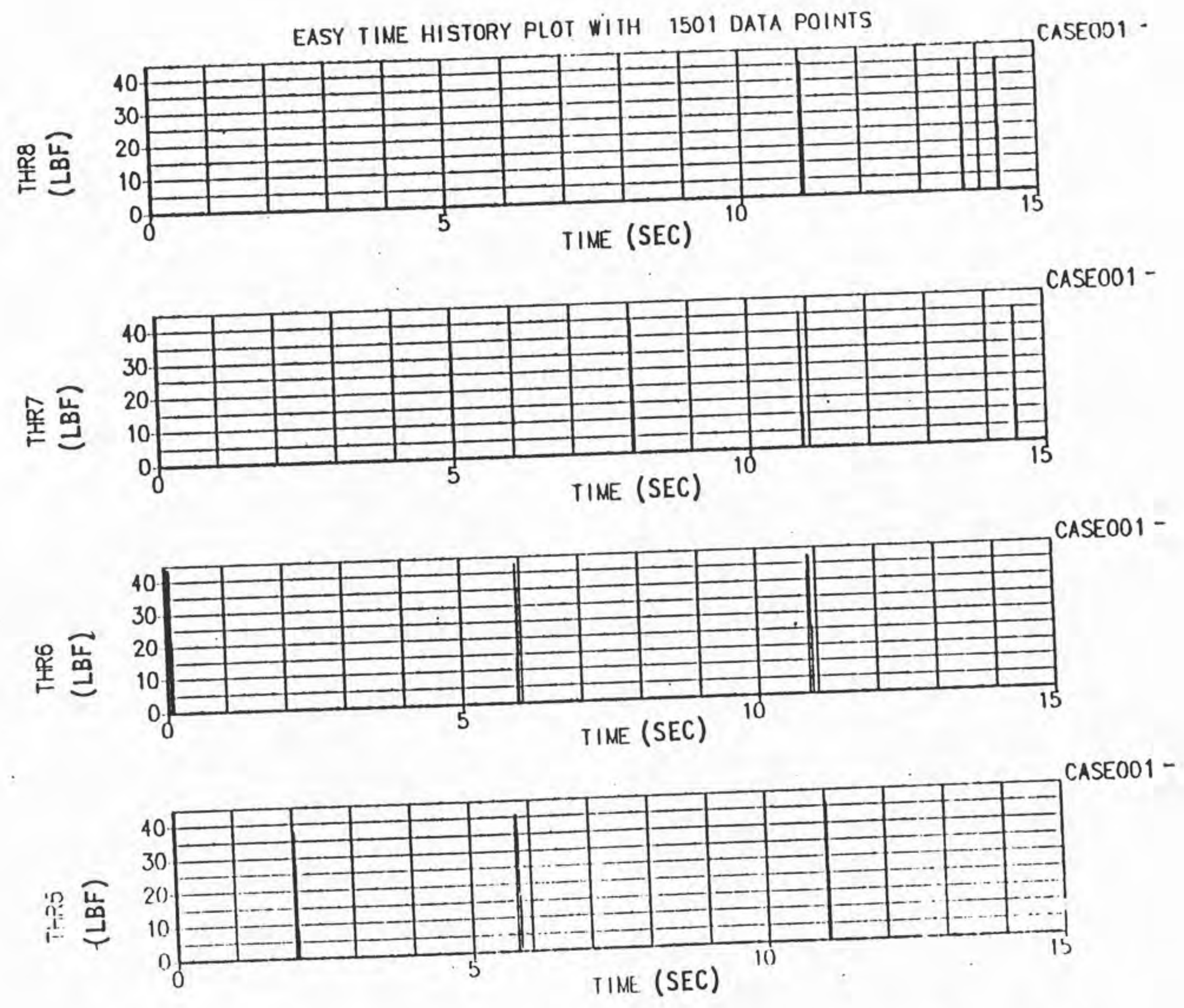


FIGURE 35 JET THRUST HISTORY FOR ATTITUDE HOLD MANEUVER

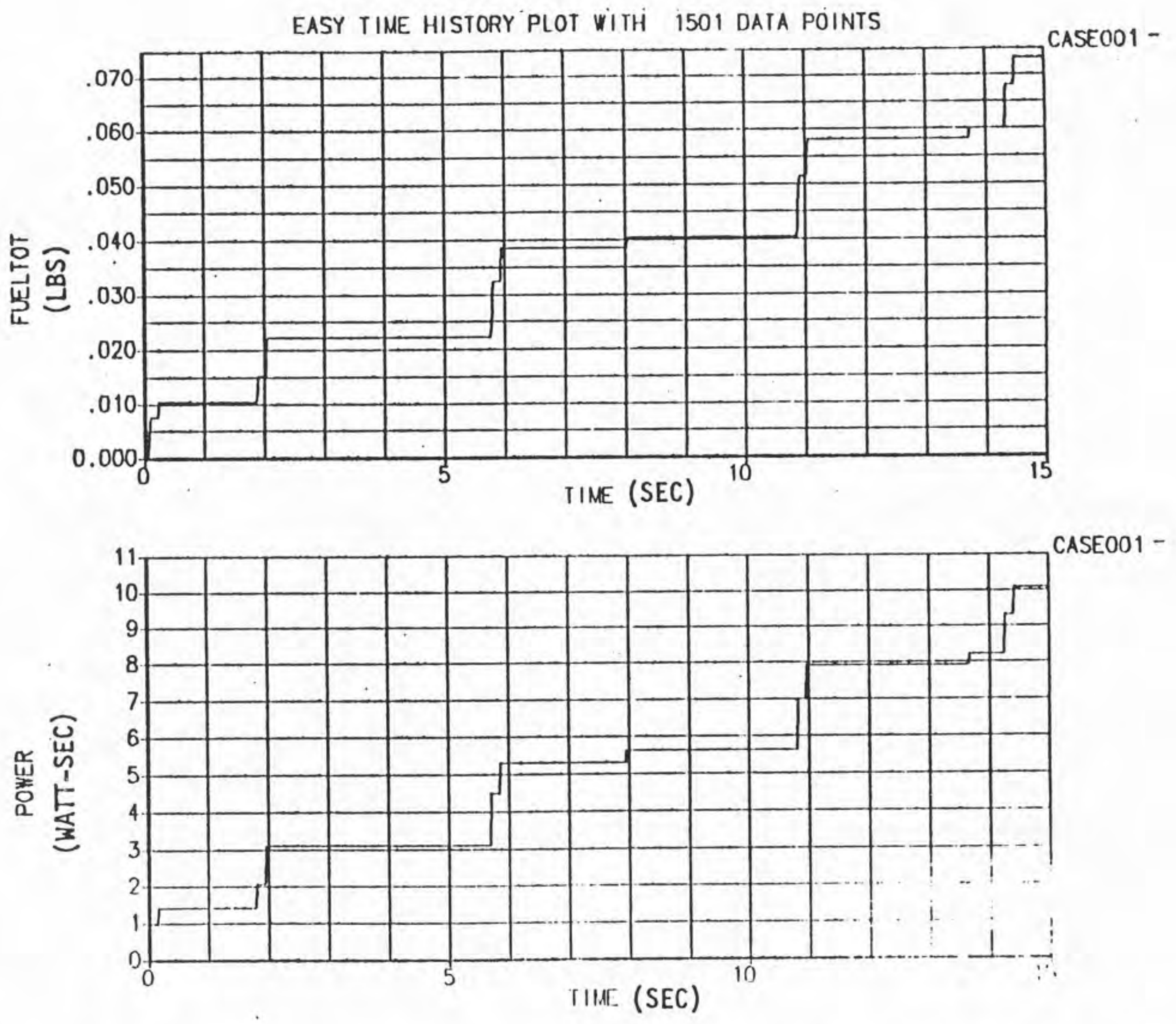


FIGURE 36 POWER AND FUEL CONSUMPTION FOR ATTITUDE HOLD MANEUVER

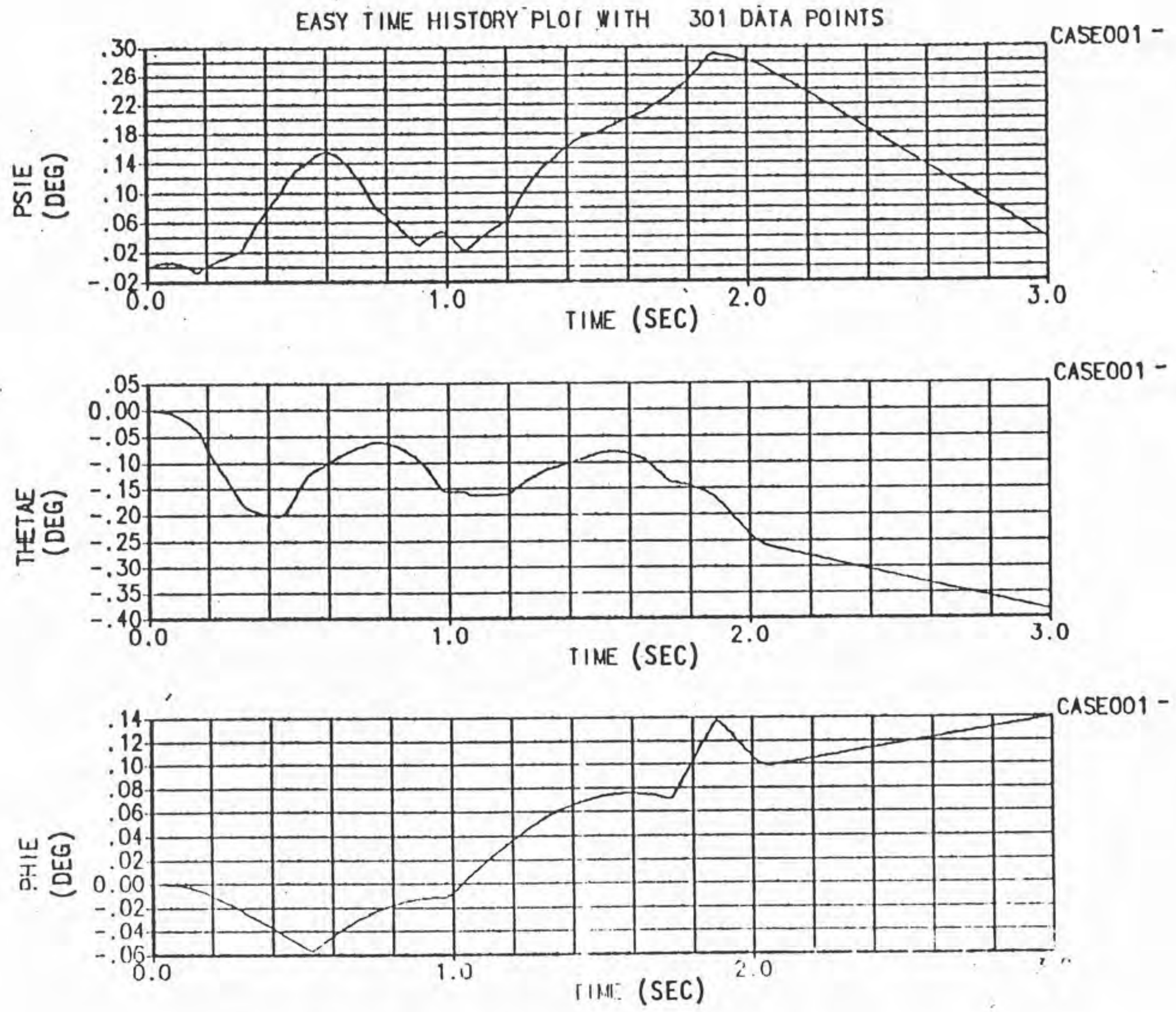


FIGURE 37 ATTITUDE ERROR FOR AXIAL VELOCITY MANEUVER

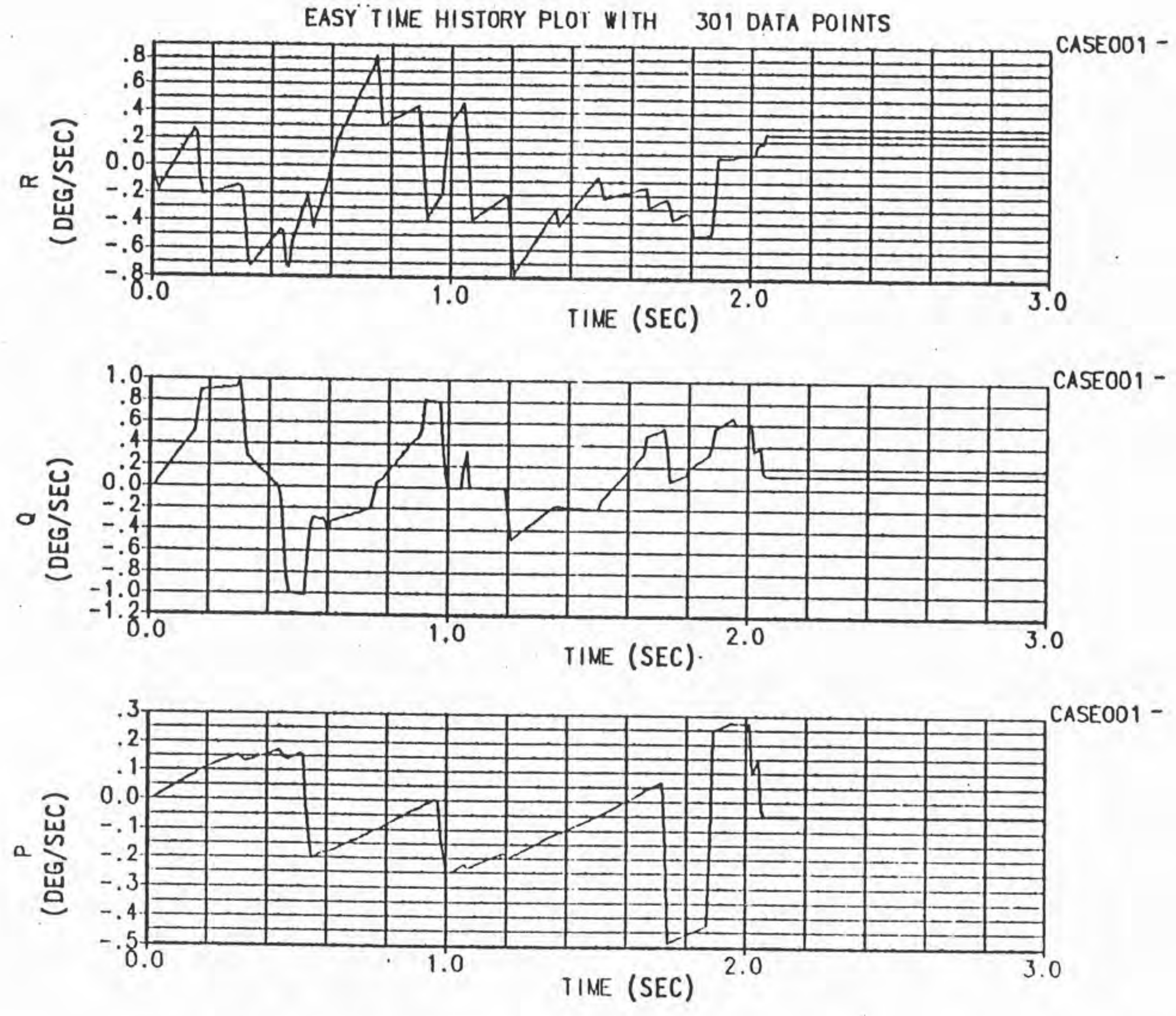


FIGURE 38 ANGULAR VELOCITY FOR AXIAL VELOCITY MANEUVER

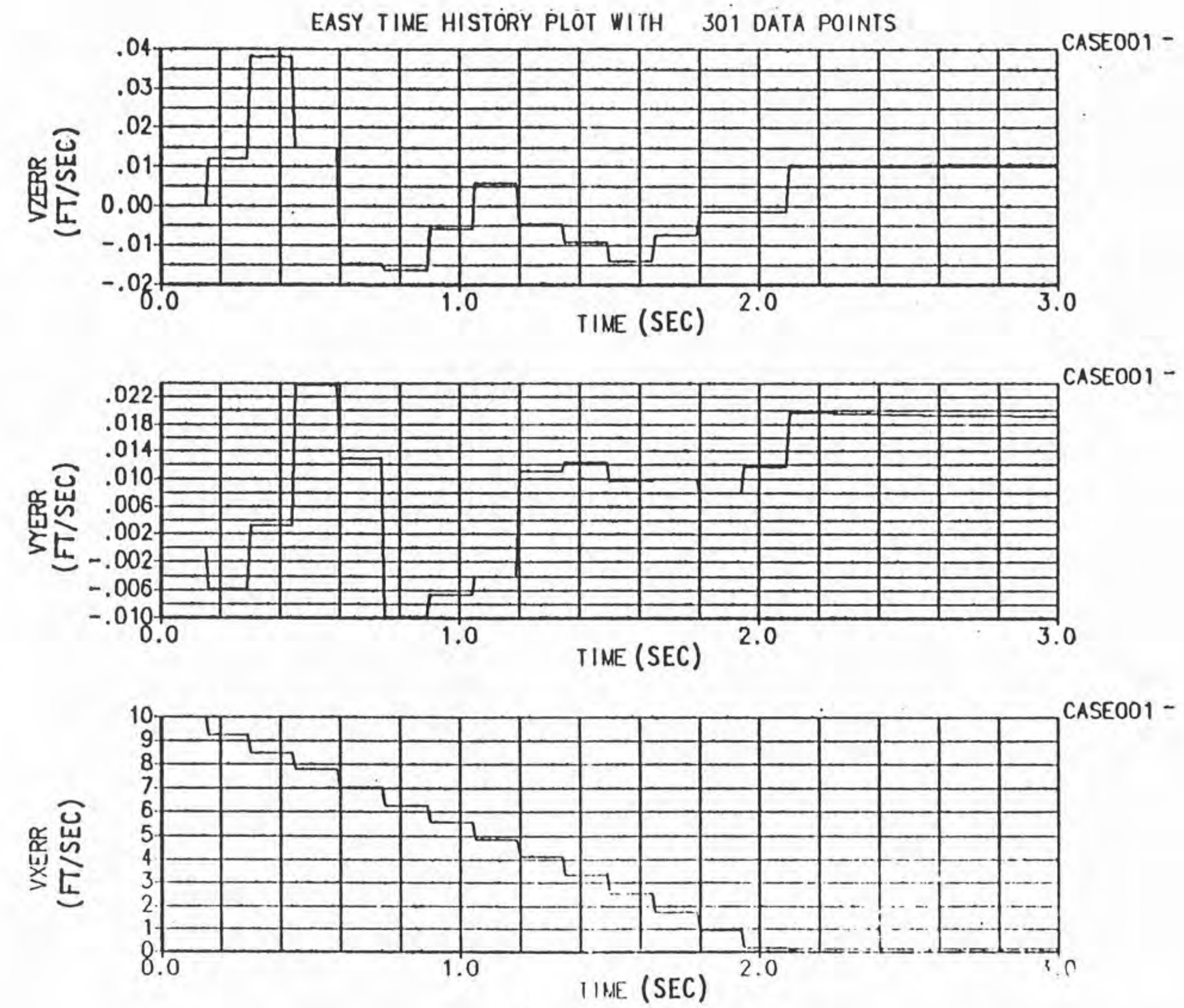


FIGURE 39 LINEAR VELOCITY ERROR FOR AXIAL VELOCITY MANEUVER

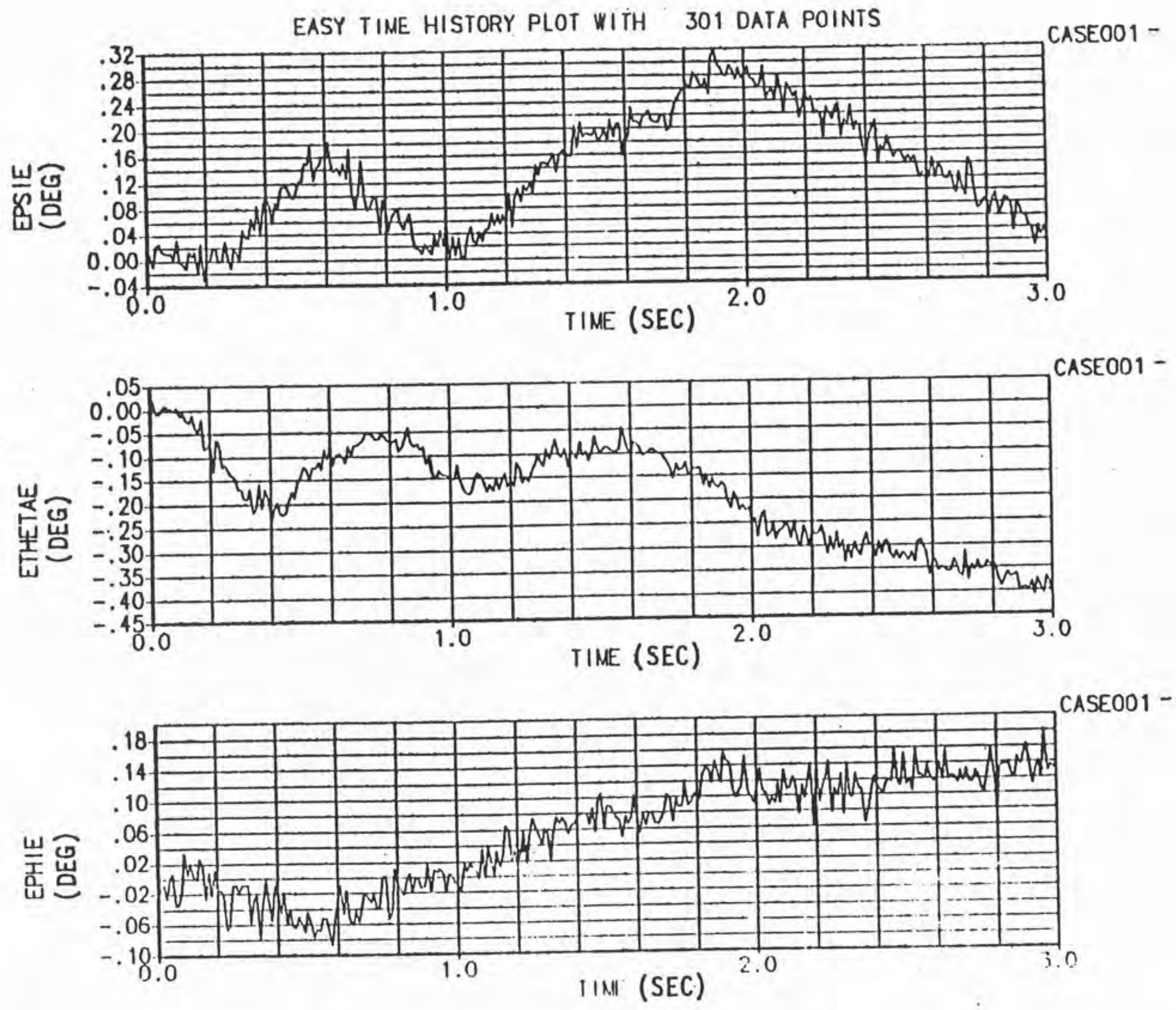


FIGURE 40 ESTIMATED ATTITUDE ERROR FOR AXIAL VELOCITY MANEUVER

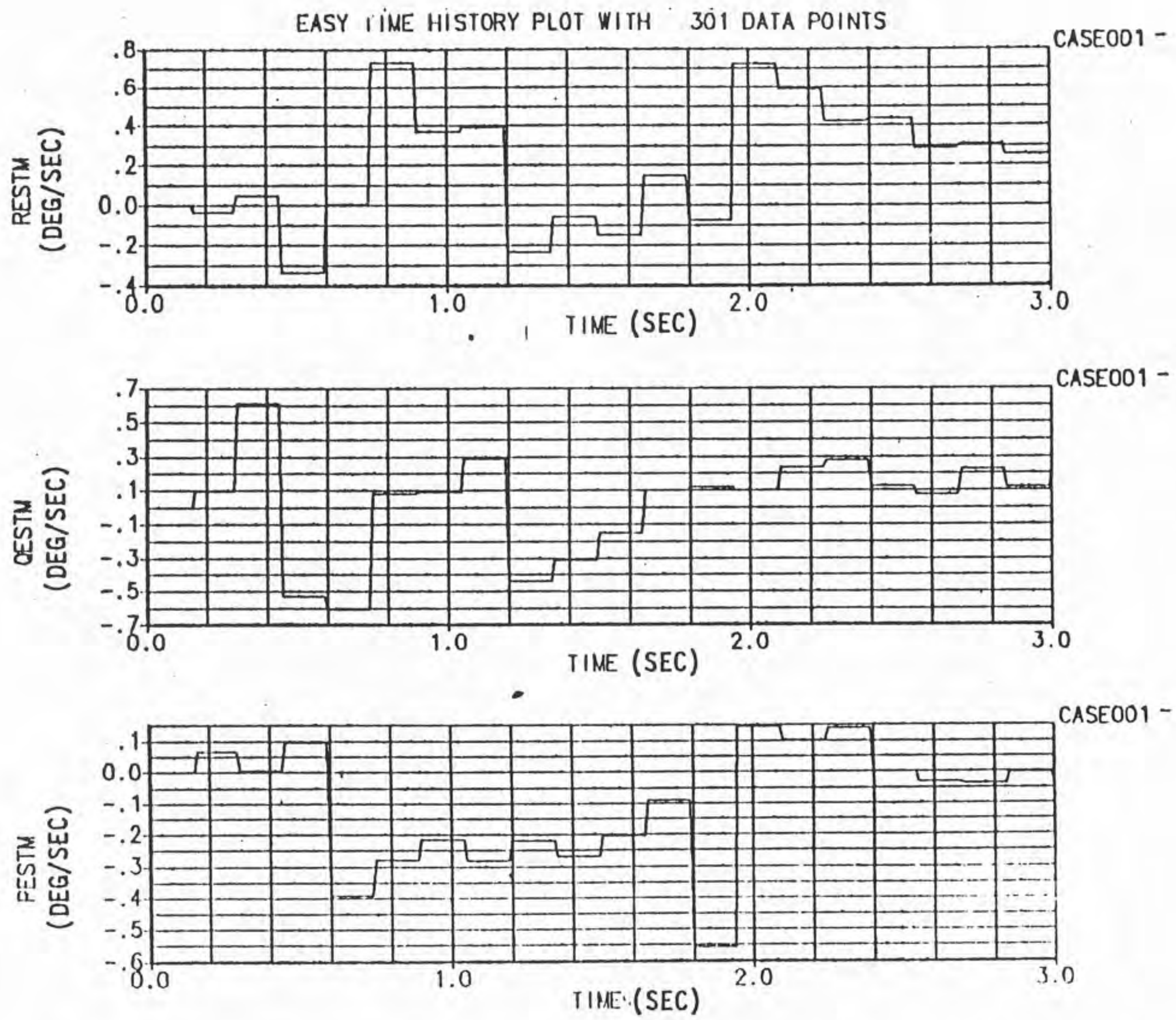


FIGURE 41 ESTIMATED ANGULAR VELOCITY FOR AXIAL VELOCITY MANEUVER

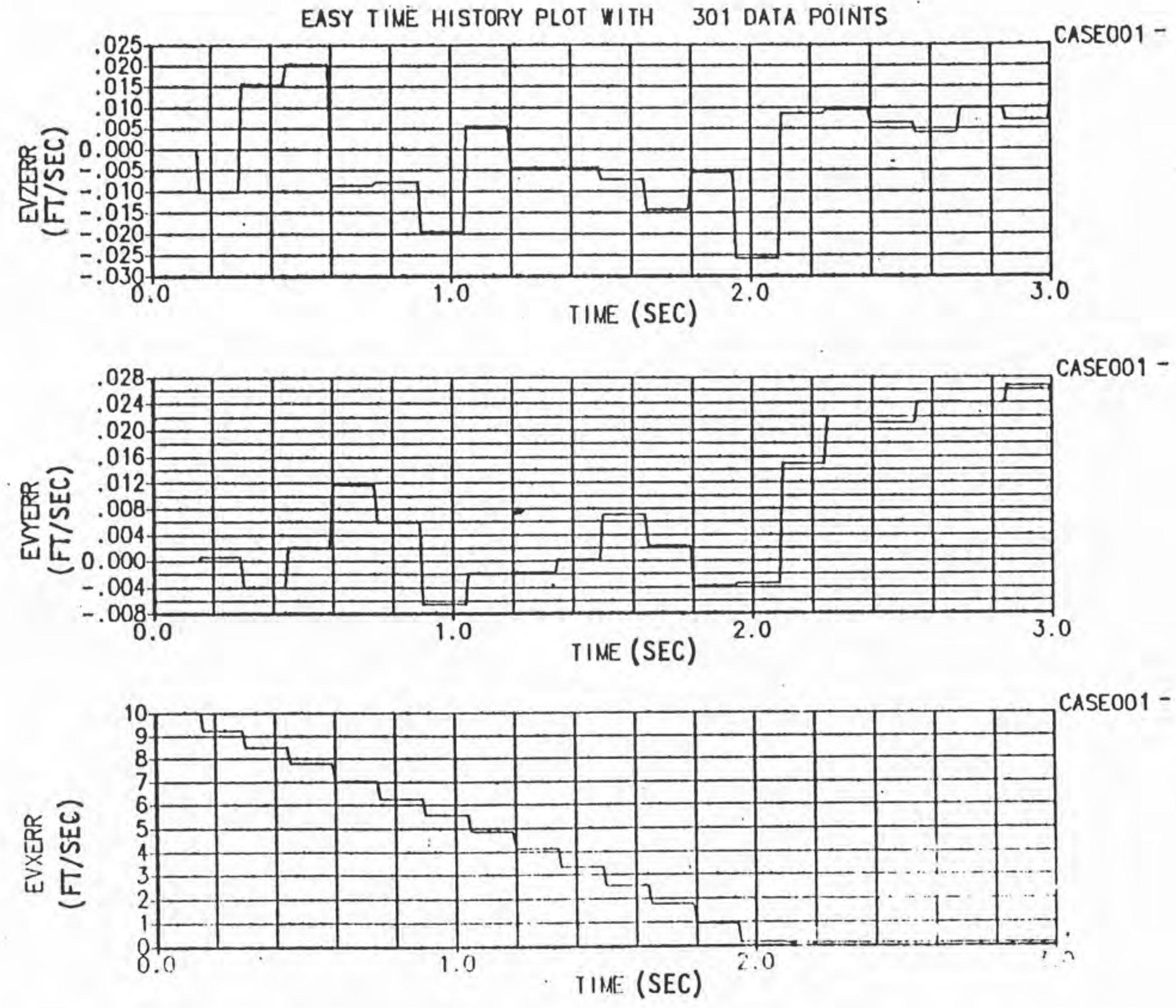


FIGURE 42 ESTIMATED LINEAR VELOCITY FOR AXIAL VELOCITY MANEUVER

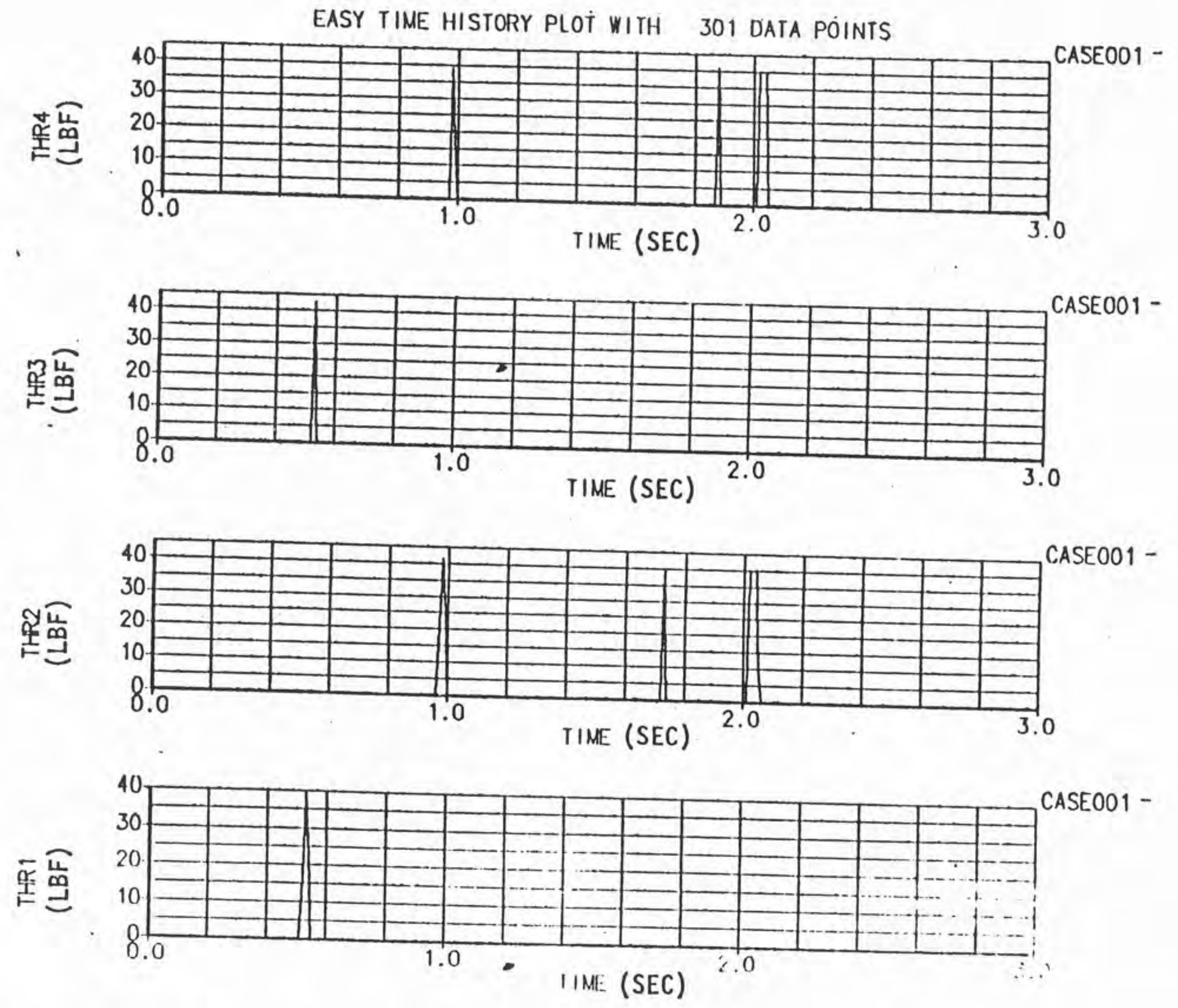


FIGURE 43 JET THRUST HISTORY FOR AXIAL VELOCITY MANEUVER

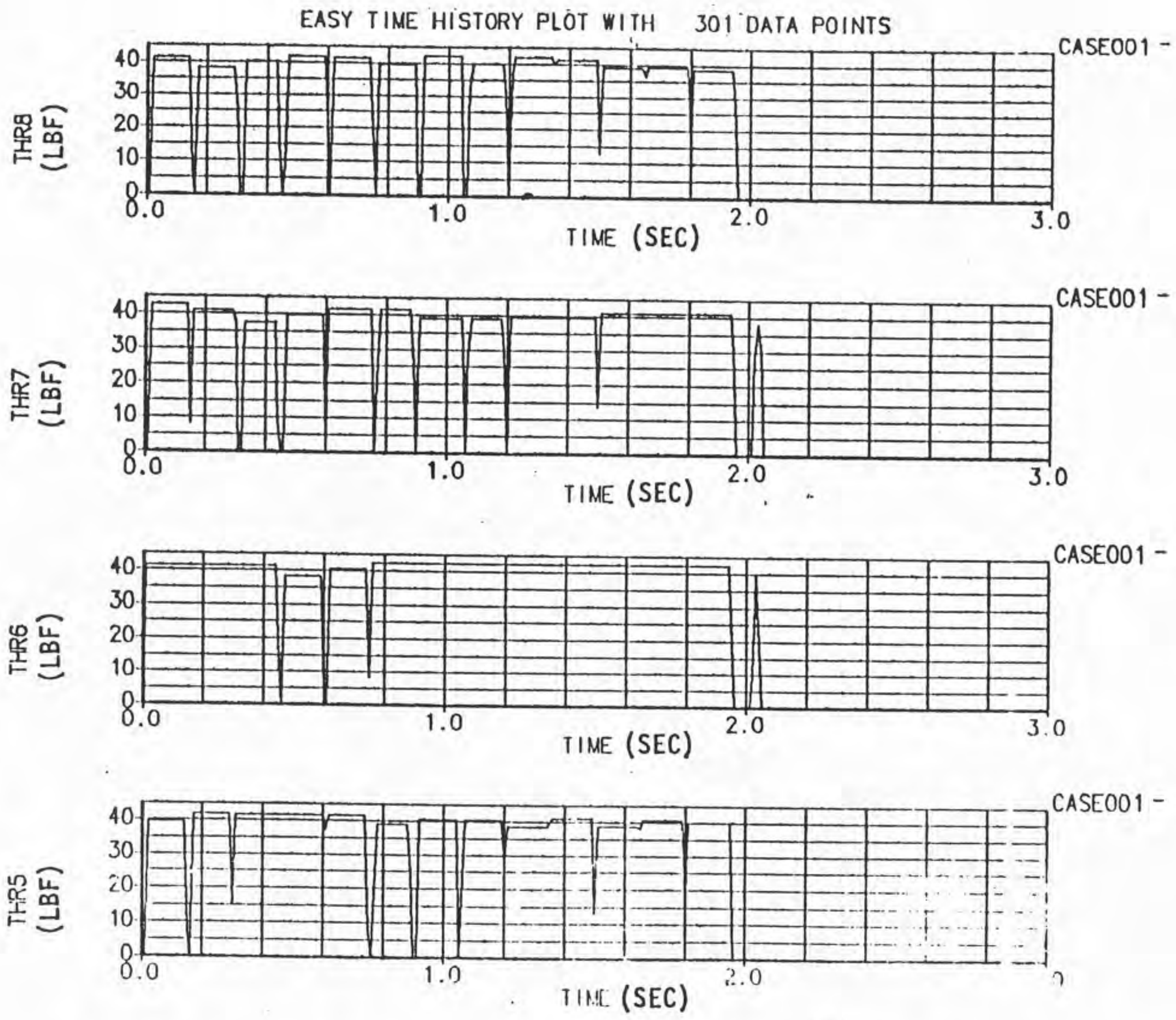


FIGURE 44 JET THRUST HISTORY FOR AXIAL VELOCITY MANEUVER

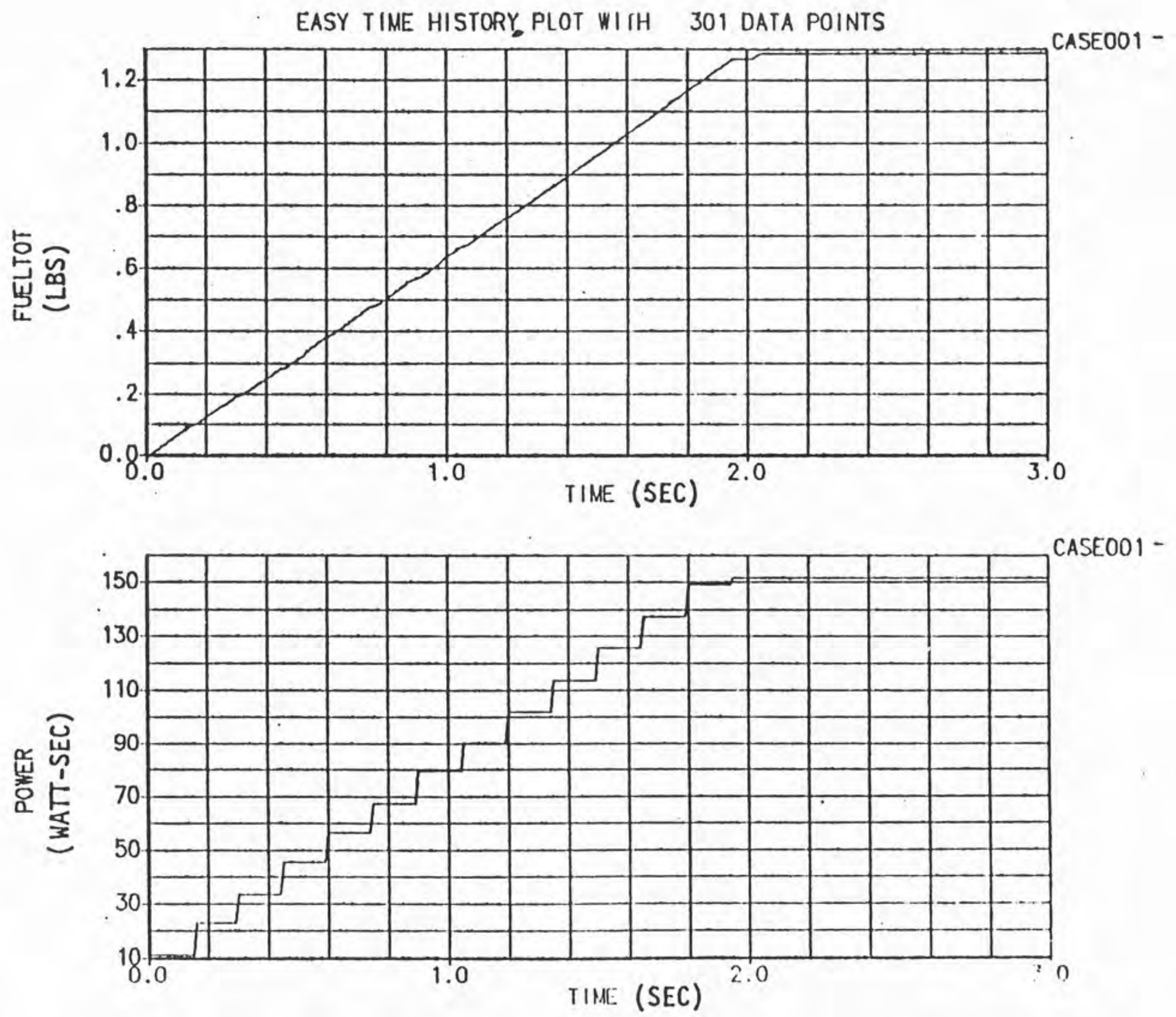


FIGURE 45 POWER AND FUEL CONSUMPTION FOR AXIAL VELOCITY MANEUVER

APPENDIX A

```
********************************************
*** PBV 6-DOF EASY5 MODEL GENERATION FILE ***
*** DESIGNED BY:      R. G. BORST  3-3-84 ***
********************************************
MODEL DESCRIPTION      PBV 6-DOF CONTROL PERFORMANCE MODEL







***********************
*** VEHICLE DYNAMICS ***
***********************
LOCATION=66    PM     INPUTS=VA(LA=LA,LO=LO,TI=TI,DA=DA),
FORT(F=F,MASSV=MA)
LOCATION=22    VA     INPUTS=FORT(T=T,INERV=IN)
LOCATION=68    PO     INPUTS=PM(R=R,RD=RD),VA(A=A,TI=TI,DA=DA)
LOCATION=44    FOTR   INPUTS=Q  VA
ADD VARIABLES    IDM(3,3),ITBDC(3,3),BTIDC(3,3)
FORTRAN STATEMENTS
C      BUILD BODY TO INERTIAL TRANSFORMATION MATRIX
       /IDM/=/1/
       CALL DCFQUAT(Q  VA,ITBDC)
