2
TM X

MSC - 04217
REVISION B

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

PROJECT/SPACE SHUTTLE

SPACE SHUTTLE GUIDANCE, NAVIGATION AND CONTROL DESIGN EQUATIONS

VOLUME III
ORBITAL OPERATIONS

REVISED
DECEMBER 1, 1971

(NASA-TM-X-68368) SPACE SHUTTLE GUIDANCE,
NAVIGATION AND CONTROL DESIGN EQUATIONS.
VOLUME 3: ORBITAL OPERATIONS (NASA)
1 Dec. 1971 550 p
N72-25873
CSCL 22B
Unclas
G3/31 31261

SYSTEMS ANALYSIS BRANCH
GUIDANCE AND CONTROL DIVISION
MANNED SPACECRAFT CENTER
HOUSTON, TEXAS

Sgt · 62304

MSC - 04217
REVISION B

NASA SPACE SHUTTLE PROGRAM WORKING PAPER

SPACE SHUTTLE GUIDANCE, NAVIGATION AND CONTROL DESIGN EQUATIONS

VOLUME III
ORBITAL OPERATIONS

Revised
December 1, 1971

[illegible]

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
MANNED SPACECRAFT CENTER
HOUSTON, TEXAS

Prepared by
Systems Analysis Branch
Guidance and Control Division

K. J. Cox

K. J. Cox, Chief
Systems Analysis Branch

Authorized for
Distribution William C. Bradford
for Maxime A. Faget
Director of Engineering and Development

PRECEDING PAGE BLANK NOT FILMED

TABLE OF CONTENTS

PAGE

1. PURPOSE . . . . 1-1 *

2. SCOPE . . . . 2-1 *

3. APPLICABILITY . . . . 3-1 *

VOLUME I

4. DEFINITION OF TERMS 4-1

5. REFERENCE DOCUMENTS . . . . 5-1 *

6. GN&C SYSTEM DESCRIPTION . . . . 6-1 *

7. GN&C SOFTWARE FUNCTIONAL REQUIREMENTS . . . . 7-1 *

8. COORDINATE SYSTEMS AND TRANSFORMATIONS . . . . 8-1 *

VOLUME II - PREFLIGHT THROUGH ORBIT INSERTION

9. DESCRIPTION OF EQUATIONS 9-1

9.1 Preflight 9.1-1

9.1.1 GN&C System Initialization, Monitor, Test and Checkout . . . . TBD

9.1.2 Prelaunch Targeting . . . . TBD

9.1.3 Sensor Calibration and Alignment . . . . . . TBD

9.2 Boost Phase Program 9.2-1

9.2.1 Rapid Real-Time State Advancement During Specific Force Sensing 9.2-4

9.2.2 Booster Control . . . . 9.2-15 *

9.3 Separation 9.3-1

9.3.1 Attitude Control . . . . TBD

9.4 Orbit Insertion Program 9.4-1

9.4.1 Navigation (same as 9.2.1) 9.4-1

9.4.2 Powered Ascent Guidance 9.4-3

9.4.3 Thrust Vector Control . . . . 9.4-26 *

TABLE OF CONTENTS (CONTINUED)