C      BUILD INERTIAL TO BODY TRANSFORMATION MATRIX
       /ITBDUM/=/ITBDC/
       /BTIDC/=/ITBDUM/-1/IDM/
LOCATION=64    FOFT   INPUTS=ITBDC,BTIDC,THETAXV,THETAYV,THETAZV,CGV
ADD PARAMETERS   XT(12),YT(12),ZT(12),XIMP,YIMP,ZIMP,RPD,
IMPFLAG,NUMJET
ADD VARIABLES    FB(3),FI(3),RJETV(3),
MT(3),FBIMP(3),FIIMP(3),RIMP(3),MIMP(3),F(3),T(3)
ADD TABLES       FXIMP,20,1,FYIMP,20,1,FZIMP,20,1
FORTRAN STATEMENTS
       /F/=/0/
       NJET=NUMJET
       /T/=/0/
C      FORCES IN BODY COORDINATES
       DO 100 I=1,NJET
       FB(1)=THR(I)*COS(THETAXV(I)*RPD)
       FB(2)=THR(I)*COS(THETAYV(I)*RPD)
       FB(3)=THR(I)*COS(THETAZV(I)*RPD)
C      FORCES IN INERTIAL COORDINATES
       /FI/=/BTIDC/*/FB/
C      CALCULATE JET POSITION VECTOR
       RJETV(1)=XT(I)-CGV(1)
       RJETV(2)=YT(I)-CGV(2)
       RJETV(3)=ZT(I)-CGV(3)
C      CALCULATE TORQUES
       /MT/=/RJETV/X/FB/
C      CALCULATE FORCES AND TORQUES ABOUT VEHICLE CG DUE TO JET FIRINGS
       F(1)=F(1)+FI(1)
```

```
      F(2)=F(2)+FI(2)
      F(3)=F(3)+FI(3)
      T(1)=T(1)+MT(1)
      T(2)=T(2)+MT(2)
  100 T(3)=T(3)+MT(3)
      IF(IMPFLAG .EQ. 1.0)THEN
C     CALCULATE IMPULSE DISTURBANCE
      CALL FU(FXIMP,FBIMP(1),TIME,1.0)
      CALL FU(FYIMP,FBIMP(2),TIME,1.0)
      CALL FU(FZIMP,FBIMP(3),TIME,1.0)
      /FIIMP/=/BTIDC/*/FBIMP/
      RIMP(1)=XIMP-CGV(1)
      RIMP(2)=YIMP-CGV(2)
      RIMP(3)=ZIMP-CGV(3)
      /MIMP/=/RIMP/X/FBIMP/
C     CALCULATE TOTAL FORCES AND TORQUES ABOUT VEHICLE CG
      F(1)=F(1)+FIIMP(1)
      F(2)=F(2)+FIIMP(2)
      F(3)=F(3)+FIIMP(3)
      T(1)=T(1)+MIMP(1)
      T(2)=T(2)+MIMP(2)
      T(3)=T(3)+MIMP(3)
      END IF




*************************************************
*** STATISTICAL VARIATION OF SYSTEM PARAMETERS ***
*************************************************
LOCATION=216   FOSV
ADD PARAMETERS   CONCYL,MASSPV,SDCG,THRPV,INERPV,SDANG,SDAM,SDVM,