PAGE

9.5 Ascent Aborts 9.5-1

9.5.1 Abort Decision and Mode Determination . . . . TBD

9.5.2 Abort Targeting . . . . TBD

9.5.3 Abort Guidance . . . . TBD

9.5.4 Navigation . . . . TBD

9.5.5 Attitude Control . . . . TBD

VOLUME III - ORBITAL OPERATIONS

9.6 Orbital Coast 9.6-1

9.6.1 Navigation 9.6.1

9.6.1.1 Conic State Extrapolation 9.6-3

9.6.1.2 Precision State and Filter Weighting Matrix Extrapolation . . . . 9.6-32 *

9.6.1.3 Orbital Navigation using Navigation Sensors . . . . 9.6-58 *

9.6.2 Sensor Calibration and Alignment 9.6-92

9.6.2.1 IRU Alignment . . . . 9.6-93 *

9.6.3 Orbital Coast Attitude Control . . . . TBD

9.7 Orbital Powered Flight 9.7-1

9.7.1 Required Velocity Determination 9.7-2

9.7.1.1 Conic . . . . 9.7-3 *

9.7.1.2 Precision . . . . 9.7-49 *

9.7.2 Navigation (same as 9.2.1) 9.7-67

9.7.3 Powered Flight Guidance . . . . 9.7-68 *

9.7.4 Thrust Vector Control 9.7-110

9.7.4.1 Fixed Engines . . . . 9.7-111 *

9.7.4.2 Gimballed Engines . . . . 9.7-141 *

9.8 Rendezvous Mission Phase 9.8-1

9.8.1 Targeting 9.8-1

9.8.1.1 Rendezvous Targeting . . . . 9.8-2 *

9.8.1.2 Braking . . . . 9.8-56 *

TABLE OF CONTENTS (CONTINUED)

PAGE

9.8.2 Relative State Updating 9.8-87
9.8.3 Rendezvous Guidance (same as 9.7.3). 9.8-129
9.8.4 Rendezvous Attitude Control . . . . TBD
9.9 Station Keeping Mission Phase 9.9-1
9.9.1 Relative State Estimation . . . . TBD
9.9.2 Station Keeping Guidance . . . . TBD
9.9.3 Station Keeping Attitude Control . . . . . . TBD
9.10 Docking and Undocking 9.10-1
9.10.1 Docking and Undocking Navigation 9.10-5
9.10.2 Automatic Docking Control Law 9.10-32
9.11 Docked Operations (TBD). 9.11-1

VOLUME IV - DEORBIT AND ATMOSPHERIC OPERATIONS

9.12 Deorbit and Entry 9.12-1
9.12.1 Deorbit Targeting . . . . 9.12-3 *
9.12.2 Entry Navigation . . . . TBD
9.12.3 Entry Guidance . . . . 9.12-35 *
9.12.4 Entry Autopilot . . . . 9.12-81 *
9.13 Transition 9.13-1
9.13.1 Transition Attitude Control . . . . 9.13-2 *
9.14 Cruise and Ferry Cruise 9.14-1
9.14.1 Navigation . . . . TBD
9.14.2 Guidance . . . . TBD
9.14.3 Cruise Autopilot 9.14-1
9.14.3.1 Manual Modes . . . . 9.14-3 *
9.14.3.2 Digital Control . . . . 9.14-11 *
9.15 Approach and Landing 9.15-1
9.15.1 Navigation . . . . TBD
9.15.2 Guidance 9.15-1

TABLE OF CONTENTS (CONTINUED)

PAGE

9.15.2.1 Approach . . . . 9.15-3 *
9.15.2.2 Final Approach 9.15-59*
9.15.3 Control . . . . TBD
9.16 Horizontal Takeoff . . . . TBD
9.17 Communications and Pointing . . . . TBD
9.18 Failure Detection . . . . TBD

VOLUME V - FLOW DIAGRAMS

10. Flow Diagrams . . . . TBD

VOLUME VI - CONSTANTS AND KEYBOARD
ACCESSIBLE PARAMETERS

11. SUMMARY OF CONSTANTS . . . . TBD

12. REQUIRED KEYBOARD ACCESSIBLE PARAMETERS . . . . TBD

1. PURPOSE

The purpose of this document is to specify the equations necessary to perform the guidance, navigation and control onboard computation functions for the space shuttle orbiter vehicle. This equations document will provide as comprehensive a set of equations as possible from which modules may be chosen to develop Part I Specifications for particular vehicles, computers and missions. This document is expected to be the source of any equations used to develop software for hardware/software feasibility testing, for ground-based simulations or flight test demonstrations.