THETAXA(12),THETAYA(12),THETAZA(12),THRMAGA(12),COMCYL,SAFLAG,
MASSA,INERA(3,3),CGA(3)
ADD VARIABLES    MASSV,INERV(3,3),CGV(3),THRMN(12),THRMAGV(12),
THETAXV(12),THETAYV(12),THETAZV(12),PTPPV(12),QP1(4),QP2(4),
VNOISE(3),ANOISE(3),IV(3),RUV(3),UV(3),PV(3),QTRN(4),QCN(4),QA(4),
SEED(88),PASFACT(12),PASTJET(12)
ADD TABLES       IBRT,20,1
FORTRAN STATEMENTS
C     EXECUTE FIRST PASS THRU EQMO ONLY
      IF(INCALL .EQ. 2.0)THEN
C     INITIALIZE OUTPUT VARIABLES
      MASSV=MASSA
      /CGV/=/CGA/
      /INERV/=/INERA/
      /THETAXV/=/THETAXA/
      /THETAYV/=/THETAYA/
      /THETAZV/=/THETAZA/
      /THRMAGV/=/THRMAGA/
      END IF
      IF(SAFLAG .EQ. 0.0)GO TO 190
      IF(INST .EQ. 26 .AND. FSOFLAG .EQ. 0.0)THEN
C     EXECUTE FIRST PASS THRU EQMO OF FIRST SIMULATION ANALYSIS ONLY
      FSOFLAG=1.0
C     GENERATE INITIAL RANDOM SEED VALUES
      SEED(1)=7.0
      DO 110 I=2,88
      SEED(I)=SEED(I-1)
  110 CALL RNG(DUMM,SEED(I),1000.,100000.)
      END IF
C     EXECUTE FIRST PASS THRU EQMO EACH SIMULATION ANALYSIS
      IF(INST .EQ. 26 .AND. INCALL .EQ. 1.0)THEN
C     CALCULATE MASS OF VEHICLE
      CALL RNG(FACT,SEED(1),MASSPV,1.0)
      IF(FACT .LT. 0.0)FACT=0.0
      MASSV=FACT*MASSA
C     CALCULATE VEHICLE CG
      DO 120 I=1,3
  120 CALL RNG(CGV(I),SEED(I+1),SDCG,CGA(I))
C     CALCULATE VEHICLE INERTIA TENSOR
C     MASS MOMENTS OF INERTIA
      DO 130 I=1,3
      CALL RNG(FACT,SEED(I+4),INERPV,1.0)
      IF(FACT .LT. 0.0)FACT=0.0
  130 INERV(I,I)=FACT*INERA(I,I)
C     MASS PRODUCTS OF INERTIA
      DO 140 I=1,2
      CALL RNG(FACT,SEED(I+7),INERPV,1.0)
      IF(FACT .LT. 0.0)FACT=0.0
      INERV(1,I+1)=FACT*INERA(1,I+1)
  140 INERV(I+1,1)=INERV(1,I+1)
      CALL RNG(FACT,SEED(10),INERPV,1.0)
      IF(FACT .LT. 0.0)FACT=0.0
      INERV(2,3)=FACT*INERA(2,3)
      INERV(3,2)=INERV(2,3)
      K=22
      DO 160 I=1,12
C     CALCULATE STEADY STATE MEAN THRUSTER LEVELS