2. SCOPE

This document defines a baseline set of equations which fulfill the computation requirements for guidance, navigation and control of the space shuttle orbiter vehicle. All shuttle mission phases are covered from Prelaunch through Landing/Rollout. The spacecraft flight mode and the aircraft flight mode are addressed. Equations are included for the Mark I systems and Mark II systems through the all-up shuttle configuation. Control of the booster during launch is covered. The baseline equations may be implemented in a single GN&C computer or may be distributed among several subsystem computers, depending upon the outcome of centralization/decentralization deliberations currently in progress.

3. APPLICABILITY

This document is applicable to the guidance, navigation and control (GN&C) computation functions for the space shuttle orbiter vehicle. It specifies a set of baseline design equations which may be used for the shuttle program software specification and hardware sizing. It defines the baseline equations for MSC G&CD hardware/software simulation.

PRECEDING PAGE BLANK NOT FILMED

FOREWORD

This second publication of the Space Shuttle GN&C Design Equation Document contains baseline equations for approximately fifty percent of the GN&C computation requirements as specified in the GN&C S/W Functional Requirements Document (MSC-03690 Rev. B). This document supercedes the original MSC-04217 and the subsequently published revision. Additions or corrections to this document since its original publication are indicated in the Table of Contents by asterisks in the margin.

It is planned to republish this document in a new revision in approximately four months time. At that time it is anticipated that equations will be available for virtually all requirements. The new revision will be issued with format changes intended to stress interdependency of related submittals and to eliminate duplication to the greatest degree practicable.

This issue has been modified to reflect the shuttle-structure and avionics-configuration changes which have occurred subsequent to the first issue. A significant change is that orbiter control of the booster has been added as a requirement. Decentralization of the computations and allocation to subsystems is the current trend with the MARK I & MARK II shuttle configurations. The computation requirements for shuttle vehicles and missions may be much less than those allowed for in this document. However, since the configurations are very fluid at this state in the shuttle development, the approach adopted in this document is to include as complete a set of design equations as possible to cover reasonable possibilities. Therefore, subsets of equations may be extracted from this document to form specifications for specific vehicles, computers and missions.

The GN&C Design Equations document is the result of the efforts of many people from NASA and support contractors. The list is too long to credit all contributors; however, contractors which made direct contributions to the document are as follows:

a. TRW Systems Group, Inc., Houston Operations
b. MIT/Charles Stark Draper Laboratory
c. Lockheed Electronics Co., Inc., Houston Aerospace Systems Division
d. The Boeing Co., Houston, Texas

The equations are reviewed by the GN&C Formulation and Implementation Panel and their comments included on submittal forms where appropriate. The names of equation submitters are included on the submittal sheet in each section. Comments on the submittals should be referred to the individual submitter or to the responsible NASA engineer. General comments on the document or proposed submittals should be referred to the System Analysis Branch, Guidance and Control Division.

9. DESCRIPTIONS OF EQUATIONS

The detailed equations for the GN&C functions are defined in this section. The organization of this section is tentative and will be modified so as to present the equations as they are designed in as clear a fashion as possible. As an introduction to each major subsection (usually a mission phase), the general GN&C software functions to be implemented will be identified and, where appropriate, a conceptual discussion and top level flow of the computations, inputs and outputs will be included in order to understand and summarize what is to be covered. This should be an order of magnitude less detailed than the flow diagrams of the equations which come later.

A GN&C Equation Submittal sheet will introduce each of the GN&C equation submittals and summarize the GN&C functions, and identify the source and NASA contact for each.

The detailed data to be presented for each GN&C function within each of the major subsections (usually a mission phase) is summarized below. Although items 6 through 10 are to be referenced only in the equations document, they are required submittals before the equations can be approved and finalized for flight software development.

1. Functional Requirements

   The specific functional requirements (from the GN&C Software Functional Requirements Document) which are satisfied by the equations should be identified.

2. Functional Diagram

   A brief functional explanation and description of the overall concept and approach. A functional block diagram should be used where clarity is enhanced. Inputs, outputs, and interfaces will be provided.

3. Equations and Flows

   Detailed equations and a descriptive text which guides the reader through the flows of Section 10 should be provided. The minimum frequency of the computations shall be specified and rationale given or referenced.