      PASTJET(I)=TJET(I)
      CALL RNG(FACT,SEED(I+10),THRPV,1.0)
      IF(FACT .LT. 0.0)FACT=0.0
      THRMN(I)=FACT*THRMAGA(I)
C     CALCULATE THRUSTER ORIENTATION ANGLES
      IV(1)=COS(THETAXA(I)*RPD)
      IV(2)=COS(THETAYA(I)*RPD)
      IV(3)=COS(THETAZA(I)*RPD)
      DO 150 J=1,3
      K=K+1
  150 CALL RNG(RUV(I),SEED(K),1.0,0.0)
      CALL RNG(TOL,SEED(I+58),SDANG,0.0)
      ARG=SQRT(RUV(1)**2+RUV(2)**2+RUV(3)**2)
      UV(1)=RUV(1)/ARG
      UV(2)=RUV(2)/ARG
      UV(3)=RUV(3)/ARG
      /PV/=/IV/X/UV/
      ARG=SQRT(PV(1)**2+PV(2)**2+PV(3)**2)
      PV(1)=PV(1)/ARG
      PV(2)=PV(2)/ARG
      PV(3)=PV(3)/ARG
      ARG=SIN(TOL*RPD/2.0)
      QTRN(1)=COS(TOL*RPD/2.0)
      QTRN(2)=ARG*PV(1)
      QTRN(3)=ARG*PV(2)
      QTRN(4)=ARG*PV(3)
      CALL QCONJ(QTRN,QCN)
      QA(1)=0.0
      QA(2)=IV(1)
      QA(3)=IV(2)
      QA(4)=IV(3)
      CALL QPROD(QCN,QA,QP1)
      CALL QPROD(QP1,QTRN,QP2)
      THETAXV(I)=ACOS(QP2(2))/RPD
      THETAYV(I)=ACOS(QP2(3))/RPD
  160 THETAZV(I)=ACOS(QP2(4))/RPD
      END IF
C     EXECUTE EVERY FIRING CYCLE
      IF(JETCOM1 .EQ. 1.0)THEN
      JETCOM1=0
      DO 170 I=1,12
C     DETERMINE PULSE TO PULSE PERCENT VARIATION
      CALL FU(IBRT,PTPPV(I),TJET(I),1.0)
C     CALCULATE JET THRUST LEVELS
      CALL RNG(FACT,SEED(I+70),PTPPV(I),1.0)
      IF(INCALL .EQ. 1.0)PASFACT(I)=FACT
      ARG=PASTJET(I)-CONCYL
      IF(ABS(ARG) .LT. JETCYL)FACT=PASFACT(I)
      PASFACT(I)=FACT
      PASTJET(I)=TJET(I)
      IF(FACT .LT. 0.0)FACT=0.0
  170 THRMAGV(I)=FACT*THRMN(I)
      END IF
C     EXECUTE EVERY COMPUTER CYCLE
      IF(TIME-ICOMPAS*COMCYL .LT. 0.0 .AND. INST .EQ. 26)THEN
      DO 180 I=1,3
C     CALCULATE VELOCITY MEASUREMENT NOISE
      CALL RNG(VNOISE(I),SEED(I+82),SDVM,0.0)
C     CALCULATE ATTITUDE MEASUREMENT NOISE
  180 CALL RNG(ANOISE(I),SEED(I+85),SDAM,0.0)
      END IF
  190 CONTINUE
```