4. Coordinate System

   The coordinate systems used shall be defined.

5. Constants/Variables Summary

   Constants and variables shall be summarized in tabular form with the following information:

   a. Variables/constants symbols and definitions
   b. Units
   c. Allowable quantization
   d. Range of values

6. FORTRAN Coding

   The FORTRAN coding of the function for verification using the Space Shuttle Flight Simulation (SSFS) will be referenced.

7. Simulation

   The SSFS specifications, description and user's guide used to verify each GN&C function will be referenced.

8. Testing

   Test plans and test results will be referenced.

9. Derivation

   The mathematical derivation of the equations including all mathematical assumptions shall be referenced.

10. Assumptions

    The following will be referenced:

    a. Avionics baseline system assumed
    b. Reference missions assumed
    c. Vehicle mass properties assumed
    d. Propulsion models assumed
    e. Environment models assumed
    f. Error models assumed

The major subsections of this section are identified and partially expanded in the following.

9.6 ORBITAL COAST

The following GN&C software functions are envisioned for the orbital coast phase:

Sensor Alignment and Calibration

1. Perform automatic calibration of sensors and compute compensation values during coasting orbital flight.

2. Perform automatic sensor pointing and alignment during coasting orbital flight.

Orbit Navigation

3. Advance inertial vector with conic solutions from an initial state to a final state as a function of time or anomaly.

4. Augment conic state advancement with numerical integration to account for complex gravity potential models.

5. Reduce uncertainties in inertial state by accepting and processing data from navigation sensors (ground beacons, radar altimeter).

Attitude Control

6. Maintain attitude-hold about a desired orientation.

7. Provide attitude rate-hold about a desired rate for orbital rate control, station keeping, passive thermal control or other constant-rate maneuvers.

8. Provide semi-automatic control by initializing attitude hold following manual maneuvers.

9. Implement minimum-impulse jet firings when required by the autopilot or selected by the crew for manual control.

10. Maintain attitude for target visibility at crew and radar locations during the coast periods of rendezvous, station keeping and docking approach.

9.6.1 Orbital Navigation

SPACE SHUTTLE

GN&C SOFTWARE EQUATION SUBMITTAL

Software Equation Section: Conic State Extrapolation Submittal No. 6

Function: Advance inertial state with conic solutions

Module No. ON2 Function No. 1 (MSC 03690)

Submitted by: W. M. Robertson Co. MIT No. 3-71

Date: Feb 1971

NASA Contact: J. Suddath Organization: GCD

Approved by Panel III: K. Cox K.J. Cox Date: 3/10/71

Summary Description: Provides the capability to advance a geocentric inertial state as a function of time or true anomaly. The extrapolation is done analytically assuming Keplerian motion.

Shuttle Configuration: These equations are independent of Shuttle configuration.

Comments:

(Design Status) ____________________

(Verification Status) ____________________

Panel Comments:

9.6.1.1 Conic State Extrapolation

1. INTRODUCTION

The Conic State Extrapolation Routine provides the capability to conically extrapolate any spacecraft inertial state vector either backwards or forwards as a function of time or as a function of transfer angle. It is merely the coded form of two versions of the analytic solution of the two-body differential equations of motion of the spacecraft center of mass. Because of its relatively fast computation speed and moderate accuracy, it serves as a preliminary navigation tool and as a method of obtaining quick solutions for targeting and guidance functions. More accurate (but slower) results are provided by the Precision State Extrapolation Routine.

9.6.1.1 Conic State Extrapolation (continued)

NOMENCLATURE

a	Semi-major axis of conic
c_1	First conic parameter ($\underline{r}_0 \cdot \underline{v}_0) / \mu_E)$
c_2	Second conic parameter ($r_0 v_0^2 / \mu_E - 1$)
c_3	Third conic parameter ($r_0 v_0^2 / \mu_E$)
$C(\xi)$	Power series in ξ defined in text
E	Eccentric anomaly
f	True anomaly
H	Hyperbolic analog of eccentric anomaly
i	Counter
p	Semilatus rectum of conic
p_N	Normalized semilatus rectum (p / r_0)
P	Period of conic orbit
r_0	Magnitude of $\underline{r}_0$
$\underline{r}_0$	Inertial position vector corresponding to initial time t_0
r	Magnitude of $\underline{r}(t)$
$\underline{r}(t)$	Inertial position vector corresponding to time t
s	Switch used in Secant Iterator to determine whether secant method or offsetting will be performed

$S(\xi)$ Power series in ξ defined in text

t Final time (end of time interval through which an extrapolation is made)

t_0 Initial time (beginning of time interval through which an extrapolation is to be made)

$(t - t_0)$ Specified transfer time interval

$(t - t_0)_c$ Value of the transfer time interval calculated in the Universal Kepler Equation as a function of x and the conic parameters

$(t - t_0)'_c$ Previous value of $(t - t_0)_c$

$(t - t_0)_c^{(i)}$ The "i-th" value of the transfer time interval calculated in the Universal Kepler Equation as a function of the "i-th" value x_i of x and the conic parameters

t_{ERR} Difference between specified time interval and that calculated by Universal Kepler Equation

v_0 Magnitude of $\underline{v}_0$

$\underline{v}_0$ Inertial velocity vector corresponding to initial time t_0

$\underline{v}(t)$ Inertial velocity vector corresponding to time t

x Universal eccentric anomaly difference (independent variable in Kepler iteration scheme)

x' Previous value of x

x_c Value of x to which the Kepler iteration scheme converged

x'_c Previous value of x_c

x_i	The "i-th" value of x
x_{min}	Lower bound on x
x_{max}	Upper bound on x
α_0	Reciprocal of semi-major axis at initial point $\underline{r}_0$
α_N	Normalized semi-major axis reciprocal (αr_0)
γ_0	Angle from $\underline{r}_0$ to $\underline{v}_0$
Δt_{max}	Maximum time interval which can be used in computer due to scaling limitations
Δx	Increment in x
ϵ_t	Relative convergence tolerance factor on transfer time interval
ϵ_x	Convergence tolerance on independent variable x
θ	Transfer angle (true anomaly increment)
μ_E	Gravitational parameter of the earth
ξ	Product of α_0 and square of x
$\chi_0, \chi_1, \chi_2, \chi_3$	Coefficients of power series inversion of Universal Kepler Equation
$\underline{1}_{r0}$	Unit vector in direction of $\underline{r}_0$
$\underline{1}_{v0}$	Unit vector in direction of $\underline{v}_0$

2. FUNCTIONAL FLOW DIAGRAM

The Conic State Extrapolation Routine basically consists of two parts - one for extrapolating in time and one for extrapolating in transfer angle. Several portions of the formulation are, however, common to the two parts, and may be arranged as subroutines on a computer.

2.1 Conic State Extrapolation As A Function Of Time (Kepler Routine)

This routine involves a single loop iterative procedure, and hence is organized in three sections: initialization, iteration, and final computations, as shown in Fig. 1. The variable "x" is the independent variable in the iteration procedure. For a given initial state, the variable "x" measures the amount of transfer along the extrapolated trajectory. The transfer time interval and the extrapolated state vector are very conveniently expressed in terms of "x". In the iteration procedure, "x" is adjusted until the transfer time interval calculated from it agrees with the specified transfer time interval (to within a certain tolerance). Then the extrapolated state vector is calculated from this particular value of "x".

2.2 Conic State Extrapolation As A Function Of Transfer Angle (Theta Routine)

This routine makes a direct calculation (i. e. does not have an iteration scheme), as shown in Fig. 2. Again, the extrapolated state vector is calculated from the parameter "x". The value of "x" however, is obtained from a direct computation in terms of the conic parameters and the transfer angle θ. It is not necessary to iterate to determine "x", as was the case in the Kepler Routine.