****

```
*** ATTITUDE AND VELOCITY COMMANDS ***
**************************************
LOCATION=165   FOCM   INPUTS=RD2PM,ITBDC,BTIDC
ADD PARAMETERS   CEA(3),CVC(3)
ADD VARIABLES    QGUID(4),VCOM(3)
FORTRAN STATEMENTS
      IF(INCALL .GE. 1.0)THEN
C     CALCULATE GUIDANCE QUATERNION
      CALL QUATFEA(CEA,QGUID)
C     CALCULATE INITIAL INERTIAL VELOCITY COMMAND
      /VINI/=/ITBDC/*/RD2PM/
      /VCOMB/=/VINI/+/CVC/
      /VCOMI/=/BTIDC/*/VCOMB/
      END IF
C     CALCULATE CURRENT BODY AXIS VELOCITY COMMAND
      /VCOM/=/ITBDC/*/VCOMI/
```

```
*********************************************
*** QUATERNION SINGLE AXIS ROTATION ROUTINE ***
*********************************************
LOCATION=163   FOQR   INPUTS=EA VA,QGUID,ANOISE
ADD VARIABLES     QNAVA(4),QNAVM(4),EULAXE(3),AERR(3),AAERR(3),
EAWN(3),ALPHA
FORTRAN STATEMENTS
C     EXECUTE EVERY COMPUTER FRAME CYCLE
      IF(TIME-ICOMPAS*COMCYL .LT. 0.0 .AND. INST .EQ. 26)GO TO 200
C     CALCULATE ACTUAL NAVIGATION QUATERNION
      CALL QUATFEA(EA VA,QNAVA)
C     ADD NOISE TO EULER ANGLES
      EAWN(1)=EA VA(1)+ANOISE(1)
      EAWN(2)=EA VA(2)+ANOISE(2)
      EAWN(3)=EA VA(3)+ANOISE(3)
C     CALCULATE MEASURED NAVIGATION QUATERNION
      CALL QUATFEA(EAWN,QNAVM)
C     CALCULATE ACTUAL ATTITUDE CONTROL ERRORS
      CALL CEFQUAT(QNAVA,QGUID,ALPHA,EULAXE,AAERR)
C     CALCULATE ESTIMATED ATTITUDE CONTROL ERRORS
      CALL CEFQUAT(QNAVM,QGUID,ALPHA,EULAXE,AERR)
  200 CONTINUE
```

```
****************************************
*** INERTIAL MEASUREMENT UNIT MODEL ***
****************************************
LOCATION=112   FOIM   INPUTS=W  VA,RD2PM,VNOISE,AERR,ITBDC
ADD PARAMETERS   XIMU,YIMU,ZIMU,XRVCG,YRVCG,ZRVCG
ADD VARIABLES    VIMU(3),VCGS(3),PASAERR(3),VRV(3),ANGRTE(3),AIV(3),
RIMUV(3),RRVV(3)
FORTRAN STATEMENTS
C      SAMPLE EVERY COMPUTER FRAME CYCLE
       IF(TIME-ICOMPAS*COMCYL .LT. 0.0 .AND. INST .EQ. 26)GO TO 300
       IF(INCALL .GE. 1.0)THEN
       /PASAERR/=/AERR/
C      CALCULATE ACTUAL IMU POSITION VECTOR
       RIMUV(1)=XIMU-CGV(1)
       RIMUV(2)=YIMU-CGV(2)
       RIMUV(3)=ZIMU-CGV(3)
C      CALCULATE ACTUAL CONTROL POINT POSITION VECTOR
       RRVV(1)=XRVCG-CGV(1)
       RRVV(2)=YRVCG-CGV(2)
       RRVV(3)=ZRVCG-CGV(3)
       END IF
C      CALCULATE SPECIFIC VELOCITY (TOTAL VEL.-GRAV. TERM) AT VEHICLE CG
C      NOTE: GRAVITY TERM HAS BEEN REMOVED FROM STANDARD COMPONENT VA
       /VCGS/=/ITBDC/*/RD2PM/
       ANGRTE(1)=W  VA(1)*RPD
       ANGRTE(2)=W  VA(2)*RPD
       ANGRTE(3)=W  VA(3)*RPD
       /OCRIMUV/=/ANGRTE/X/RIMUV/
       /OCRRVV/=/ANGRTE/X/RRVV/
C      CALCULATE ACTUAL VELOCITY AT CONTROL POINT
       /VRV/=/VCGS/+/OCRRVV/
C      CALCULATE VELOCITY MEASURED BY IMU
       /VIMU/=/VCGS/+/OCRIMUV/
C      ADD RANDOM NOISE TO MEASURED IMU VELOCITY
       VIMU(1)=VIMU(1)+VNOISE(1)
       VIMU(2)=VIMU(2)+VNOISE(2)
       VIMU(3)=VIMU(3)+VNOISE(3)
C      EXECUTE EVERY CONTROL FRAME
   300 IF(TIME-ICONPAS*CONCYL .LT. 0.0 .AND. INST .EQ. 26)GO TO 400
C      CALCULATE ATTITUDE INCREMENT VECTOR AS PROVIDED BY IMU
       /AIV/=/PASAERR/-/AERR/
       /PASAERR/=/AERR/
   400 CONTINUE




*****************************************************
```

```
*** POST BOOST VEHICLE UNIFIED PHASE SPACE AUTOPILOT ***
********************************************************
LOCATION=135   FOA    INPUTS=▼  VA,AERR,VCOM,VIMU,AIV,ALPHA,VRV,VCGS
ADD PARAMETERS    JETCYL,TCOAST,TAU,RLIM,MIB(12),
ADB,TG01,TGODB,VDB,RDB,VLIM,LARGE,SMALL,DOTMAX,RCSFLAG,JSGAIN,RESDB,
CF1(12),CF2(12),CF3(12)
ADD VARIABLES     REST(3),RES(6),POMEGA(6),RIMUA(3),RRVA(3),
SUM(3),RGAIN(3),VERR(3),OMEGA(6),INERI(3,3),RVVERR(3),TMJET(12),
TJET(12),RCOM(3),VGAIN(3),MTM(3),ACCM(6,12),UI(3),TMIN(12),ACCN(6,12),
FXM(12),FYM(12),FZM(12),MXM(12),MYM(12),MZM(12),DP(12),TJETBS(12),
AI1(3),AI2(3),ACC(6,12),TJCOFF(12),COEFF(5),DOTSTEP,RJETA(3),NFACT(12),
TJCON(12),CGVERR(3),RCSMODE,TGO,DPMAX
FORTRAN STATEMENTS
C      ******************************
C      *** INITIALIZATION ROUTINE ***
C      ******************************
       IF(INCALL .EQ. 0.0)GO TO 1010
       IF(INST .EQ. 61)RCSMODE=RCSFLAG
       ICOMPAS=0
       ICONPAS=0
       JETCOM1=0
       JETCOM2=0
       TERMFLG=0.0
       TJETTOT=0.0
       /TJCOFF/=/0/
       /POMEGA/=/0/
       /INER/=/INERA/
       /PREST/=/▼  VA/
C      CALCULATE INVERSE INERTIA TENSOR
       /INERI/=/INER/-1/IDM/
C      CALCULATE ESTIMATED IMU POSITION VECTOR
       RIMUA(1)=XIMU-CGA(1)
       RIMUA(2)=YIMU-CGA(2)
       RIMUA(3)=ZIMU-CGA(3)
C      CALCULATE ESTIMATED CONTROL POINT POSITION VECTOR
       RRVA(1)=XRVCG-CGA(1)
       RRVA(2)=YRVCG-CGA(2)
       RRVA(3)=ZRVCG-CGA(3)
       DO 1000 I=1,NJET
C      CALCULATE MINIMUM JET FIRING TIME
       TMJET(I)=MIB(I)/THRMAGA(I)
C      CALCULATE ESTIMATED JET POSITION VECTOR
       RJETA(1)=XT(I)-CGA(1)
       RJETA(2)=YT(I)-CGA(2)
       RJETA(3)=ZT(I)-CGA(3)
       UI(1)=THRMAGA(I)*COS(THETAXA(I)*RPD)
       UI(2)=THRMAGA(I)*COS(THETAYA(I)*RPD)
       UI(3)=THRMAGA(I)*COS(THETAZA(I)*RPD)
       FXM(I)=UI(1)
       FYM(I)=UI(2)
       FZM(I)=UI(3)
       /MTM/=/RJETA/X/UI/
       MXM(I)=MTM(1)
       MYM(I)=MTM(2)
       MZM(I)=MTM(3)
       /AI1/=/INERI/*/MTM/
```

```
      AI2(1)=UI(1)/MASSA
      AI2(2)=UI(2)/MASSA
      AI2(3)=UI(3)/MASSA
C     BUILD JET ACCELERATION MATRIX
      ACC(1,I)=AI1(1)/RPD
      ACC(2,I)=AI1(2)/RPD
      ACC(3,I)=AI1(3)/RPD
      ACC(4,I)=AI2(1)
      ACC(5,I)=AI2(2)
      ACC(6,I)=AI2(3)
C     MODIFY JET ACCELERATION MATRIX BY WEIGHTING FACTOR
      ACCM(1,I)=JSGAIN*AI1(1)
      ACCM(2,I)=JSGAIN*AI1(2)
      ACCM(3,I)=JSGAIN*AI1(3)
      ACCM(4,I)=ACC(4,I)
      ACCM(5,I)=ACC(5,I)
      ACCM(6,I)=ACC(6,I)
      IF(RCSMODE .EQ. 4.0)ACCM(4,I)=0.0
      IF(RCSMODE .EQ. 5.0)THEN
      ACCM(4,I)=0.0
      ACCM(5,I)=0.0
      ACCM(6,I)=0.0
      END IF
C     CALCULATE NORMALIZING FACTORS
      NFACT(I)=SQRT(ACCM(1,I)**2+ACCM(2,I)**2+ACCM(3,I)**2+
     1ACCM(4,I)**2+ACCM(5,I)**2+ACCM(6,I)**2)
C     NORMALIZE ACCELERATION MATRIX
      ACCN(1,I)=ACCM(1,I)/NFACT(I)
      ACCN(2,I)=ACCM(2,I)/NFACT(I)
      ACCN(3,I)=ACCM(3,I)/NFACT(I)
      ACCN(4,I)=ACCM(4,I)/NFACT(I)
      ACCN(5,I)=ACCM(5,I)/NFACT(I)
 1000 ACCN(6,I)=ACCM(6,I)/NFACT(I)
C     CALCULATE COEFFICIENTS FOR "LAGLESS FILTER"
      ICOAST=TCOAST/COMCYL+1
      COEFF(1)=1.0/(TAU+CONCYL)
      COEFF(2)=(TAU+CONCYL/2.0)/(TAU+CONCYL)
      COEFF(3)=TAU/(TAU+CONCYL)
C     CALCULATE COEFFICIENTS FOR "LEAST SQUARES" ROT. VEL. ESTIMATOR
      COEFF(4)=12.0/((ICOAST**3-ICOAST)*COMCYL)
      COEFF(5)=(-12.0*(ICOAST+1)/2.0)/((ICOAST**3-ICOAST)*COMCYL)
C     INITIALIZE MANEUVER PARAMETERS
C     RCSMODE=5.0 INDICATES LARGE ATTITUDE MANUEVER
C     RCSMODE=4.0 INDICATES AXIAL VELOCITY MANUEVER
C     RCSMODE=3.0 INDICATES TERMINAL MANUEVER
C     RCSMODE=2.0 INDICATES COAST PERIOD OF TERMINAL MANUEVER
C     RCSMODE=1.0 INDICATES ATTITUDE HOLD MODE
C     RCSMODE=0.0 INDICATES MANUEVER HAS BEEN COMPLETED
      P1=RLIM
      P2=ADB
      P3=VLIM
      P4=RESDB
      /TMIN/=/0/
      IF(RCSMODE .EQ. 5.0)THEN
      P2=0.0
      P3=0.0
      TGO=TGO1
      AERRMAG=ALPHA
      HAERR=0.5*AERRMAG
      END IF
      IF(RCSMODE .EQ. 4.0)THEN
      TGO=0.90
      END IF
      IF(RCSMODE .EQ. 3.0)THEN
      P2=0.0
      TGO=2.0*(CONCYL+TCOAST)
      END IF
      IF(RCSMODE .EQ. 1.0)THEN
      P3=0.0
      END IF
C     EXECUTE EVERY COMPUTER FRAME CYCLE
 1010 IF(TIME-ICOMPAS*COMCYL .LT. 0.0 .AND. INST .EQ. 26)GO TO 7000
      IF(INST .EQ. 26)ICOMPAS=ICOMPAS+1
      AERRMAG=ALPHA
      IF(RCSMODE .NE. 2.0)GO TO 2010
C     ***************************************************
C     *** COAST ANGULAR VELOCITY ESTIMATION ROUTINE ***
C     ***************************************************
 2000 ICOUNT=ICOUNT+1
      SUM(1)=SUM(1)-(COEFF(4)*ICOUNT+COEFF(5))*AERR(1)
      SUM(2)=SUM(2)-(COEFF(4)*ICOUNT+COEFF(5))*AERR(2)
      SUM(3)=SUM(3)-(COEFF(4)*ICOUNT+COEFF(5))*AERR(3)
      ICONPAS=TIME/CONCYL+1
      IF(ICOUNT .GE. ICOAST)THEN
C     CALCULATE ANGULAR VELOCITY ESTIMATE
      /REST/=/SUM/
      RMAG=SQRT(REST(1)**2+REST(2)**2+REST(3)**2)
      GO TO 3100
      ELSE
      GO TO 7000
      END IF
C     EXECUTE EVERY CONTROL FRAME
 2010 IF(TIME-ICONPAS*CONCYL .LT. 0.0 .AND. INST .EQ. 26)GO TO 7000
      IF(INST .EQ. 26)ICONPAS=ICONPAS+1.0
C     ********************************
C     *** RATE ESTIMATION ROUTINE ***
C     ********************************
 3000 CONTINUE
      IF(RCSMODE .EQ. 0.0)THEN
      /POMEGA/=/0/
      END IF
C     CALCULATE ANGULAR VELOCITY ESTIMATE
      REST(1)=COEFF(1)*AIV(1)+COEFF(2)*POMEGA(1)+COEFF(3)*PREST(1)
      REST(2)=COEFF(1)*AIV(2)+COEFF(2)*POMEGA(2)+COEFF(3)*PREST(2)
      REST(3)=COEFF(1)*AIV(3)+COEFF(2)*POMEGA(3)+COEFF(3)*PREST(3)
      RMAG=SQRT(REST(1)**2+REST(2)**2+REST(3)**2)
 3100 ANGRTE(1)=REST(1)*RPD
      ANGRTE(2)=REST(2)*RPD
      ANGRTE(3)=REST(3)*RPD
      /PREST/=/REST/
      /OCRIMUA/=/ANGRTE/X/RIMUA/
C     CALCULATE ESTIMATED VELOCITY ERROR AT VEHICLE CG
```

```
      VERR(1)=VCOM(1)-(VIMU(1)-OCRIMUA(1))
      VERR(2)=VCOM(2)-(VIMU(2)-OCRIMUA(2))
      VERR(3)=VCOM(3)-(VIMU(3)-OCRIMUA(3))
C     CALCULATE ACTUAL VELOCITY ERROR AT VEHICLE CG
      /CGVERR/=/VCOM/-/VCGS/
C     CALCULATE ACTUAL VELOCITY ERROR AT CONTROL POINT
      /RVVERR/=/VCOM/-/VRV/
      VERRMAG=SQRT(VERR(1)**2+VERR(2)**2+VERR(3)**2)
C     TERMINATE SIMULATION AT COMPLETION OF MANEUVER
      IF(RCSMODE .EQ. 0.0 .AND. INST .EQ. 26)THEN
      TERMFLG=1.0
      GO TO 7000
      END IF
C     ********************************
C     *** CONTROL DECISION ROUTINE ***
C     ********************************
C     LARGE ATTITUDE MANUEVER
 4000 IF(RCSMODE .EQ. 5.0)THEN
      IF(AERRMAG .LT. HAERR)GO TO 4010
      TGO=TGO+CONCYL
      GO TO 4020
 4010 TGO=TGO-CONCYL
 4020 IF(TGO .GE. CONCYL)GO TO 5000
      RCSMODE=0.0
      GO TO 7000
      END IF
C     AXIAL VELOCITY MANUEVER
      IF(RCSMODE .EQ. 4.0)THEN
      IF(ABS(VERR(1)) .GT. VLIM)GO TO 5000
      RCSMODE=0.0
      GO TO 5000
      END IF
C     TERMINAL MANUEVER
      IF(RCSMODE .EQ. 3.0)THEN
      IF(AERRMAG .GE. ADB)GO TO 5000
      IF(VERRMAG .GE. VDB)GO TO 5000
      IF(RMAG .GE. RDB)GO TO 5000
      IF(INST .NE. 26)GO TO 7000
      RCSMODE=2.0
      /SUM/=/0/
      ICOUNT=0
      GO TO 2000
      END IF
C     COAST MODE
      IF(RCSMODE .EQ. 2.0)THEN
      P1=0.0
      P4=0.0
      /TMIN/=/TMJET/
      RCSMODE=0.0
      END IF
C     ATTITUDE HOLD MODE
      IF(RCSMODE .EQ. 1.0)THEN
      TGO=SMALL
      IF(AERRMAG .LT. TGODB)TGO=LARGE
      END IF
C     **************************************************
```

```
C     *** VELOCITY-TO-BE-GAINED CALCULATION ROUTINE ***
C     *********************************************
 5000 CONTINUE
      /VGAIN/=/VERR/
C     LIMIT LINEAR VELOCITY-TO-BE-GAINED
      DO 5010 I=1,3
 5010 IF(ABS(VGAIN(I)) .GT. P3)VGAIN(I)=P3*VGAIN(I)/ABS(VGAIN(I))
C     CALCULATE ANGULAR VELOCITY COMMAND
      RCOM(1)=(2.0*AERR(1)-CONCYL*REST(1))/TGO
      RCOM(2)=(2.0*AERR(2)-CONCYL*REST(2))/TGO
      RCOM(3)=(2.0*AERR(3)-CONCYL*REST(3))/TGO
C     LIMITING AND PROPORTIONAL SCALING OF ANGULAR VELOCITY COMMAND
      RCLARGE=0.0
      DO 5020 I=1,3