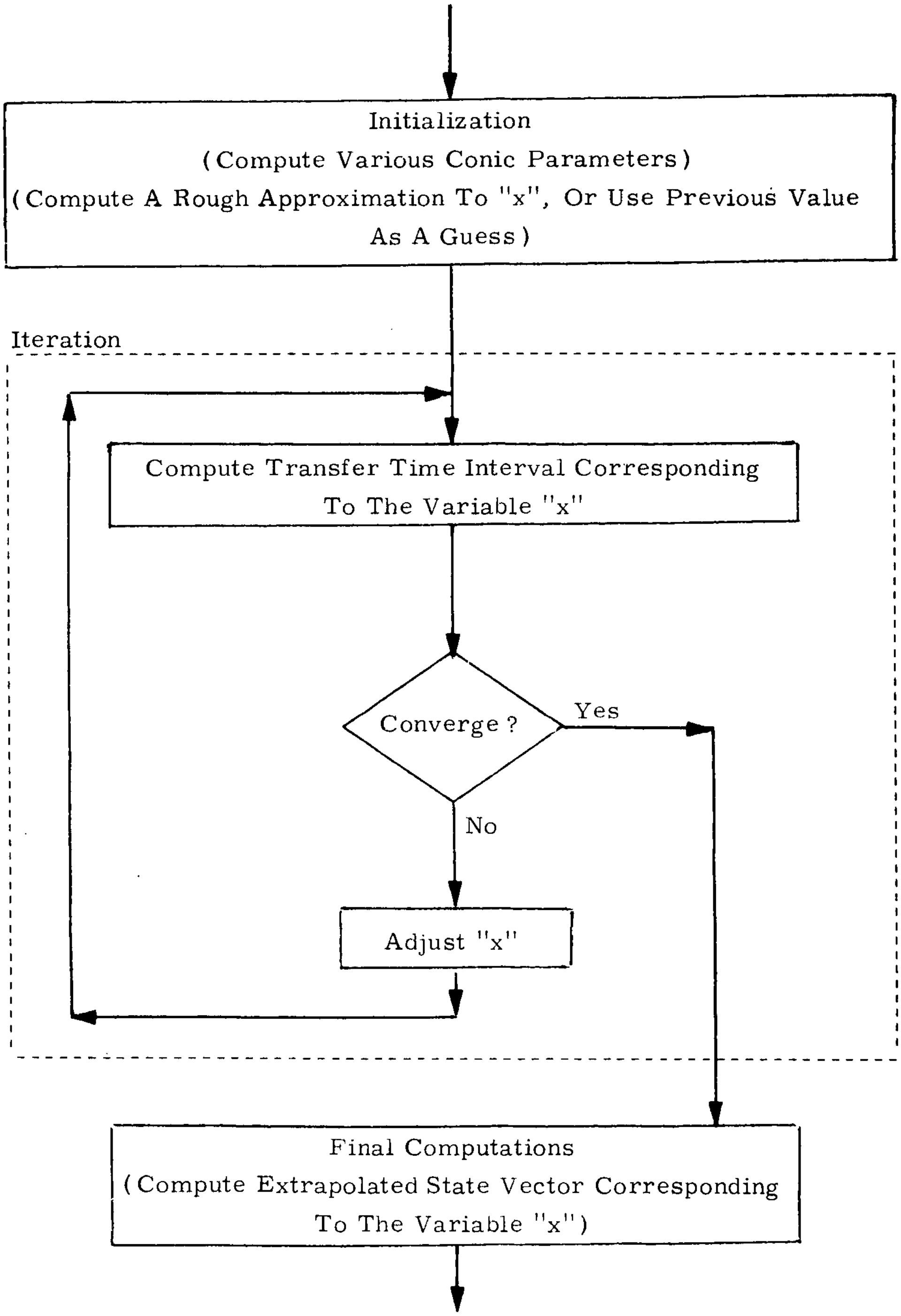

Initialization
(Compute Various Conic Parameters)
(Compute A Rough Approximation To "x", Or Use Previous Value As A Guess)
Iteration
Compute Transfer Time Interval Corresponding To The Variable "x"
Converge ?
Yes
No
Adjust "x"
Final Computations
(Compute Extrapolated State Vector Corresponding To The Variable "x")