 5020 IF(ABS(RCOM(I)) .GT. RCLARGE)RCLARGE=ABS(RCOM(I))
      IF(RCLARGE .GT. P1)THEN
      DO 5030 I=1,3
 5030 RCOM(I)=P1*RCOM(I)/RCLARGE
      END IF
      /RGAIN/=/RCOM/-/REST/
      IF(RMAG .LT. RDB .AND. AERRMAG .LT. P2)THEN
      /RGAIN/=/0/
      END IF
C     BUILD VELOCITY-TO-BE-GAINED VECTOR
      OMEGA(1)=RGAIN(1)
      OMEGA(2)=RGAIN(2)
      OMEGA(3)=RGAIN(3)
      OMEGA(4)=VGAIN(1)
      OMEGA(5)=VGAIN(2)
      OMEGA(6)=VGAIN(3)
C     ****************************************
C     *** DOT PRODUCT JET SELECTION ROUTINE ***
C     ****************************************
 6000 DOTSTEP=0.0
      TI=0.0
      /TJET/=/0/
C     MODIFY VELOCITY-TO-BE-GAINED VECTOR BY WEIGHTING FACTOR
      RES(1)=JSGAIN*RPD*OMEGA(1)
      RES(2)=JSGAIN*RPD*OMEGA(2)
      RES(3)=JSGAIN*RPD*OMEGA(3)
      RES(4)=OMEGA(4)
      RES(5)=OMEGA(5)
      RES(6)=OMEGA(6)
C     FIRE AXIAL JETS IF THIS IS AN AXIAL MANUEVER
      IF(RCSMODE .NE. 4.0)GO TO 6010
C     FIRE JET NUMBERS 1 AND 2 FOR NEGATIVE VELOCITY GAIN
      IF(RES(4) .LT. 0.0)THEN
      TJET(1)=CONCYL
      TJET(2)=CONCYL
      ELSE
C     FIRE JET NUMBERS 5,6,7, AND 8 FOR POSITIVE VELOCITY GAIN
      TJET(5)=CONCYL
      TJET(6)=CONCYL
      TJET(7)=CONCYL
      TJET(8)=CONCYL
      END IF
      GO TO 6050
C     TOP OF ITERATION LOOP
 6010 DOTSTEP=DOTSTEP+1.0
C     CALCULATE RESIDUE MAGNITUDE
      RESMAG=SQRT(RES(1)**2+RES(2)**2+RES(3)**2+
     1RES(4)**2+RES(5)**2+RES(6)**2)
C     EXIT ITERATION LOOP IF RESIDUE MAGNITUDE IS WITHIN DEADBAND
      IF(RESMAG .LT. P4)GO TO 6070
C     EXIT ROUTINE IF NUMBER OF ITERATIONS EXCEEDS MAXIMUM ALLOWED
      IF(DOTSTEP .GT. DOTMAX)GO TO 6070
C     CALCULATE DOT PRODUCTS OF EACH JET
      /DP/=/0/
      DO 6030 I=1,6
      DO 6020 J=1,NJET
 6020 DP(J)=DP(J)+RES(I)*ACCN(I,J)
 6030 CONTINUE
C     DETERMINE WHICH JET HAS LARGEST DOT PRODUCT
      DPMAX=0.0
      DO 6040 I=1,NJET
      IF(DP(I) .LE. DPMAX)GO TO 6040
      DPMAX=DP(I)
      LDP=I
 6040 CONTINUE
C     CALCULATE TIME TO FIRE JET WITH LARGEST DOT PRODUCT
      TI=DPMAX/NFACT(LDP)
      IF(TJET(LDP)+TI .LT. TMIN(LDP))TI=TMIN(LDP)
C     TMIN IS ZERO EXCEPT DURING TERMINAL MANUEVER
C     CALCULATE TOTAL "ON TIME" OF JET
      TJET(LDP)=TJET(LDP)+TI
C     CALCULATE REMAINING RESIDUES
 6050 FACT=RPD*JSGAIN
      /CHANGE/=/ACCM/*/TJET/
      DO 6060 I=1,6
      IF(I .GE. 4)FACT=1.0
 6060 RES(I)=FACT*OMEGA(I)-CHANGE(I)
C     SET RESIDUES TO ZERO IF LINEAR VELOCITY CONTROL NOT NEEDED
      IF(P3 .EQ. 0.0)THEN
      RES(4)=0.0
      RES(5)=0.0
      RES(6)=0.0
      END IF
      IF(RCSMODE .EQ. 4.0)RES(4)=0.0
C     GO TO TOP OF ITERATION LOOP
      GO TO 6010
C     SCALE JET TIMES TO FIT WITHIN CONTROL CYCLE
 6070 TLARGE=CONCYL
      /CHANGE/=/ACC/*/TJET/
      /TJETBS/=/TJET/
      DO 6080 I=1,NJET
 6080 IF(TJET(I) .GT. TLARGE)TLARGE=TJET(I)
      SCALE=CONCYL/TLARGE
      DO 6090 I=1,NJET
      TJET(I)=SCALE*TJET(I)
 6090 IF(TJET(I) .LT. TMJET(I))TJET(I)=0.0
      /POMEGA/=/ACC/*/TJET/
      DO 6100 I=1,NJET
```

```
C      ADJUST JET FIRING TIME VECTOR FOR BUILD-UP AND DECAY EFFECTS
       TJET(I)=CF1(I)+CF2(I)*TJET(I)+CF3(I)*TJET(I)**2
C      CENTER PULSE IN CONTROL FRAME AND CONVERT TO "ELECTRICAL TIME"
       TJET(I)=JETCYL*NINT(TJET(I)/JETCYL)
       IF(TJET(I) .GT. CONCYL)TJET(I)=CONCYL
       IF(TJET(I) .LT. CF1(I)+JETCYL)THEN
       TJET(I)=0.0
       TJCON(I)=1E+20
       ELSE
       ARG=((CONCYL-TJET(I))/2.0+TIME)/JETCYL
       TJCON(I)=JETCYL*NINT(ARG)
       END IF
       TJCOFF(I)=TJCON(I)+TJET(I)
 6100  TJETTOT=TJETTOT+TJET(I)
       JETCOM1=1
       JETCOM2=1
 7000  CONTINUE
```

```
***********************************
*** REACTION CONTROL JET MODEL ***
***********************************
LOCATION=138    FOTM    INPUTS=TJCON,TJCOFF,THRMAGV
ADD PARAMETERS    ENGFLAG
ADD VARIABLES     THR(12),TJON(12),TJOFF(12),PERON(12),PEROFF(12),
TTJCOFF(12),THRFLAG(12)
ADD TABLES        JBUH,20,1,JDH,20,1,RJBUH,20,1,RJDH,20,1
FORTRAN STATEMENTS
      IF(INCALL .GE. 1.0)THEN
      /THR/=/0/
      GO TO 7050
      END IF
      IF(ENGFLAG .EQ. 1.0)GO TO 7020
C     APPROXIMATE THRUST LEVELS AS SQUARE WAVES
      /THRFLAG/=/0/
      DO 7010 I=1,NJET
      IF(TIME .GE. TJCON(I) .AND. TIME .LE. TJCOFF(I))THRFLAG(I)=1.0
 7010 THR(I)=THRFLAG(I)*THRMAGV(I)
      GO TO 7050
C     USE TABLE VALUES TO MODEL REACTION CONTROL JETS
 7020 IF(JETCOM2 .EQ. 1)THEN
      /TTJCOFF/=/0/
      END IF
      DO 7040 I=1,NJET
      IF(THR(I) .NE. 0.0 .AND. JETCOM2 .EQ. 1)THEN
      CALL FU(RJDH,TJOFF(I),PEROFF(I),-1.0)
      TTJCOFF(I)=TIME-TJOFF(I)
      END IF
      IF(TTJCOFF(I) .EQ. 0.0)GO TO 7030
      IF(TIME .LE. TJCON(I))THEN
      PERON(I)=1.0
      TJOFF(I)=TIME-TTJCOFF(I)
      CALL FU(JDH,PEROFF(I),TJOFF(I),-1.0)
      GO TO 7040
      ELSE
      CALL FU(RJBUH,TJON(I),PEROFF(I),-1.0)
      TJCON(I)=TIME-TJON(I)
      TTJCOFF(I)=0.0
      END IF
 7030 IF(TIME .GT. TJCON(I))THEN
      TJON(I)=TIME-TJCON(I)
      CALL FU(JBUH,PERON(I),TJON(I),-1.0)
      ELSE
      PERON(I)=0.0
      END IF
      TJOFF(I)=TIME-TJCOFF(I)
      IF(TJOFF(I) .GT. 0.0)THEN
      CALL FU(JDH,PEROFF(I),TJOFF(I),-1.0)
      ELSE
      PEROFF(I)=1.0
      END IF
 7040 THR(I)=THRMAGV(I)*PERON(I)*PEROFF(I)
      JETCOM2=0
 7050 CONTINUE
```

```
*************************************************
*** FUEL AND POWER CONSUMPTION CALCULATIONS ***
*************************************************
LOCATION=168   INFM   N=12   INPUTS=FORT(THR=S)
LOCATION=170   FOFC   INPUTS=S2 INFM
ADD PARAMETERS   FISP,PFACTOR
ADD VARIABLES    FUEL(12),FUELTOT,POWER
FORTRAN STATEMENTS
      FUELTOT=0.0
      DO 7060 I=1,NJET
C     CALCULATE FUEL CONSUMPTION FOR EACH JET
      FUEL(I)=S2 INFM(I)/FISP
C     CALCULATE TOTAL FUEL CONSUMPTION
 7060 FUELTOT=FUELTOT+FUEL(I)
C     CALCULATE POWER REQUIREMENTS(I.E. WATT-SECONDS)
      POWER=PFACTOR*TJETTOT
```

```
**********************************
*** STATISTICAL ERROR ANALYSIS ***
**********************************
LOCATION=236   FOEA   INPUTS=▼ VA,RVVERR,AAERR
ADD PARAMETERS   SICFLAG
ADD VARIABLES    MEAN(9),MX(9),MN(9),SD(9),RMS(9),NS(9)
FORTRAN STATEMENTS
      IF(SAFLAG .EQ. 0.0)GO TO 7080
      IF(INST .EQ. 26 .AND. INCALL .EQ. 1.0)THEN
      TSFLAG=1.0
      END IF
      IF(TERMFLG .EQ. 1.0 .AND. TSFLAG .EQ. 1.0)THEN
      TSFLAG=0.0
      ARG=SICFLAG
C     EXECUTE AT END OF MANUEVER ONLY
      DO 7070 I=1,3
C     FINAL ANGULAR RATE ERROR ANALYSIS
      CALL STAT(MEAN(I),MX(I),MN(I),SD(I),RMS(I),NS(I),▼ VA(I),ARG)
      J=I+3
C     FINAL LINEAR VELOCITY ERROR ANALYSIS
      CALL STAT(MEAN(J),MX(J),MN(J),SD(J),RMS(J),NS(J),RVVERR(I),ARG)
      K=I+6
C     FINAL ATTITUDE ERROR ANALYSIS
 7070 CALL STAT(MEAN(K),MX(K),MN(K),SD(K),RMS(K),NS(K),AAERR(I),ARG)
      SICFLAG=0.0
      END IF
 7080 CONTINUE
```

```
***************************************
*** PLOT PARAMETER RENAMING ROUTINE ***
***************************************
LOCATION=256   FORP
ADD VARIABLES   PHIE,THETAE,PSIE,P,Q,R,VXERR,VYERR,VZERR,EPHIE,ETHETAE,
EPSIE,PESTM,QESTM,RESTM,EVXERR,EVYERR,EVZERR,THR1,THR2,THR3,THR4,THR5,
THR6,THR7,THR8,THR9,THR10,THR11,THR12
FORTRAN STATEMENTS
C      RENAME STATES, AND VARIABLES FOR PLOTTING
      PHIE=AAERR(1)
      THETAE=AAERR(2)
      PSIE=AAERR(3)
      P=▼ VA(1)
```

```
Q=W   VA(2)
R=W   VA(3)
PESTM=REST(1)
QESTM=REST(2)
RESTM=REST(3)
VXERR=RVVERR(1)
VYERR=RVVERR(2)
VZERR=RVVERR(3)
EVXERR=VERR(1)
EVYERR=VERR(2)
EVZERR=VERR(3)
EPHIE=AERR(1)
ETHETAE=AERR(2)
EPSIE=AERR(3)
THR1=THR(1)
THR2=THR(2)
THR3=THR(3)
THR4=THR(4)
THR5=THR(5)
THR6=THR(6)
THR7=THR(7)
THR8=THR(8)
THR9=THR(9)
THR10=THR(10)
THR11=THR(11)
THR12=THR(12)
VXCGERR=CGVERR(1)
VYCGERR=CGVERR(2)
VZCGERR=CGVERR(3)
PMEAN=MEAN(1)
QMEAN=MEAN(2)
RMEAN=MEAN(3)
VXEMEAN=MEAN(4)
VYEMEAN=MEAN(5)
VZEMEAN=MEAN(6)
PHEMEAN=MEAN(7)
THEMEAN=MEAN(8)
PSEMEAN=MEAN(9)
PSD=SD(1)
QSD=SD(2)
RSD=SD(3)
VXESD=SD(4)
VYESD=SD(5)
VZESD=SD(6)
PHESD=SD(7)
THESD=SD(8)
PSESD=SD(9)
PMIN=MN(1)
QMIN=MN(2)
RMIN=MN(3)
VXEMIN=MN(4)
VYEMIN=MN(5)
VZEMIN=MN(6)
PHEMIN=MN(7)
THEMIN=MN(8)
PSEMIN=MN(9)
```

```
      PMAX=MX(1)
      QMAX=MX(2)
      RMAX=MX(3)
      VXEMAX=MX(4)
      VYEMAX=MX(5)
      VZEMAX=MX(6)
      PHEMAX=MX(7)
      THEMAX=MX(8)
      PSEMAX=MX(9)
      PSAM=NS(1)
      QSAM=NS(2)
      RSAM=NS(3)
      VXESAM=NS(4)
      VYESAM=NS(5)
      VZESAM=NS(6)
      PHESAM=NS(7)
      THESAM=NS(8)
      PSESAM=NS(9)
```

```
*************************************
*** USER DEFINED OUTPUT STATEMENTS ***
*************************************
PRINT STATEMENTS
      WRITE(6,8000)TIME
      WRITE(6,8010)PHIE,THETAE,PSIE
      WRITE(6,8020)P,Q,R
      WRITE(6,8030)VXERR,VYERR,VZERR
      WRITE(6,8040)EPHIE,ETHETAE,EPSIE
      WRITE(6,8050)PESTM,QESTM,RESTM
      WRITE(6,8060)EVXERR,EVYERR,EVZERR
      WRITE(6,8070)THR1,THR2,THR3,THR4
      WRITE(6,8080)THR5,THR6,THR7,THR8
      WRITE(6,8090)THR9,THR10,THR11,THR12
      WRITE(6,8100)VXCGERR,VYCGERR,VZCGERR
      WRITE(6,8110)PHEMEA,PHESD,PHEMIN,PHEMAX,PHESAM
      WRITE(6,8120)THEMEA,THESD,THEMIN,THEMAX,THESAM
      WRITE(6,8130)PSEMEA,PSESD,PSEMIN,PSEMAX,PSESAM
      WRITE(6,8140)PMEA,PSD,PMIN,PMAX,PSAM
      WRITE(6,8150)QMEA,QSD,QMIN,QMAX,QSAM
      WRITE(6,8160)RMEA,RSD,RMIN,RMAX,RSAM
      WRITE(6,8170)VXEMEA,VXESD,VXEMIN,VXEMAX,VXESAM
      WRITE(6,8180)VYEMEA,VYESD,VYEMIN,VYEMAX,VYESAM
      WRITE(6,8190)VZEMEA,VZESD,VZEMIN,VZEMAX,VZESAM
      WRITE(6,8200)RCSMODE
 8000 FORMAT(/,/,/,/,/,40X,4H*** ,'TIME=',F8.4,4H ***,/)
 8010 FORMAT(1X,'PHIE=',G12.5,' THETAE=',G12.5,' PSIE=',G12.5)
 8020 FORMAT(1X,'P=',G12.5,' Q=',G12.5,' R=',G12.5)
 8030 FORMAT(1X,'VXERR=',G12.5,' VYERR=',G12.5,' VZERR=',G12.5)
 8040 FORMAT(1X,'EPHIE=',G12.5,' ETHETAE=',G12.5,' EPSIE=',G12.5)
 8050 FORMAT(1X,'RESTM=',G12.5,' QESTM=',G12.5,' RESTM=',G12.5)
 8060 FORMAT(1X,'EVXERR=',G12.5,' EVYERR=',G12.5,' EVZERR=',G12.[illegible]
 8070 FORMAT(1X,'THR1=',G12.5,' THR2=',G12.5,' THR3=',G12.5,' TH =',
     1G12.5)
 8080 FORMAT(1X,'THR5=',G12.5,' THR6=',G12.5,' THR7=',G12.5,' TH =',
     1G12.5)
 8090 FORMAT(1X,'THR9=',G12.5,' THR10=',G12.5,' THR11=',G12.5,' THR12='
     1,G12.5)
 8100 FORMAT(1X,'VXCGERR=',G12.5,' VYCGERR=',G12.5,' VZCGERR=',G1[illegible]
 8110 FORMAT(1X,'PHEMEAN=',G12.5,' PHESD=',G12.5,' PHEMIN=',G12.5,
     1' PHEMAX=',G12.5,' PHESAM=',G12.5)
 8120 FORMAT(1X,'THEMEAN=',G12.5,' THESD=',G12.5,' THEMIN=',G12.5
     1' THEMAX=',G12.5,' THESAM=',G12.5)
 8130 FORMAT(1X,'PSEMEAN=',G12.5,' PSESD=',G12.5,' PSEMIN=',G12.5,
     1' PSEMAX=',G12.5,' PSESAM=',G12.5)
 8140 FORMAT(1X,'PMEAN=',G12.5,' PSD=',G12.5,' PMIN=',G12.5,
     1' PMAX=',G12.5,' PSAM=',G12.5)
 8150 FORMAT(1X,'QMEAN=',G12.5,' QSD=',G12.5,' QMIN=',G12.5,
     1' QMAX=',G12.5,' QSAM=',G12.5)
 8160 FORMAT(1X,'RMEAN=',G12.5,' RSD=',G12.5,' RMIN=',G12.5,
     1' RMAX=',G12.5,' RSAM=',G12.5)
 8170 FORMAT(1X,'VXEMEAN=',G12.5,' VXESD=',G12.5,' VXEMIN=',G12.5,
     1' VXEMAX=',G12.5,' VXESAM=',G12.5)
 8180 FORMAT(1X,'VYEMEAN=',G12.5,' VYESD=',G12.5,' VYEMIN=',G12.5,
     1' VYEMAX=',G12.5,' VYESAM=',G12.5)
 8190 FORMAT(1X,'VZEMEAN=',G12.5,' VZESD=',G12.5,' VZEMIN=',G12.5
     1' VZEMAX=',G12.5,' VZESAM=',G12.5)
 8200 FORMAT(1X,'RCSMODE=',G12.5)
ALPHABETIZE
END OF MODEL
PRINT
```

APPENDIX B

```
****************************************
*** PBV 6 - DOF EASY5 ANALYSIS FILE ***
*** DESIGNED BY: R. G. BORST 3-3-84 ***
****************************************
```

```
************************************
*** VEHICLE DYNAMICS PARAMETERS ***
************************************
PARAMETER VALUES
NUMJET=8.0
XT(1)=0.417,XT(2)=0.417,XT(3)=0.417,XT(4)=0.417
XT(5)=0.0,XT(6)=0.0,XT(7)=0.0,XT(8)=0.0
XT(9)=0.0,XT(10)=0.0,XT(11)=0.0,XT(12)=0.0
YT(1)=0.0,YT(2)=0.0,YT(3)=-2.083,YT(4)=2.083
YT(5)=0.0,YT(6)=2.083,YT(7)=0.0,YT(8)=-2.083
YT(9)=0.0,YT(10)=0.0,YT(11)=0.0,YT(12)=0.0
ZT(1)=-2.083,ZT(2)=2.083,ZT(3)=0.0,ZT(4)=0.0
ZT(5)=-2.083,ZT(6)=0.0,ZT(7)=2.083,ZT(8)=0.0
ZT(9)=0.0,ZT(10)=0.0,ZT(11)=0.0,ZT(12)=0.0
IMPFLAG=0.0
XIMP=0.0,YIMP=0.0,ZIMP=0.0
XRVCG=3.75,YRVCG=0.0,ZRVCG=0.0
AL1PM=971270.0,VE1PM=23423.0,AZ1PM=-90.0,GA1PM=0.0,ENGPM=1.0
ROLVA=0.0,PITVA=0.0,YAWVA=-90.0
LA1VA=70.0,LO1VA=160.0,TI1VA=0.0,DA1VA=80.0
RADPO=20927491.0
```

```
************************************************
*** ATTITUDE AND VELOCITY COMMAND PARAMETERS ***
************************************************
CEA(1)=160.0,CEA(2)=0.0,CEA(3)=-110.0
CVC(1)=0.0,CVC(2)=0.0,CVC(3)=0.0
```

```
****************************************
*** STATISTICAL ANALYSIS PARAMETERS ***
****************************************
SAFLAG=0.0
SICFLAG=1.0
MASSPV=0.0018,SDCG=0.0067,INERPV=0.0018,SDANG=0.33
THRPV=0.0167,SDAM=0.0167,SDVM=0.00167
```

```
*********************************************
*** INERTIAL MEASUREMENT UNIT PARAMETERS ***
*********************************************
XIMU=1.375,YIMU=-0.9167,ZIMU=0.0
COMCYL=0.01
```

```
************************************************
*** UNIFIED PHASE SPACE AUTOPILOT PARAMETERS ***
************************************************
MASSA=30.0
INERA(1,1)=34.4,INERA(1,2)=-1.49,INERA(1,3)=-0.69
INERA(2,1)=-1.49,INERA(2,2)=131.6,INERA(2,3)=-0.02
INERA(3,1)=-0.69,INERA(3,2)=-0.02,INERA(3,3)=139.1
CGA(1)=2.75,CGA(2)=0.0558,CGA(3)=0.0025
THETAXA(1)=118.66,THETAXA(2)=118.66,THETAXA(3)=118.66,THETAXA(4)=118.66
THETAXA(5)=0.0,THETAXA(6)=0.0,THETAXA(7)=0.0,THETAXA(8)=0.0
THETAXA(9)=90.0,THETAXA(10)=90.0,THETAXA(11)=90.0,THETAXA(12)=90.0
THETAYA(1)=113.58,THETAYA(2)=66.42,THETAYA(3)=38.67,THETAYA(4)=141.33
THETAYA(5)=90.0,THETAYA(6)=90.0,THETAYA(7)=90.0,THETAYA(8)=90.0
THETAYA(9)=90.0,THETAYA(10)=90.0,THETAYA(11)=90.0,THETAYA(12)=90.0
THETAZA(1)=38.67,THETAZA(2)=141.33,THETAZA(3)=113.58,THETAZA(4)=66.42
THETAZA(5)=90.0,THETAZA(6)=90.0,THETAZA(7)=90.0,THETAZA(8)=90.0
THETAZA(9)=90.0,THETAZA(10)=90.0,THETAZA(11)=90.0,THETAZA(12)=90.0
THRMAGA(1)=40.0,THRMAGA(2)=40.0,THRMAGA(3)40.0,THRMAGA(4)=40.0
THRMAGA(5)=40.0,THRMAGA(6)=40.0,THRMAGA(7)=40.0,THRMAGA(8)=40.0
THRMAGA(9)=0.0,THRMAGA(10)=0.0,THRMAGA(11)=0.0,THRMAGA(12)=0.0
```

```
CF1(1)=0.0035,CF1(2)=0.0035,CF1(3)=0.0035,CF1(4)=0.0035
CF1(5)=0.0035,CF1(6)=0.0035,CF1(7)=0.0035,CF1(8)=0.0035
CF1(9)=0.0,CF1(10)=0.0,CF1(11)=0.0,CF1(12)=0.0
CF2(1)=1.0,CF2(2)=1.0,CF2(3)=1.0,CF2(4)=1.0
CF2(5)=1.0,CF2(6)=1.0,CF2(7)=1.0,CF2(8)=1.0
CF2(9)=0.0,CF2(10)=0.0,CF2(11)=0.0,CF2(12)=0.0
CF3(1)=0.0,CF3(2)=0.0,CF3(3)=0.0,CF3(4)=0.0,CF3(5)=0.0,CF3(6)=0.0
CF3(7)=0.0,CF3(8)=0.0,CF3(9)=0.0,CF3(10)=0.0,CF3(11)=0.0,CF3(12)=0.0
RPD=0.01745329252
JETCYL=0.0003,CONCYL=0.150
TAU=0.25,TG01=0.31,TCOAST=0.75,JSGAIN=1.0
TGODB=0.8,LARGE=10.0,SMALL=1.0,DOTMAX=500.0
ADB=0.5,RDB=0.5,RLIM=20.0,VDB=0.03,VLIM=0.04,RESDB=0.004
RCSFLAG=3.0




****************************************
*** REACTION CONTROL JET PARAMETERS ***
****************************************
ENGFLAG=0.0
FISP=238.0
PFACTOR=20.0
MIB(1)=0.40,MIB(2)=0.40,MIB(3)=0.40,MIB(4)=0.40
MIB(5)=0.40,MIB(6)=0.40,MIB(7)=0.40,MIB(8)=0.40
MIB(9)=0.0,MIB(10)=0.0,MIB(11)=0.0,MIB(12)=0.0
GKIINFM(1)=1.0,GKIINFM(2)=1.0,GKIINFM(3)=1.0,GKIINFM(4)=1.0
GKIINFM(5)=1.0,GKIINFM(6)=1.0,GKIINFM(7)=1.0,GKIINFM(8)=1.0
GKIINFM(9)=1.0,GKIINFM(10)=1.0,GKIINFM(11)=1.0,GKIINFM(12)=1.0




*******************
*** DATA TABLES ***
*******************
TABLE,FXIMP,2
0.0,999.0
0.0,0.0
TABLE,FYIMP,2
0.0,999.0
0.0,0.0
TABLE,FZIMP,2
```

```
0.0,999.0
0.0,0.0
TABLE,IBRT,2
0.0,0.150
0.0333,0.0333
TABLE,JBUH,3
0.0,0.005,0.009
0.0,0.00001,1.0
TABLE,JDH,2
0.0,0.007
1.0,0.0
TABLE,RJBUH,3
0.0,0.00001,1.0
0.0,0.005,0.009
TABLE,RJDH,2
0.0,1.0
0.007,0.0
```

```
******************************
*** ACTIVE NOMINAL STATES ***
******************************
ALL STATES
INT CONTROL
S2 INFM(9)=0.0
S2 INFM(10)=0.0
S2 INFM(11)=0.0
S2 INFM(12)=0.0
```

```
***************************
*** INITIAL CONDITIONS ***
***************************
INITIAL CONDITIONS
R2 PM(1)=21898761.0
R2 PM(2)=0.0
R2 PM(3)=0.0
RD2PM(1)=-8011.1
RD2PM(2)=-22010.0
RD2PM(3)=0.0
```

```
Q  VA(1)=0.09960
Q  VA(2)=0.56486
Q  VA(3)=-0.80671
Q  VA(4)=-0.14224
```

```
*********************************************
*** PLOT PARAMETERS AND OUTPUT CONTROLS ***
*********************************************
TMAX=3.0,INT MODE=3,TINC=0.0001,OUTRATE=100.0,PRATE=15.0
PRINT CONTROL=8
DISPLAY1
TIME,VS,PHIE
THETAE,VS,TIME
PSIE,VS,TIME
DISPLAY2
P,VS,TIME
Q,VS,TIME
R,VS,TIME
DISPLAY3
VXERR,VS,TIME
VYERR,VS,TIME
VZERR,VS,TIME
DISPLAY4
EPHIE,VS,TIME
ETHETAE,VS,TIME
EPSIE,VS,TIME
DISPLAY5
PESTM,VS,TIME
QESTM,VS,TIME
RESTM,VS,TIME
DISPLAY6
EVXERR,VS,TIME
EVYERR,VS,TIME
EVZERR,VS,TIME
DISPLAY7
THR1,VS,TIME
THR2,VS,TIME
THR3,VS,TIME
THR4,VS,TIME
DISPLAY8
THR5,VS,TIME
THR6,VS,TIME
THR7,VS,TIME
THR8,VS,TIME
DISPLAY9
POWER,VS,TIME
FUELTOT,VS,TIME
```

```
GGP PLOTS
*PRINTER PLOTS




*************************************************
*** ALL PARAMETERS BELOW HERE ARE RUN DEPENDENT ***
*************************************************
PARAMETER VALUES
CVC(1)=0.5,CVC(2)=0.5,CVC(3)=0.5
CEA(1)=159.0,CEA(2)=1.0,CEA(3)=-109.0
ENGFLAG=1.0
SAFLAG=1.0
CALC XIC
SIMULATE
```

APPENDIX C

```
      SUBROUTINE DCFQUAT(Q,DC)
C     CALCULATE DIRECTION COSINE MATRIX FROM QUATERNION
C     DESIGNED BY:      R. G. BORST        7-7-84
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION Q(4),DC(3,3)
      QQ1=Q(1)*Q(1)
      QQ2=Q(2)*Q(2)
      QQ3=Q(3)*Q(3)
      QQ4=Q(4)*Q(4)
      DC(1,1)=QQ1+QQ2-QQ3-QQ4
      DC(2,2)=QQ1+QQ3-QQ2-QQ4
      DC(3,3)=QQ1+QQ4-QQ2-QQ3
      QQ1=Q(1)*Q(4)
      QQ2=Q(2)*Q(3)
      DC(1,2)=2*(QQ2+QQ1)
      DC(2,1)=2*(QQ2-QQ1)
      QQ1=Q(2)*Q(4)
      QQ2=Q(1)*Q(3)
      DC(1,3)=2*(QQ1-QQ2)
C     ASSURE DC(1,3) MAGNITUDE LT 1
      IF(DC(1,3) .LT. -1.0)DC(1,3)=-1.0
      IF(DC(1,3) .GT. 1.0)DC(1,3)=1.0
      DC(3,1)=2*(QQ1+QQ2)
      QQ1=Q(3)*Q(4)
      QQ2=Q(1)*Q(2)
      DC(2,3)=2*(QQ1+QQ2)
      DC(3,2)=2*(QQ1-QQ2)
      RETURN
      END


      SUBROUTINE QUATFEA(EAA,Q)
C     CALCULATE QUATERNION FROM EULER ANGLES
C     DESIGNED BY:      R. G. BORST        7-7-84
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION EAA(3),EA(3),Q(4)
      RPD=0.017453292
      EA(1)=0.5*EAA(1)*RPD
      EA(2)=0.5*EAA(2)*RPD
      EA(3)=0.5*EAA(3)*RPD
      Q(1)=COS(EA(1))*COS(EA(2))*COS(EA(3))
     1+SIN(EA(1))*SIN(EA(2))*SIN(EA(3))
      Q(2)=SIN(EA(1))*COS(EA(2))*COS(EA(3))
     1-COS(EA(1))*SIN(EA(2))*SIN(EA(3))
      Q(3)=COS(EA(1))*SIN(EA(2))*COS(EA(3))
     1+SIN(EA(1))*COS(EA(2))*SIN(EA(3))
      Q(4)=COS(EA(1))*COS(EA(2))*SIN(EA(3))
     1-SIN(EA(1))*SIN(EA(2))*COS(EA(3))
C     NORMALIZE GUIDANCE QUATERNION
      XNORM=Q(1)**2+Q(2)**2+Q(3)**2+Q(4)**2
      Q(1)=Q(1)/SQRT(XNORM)
      Q(2)=Q(2)/SQRT(XNORM)
      Q(3)=Q(3)/SQRT(XNORM)
      Q(4)=Q(4)/SQRT(XNORM)
      RETURN
      END



      SUBROUTINE CEFQUAT(Q1,Q2,ALPHA,EA,ERR)
C     CALCULATE CONTROL ERRORS FROM QUATERNIONS
C     DESIGNED BY:      R. G. BORST       7-7-84
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION Q1(4),Q2(4),Q3(4),QCON(4),EA(3),ERR(3)
      RPD=0.017453292
      CALL QCONJ(Q1,QCON)
      CALL QPROD(QCON,Q2,Q3)
C     CALCULATE ROTATION ANGLE
      ALPHA=2.0*ACOS(Q3(1))
C     CALCULATE EULER AXIS
      EA(1)=Q3(2)/SIN(0.5*ALPHA)
      EA(2)=Q3(3)/SIN(0.5*ALPHA)
      EA(3)=Q3(4)/SIN(0.5*ALPHA)
C     CALCULATE ATTITUDE CONTROL ERRORS
      ERR(1)=2.0*EA(1)*SIN(0.5*ALPHA)/RPD
      ERR(2)=2.0*EA(2)*SIN(0.5*ALPHA)/RPD
      ERR(3)=2.0*EA(3)*SIN(0.5*ALPHA)/RPD
      ALPHA=ALPHA/RPD
      RETURN
      END



      SUBROUTINE STAT(MEAN,MAX,MIN,SDEV,RMS,N,INPUT,ICFLAG)
C     STATISTICAL ANALYSIS
C     DESIGNED BY:      R. G. BORST       7-7-84
      REAL MEAN,MAX,MIN,N,INPUT,ICFLAG
      IF(ICFLAG .EQ. 0.0)GO TO 100
C     INITIAL CONDITIONS
      N=0.0
      MEAN=0.0
      MAX=INPUT
      MIN=INPUT
      SDEV=0.0
  100 CONTINUE
      ARG=N*(SDEV**2+MEAN**2)+INPUT**2
C     MEAN
      MEAN=(N*MEAN+INPUT)/(N+1.0)
C     STANDARD DEVIATION
      SDEV=SQRT(ABS(ARG/(N+1.0)-MEAN**2))
C     NUMBER OF SAMPLES
      N=N+1.0
C     ROOT-MEAN-SQUARE
      RMS=SQRT(ARG/N)
C     MINIMUM VALUE
```

```
      IF(INPUT .LT. MIN)MIN=INPUT
C     MAXIMUM VALUE
      IF(INPUT .GT. MAX)MAX=INPUT
      RETURN
      END




      SUBROUTINE RNG(RANDG,SEED,SIGMA,MEAN)
C     GENERATE A NORMALLY DISTRIBUTED RANDOM NUMBER
C     DESIGNED BY:      R. G. BORST       7-7-84
      IMPLICIT DOUBLE PRECISION(A-H,O-Z)
      REAL MEAN
      SUM=0.0
      DO 10 I=1,12
      X=(SEED+3.14159265358979)**5
      SEED=X-AINT(X)
      SUM=SUM+SEED
   10 CONTINUE
      RANDG=(SUM-6)*SIGMA+MEAN
      RETURN
      END




      SUBROUTINE PM(R,RDOT,IR,RD,RDDOT,IRD,TI2,DA2,RADIUS,F,AMASS,
     1 ALAT,ALON,ALT,TIM,DAT,VEL,AZI,GAM,ENG)
C  VERSION 1.                                 REVISED  DEC 12 1977
C  PURPOSE*  CALCULATE THE POSITION AND VELOCITY OF A POINT MASS
C            IN RESPONSE TO EXTERNAL FORCES, F, WHEN IN A SHPERICAL
C            EARTH GRAVITY FIELD.
C  CALL SEQUENCE
C    OUTPUTS*  R,RDOT,IR(3)   POSITION VECTOR,INERTIAL AXES,FT OR METERS
C              RD,RDDOT,IRD(3) - VEL VECT, INERTIAL AXES,FT/SEC OR M/SEC
C              TI2             - INITIAL TIME, HOURS
C              DA2             - INITIAL DATE, JULIAN DAYS
C              RADIUS          - EARTH RADIUS, FEET OR METERS
C    INPUTS*   F(3)            - EXT. FORCE VECTOR,INERTIAL AXES,LBS
C              AMASS           - POINT MASS, SLUGS OR KILOGRAMS
C              ALAT            - INITIAL LATITUDE,DEGREES
C              ALON            - INITIAL LONGITUDE, DEGREES
C              ALT             - INITIAL ALTITUDE,FEET OR METERS
C              TIM             - INITIAL TIME, HOURS
C              DAT             - INITIAL DATE, JULIAN DAYS
C              VEL             - INITIAL VELOCITY, FT/SEC OR M/SEC
C              AZI             - INITIAL AZIMUTH ANGLE, DEGREES
C              GAM             - INITIAL VERTICAL FLIGHT PATH ANGLE,DEG
C              ENG             - ENGLISH UNITS FLAG  O = METRIC
C
C  DESIGNED BY* J.D. BURROUGHS                  NOV 1977
C ** MODIFIED BY JDB  ADD METRIC UNITS AND ADD OUTPUTS FOR PO COMPONENT
C ** MODIFIED BY JDB  CORRECT CONSTANTS
```

```
      IMPLICIT DOUBLE PRECISION(A-H,O-Z)
      COMMON/CICCAL/ICCAL/CTIME/TIME
      DIMENSION R(3),RDOT(3),IR(3),RD(3),RDDOT(3),IRD(3),F(3)
      DIMENSION T1(3,3),V(3),V1(3)
      DATA RADIE,RPD/20925663.,.017453292519943/,V/3*0./,
     1  GRAVE/-1.4076459E16/,GRAVM/-3.986009E14/,RADIM/6378142./
C =========  PROVIDE DEFAULT VAULES FOR PARAMETERS
      IF(DAT.EQ..99999)DAT=80.
      IF(TIM.EQ..99999)TIM=12.
      IF(ALAT.EQ..99999)ALAT=0.
      IF(ALON.EQ..99999)ALON=0.
      IF(GAM.EQ..99999)GAM=0.
      IF(AZI.EQ..99999)AZI=0.
      TI2=TIM
      DA2=DAT
      GRAVC=GRAVE
      RADIUS=RADIE
      IF(ENG.NE.0.) GO TO 20
C =========  SWITCH TO METRIC UNITS
      GRAVC=GRAVM
      RADIUS=RADIM
C ========  TEST FOR INITIAL CONDITION CALCULATIONS
20    IF(ICCAL.NE.1)GO TO 100
C ==========  CALC INITIAL INERTIAL POSITION FROM LATITUDE,
C             LONGITUDE,ALTITUDE,DATE, AND TIME
      RLAT=ALAT*RPD
      COSLAT=COS(RLAT)*(RADIUS+ALT)
      SINLAT=SIN(RLAT)
      ANG=(15.*(TIM+TIME/3600.-12.)+.98563*(DAT-80.)+ALON)*RPD
      COSLON=COS(ANG)
      SINLON=SIN(ANG)
      R(1)=COSLAT*COSLON
      R(2)=COSLAT*SINLON
      R(3)=SINLAT*(RADIUS+ALT)
C ========  CALC INITIAL INERTIAL VELOCITY FROM LATITUDE,
C           LONGITUDE, DATE, TIME, VELOCITY, AND FLIGHT
C           PATH ANGLES RELATIVE TO TO LOCAL HORIZONTAL.
C --------  TRANSFORMATION THROUGH VERTICAL FLIGHT PATH ANGLE
      V(1)=VEL
      RGAM=-GAM*RPD
      CALL TRANF(T1,RGAM,2)
      CALL MATMPY(V1,T1,V,3,3,1)
C --------  TRANSFORMATION THROUGH HORIZONTAL FLIGHT PATH ANGLE
      RAZI=-AZI*RPD
      CALL TRANF(T1,RAZI,3)
      CALL MATMPY(RD,T1,V1,3,3,1)
C -------  TRANSFORMATION THROUGH LATITUDE ANGLE
      RLAT=(90.+ALAT)*RPD
      CALL TRANF(T1,RLAT,2)
      CALL MATMPY(V1,T1,RD,3,3,1)
C --------  TRANSFORMATION THROUGH DATE - TIME - LONGITUDE ANGLES
      CALL TRANF(T1,-ANG,3)
      CALL MATMPY(RD,T1,V1,3,3,1)
100   CONTINUE
C ========  CALCULATE GRAVITY ACCEL COEFF.
      CALL DOTPRD(R2,R,R,3)
```

```
C60    GMR3=GRAVC/R2/SQRT(R2)
       GMR3=GRAVC/R2/DSQRT(R2)
C ========  EQUATIONS OF MOTION IN INERTIAL AXES
       DO 1000 I=1,3
       IF(IR(I).NE.0)RDOT(I)=RD(I)
C      REMOVE GRAVITY TERM             R. G. BORST     3-16-84
C      IF(IRD(I).NE.0)RDDOT(I)=F(I)/AMASS+GMR3*R(I)
       IF(IRD(I).NE.0)RDDOT(I)=F(I)/AMASS
1000   CONTINUE
       RETURN
       END
```