Figure 1 KEPLER ROUTINE FUNCTIONAL FLOW DIAGRAM

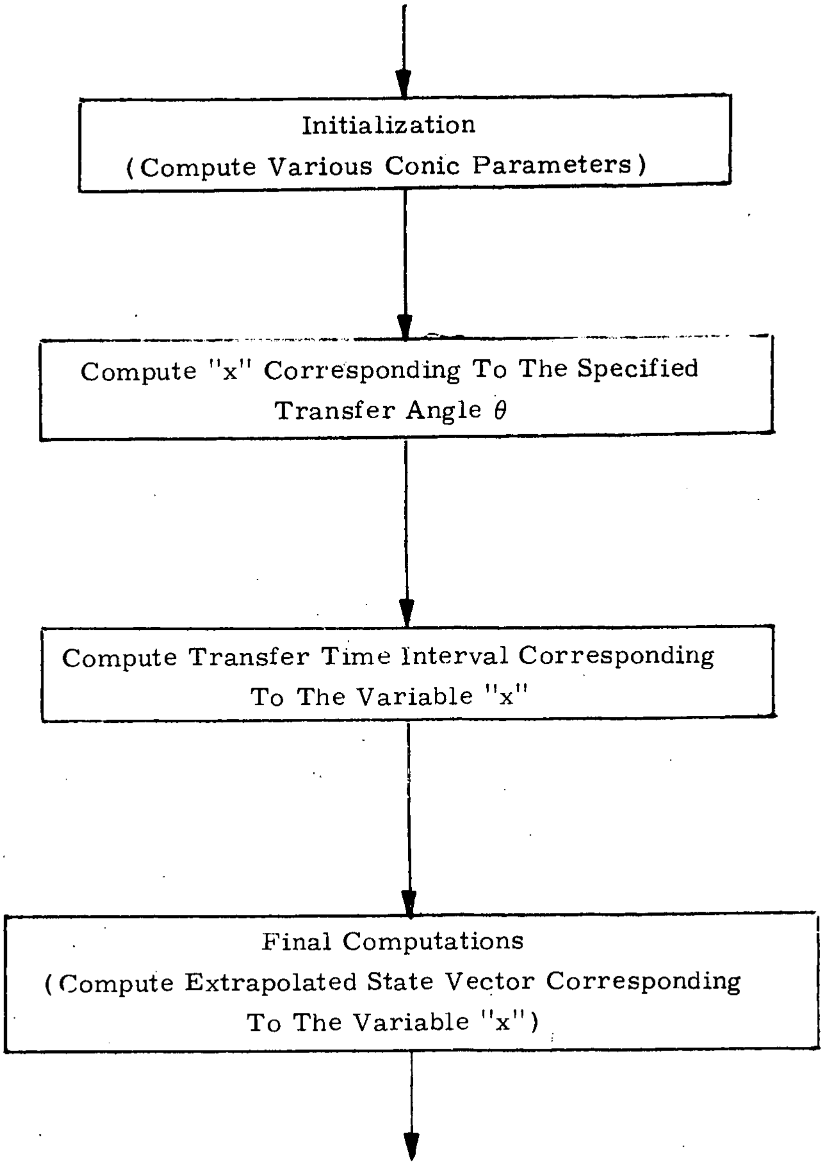

Initialization
(Compute Various Conic Parameters)
Compute "x" Corresponding To The Specified Transfer Angle θ
Compute Transfer Time Interval Corresponding To The Variable "x"
Final Computations
(Compute Extrapolated State Vector Corresponding To The Variable "x")

Figure 2 THETA ROUTINE FUNCTIONAL FLOW DIAGRAM

3. ROUTINE INPUT-OUTPUT

The Conic State Extrapolation Routine has only one system parameter input: the gravitational parameter of the earth. Its principal real-time inputs are the inertial state vector which is to be extrapolated and the transfer time interval or transfer angle through which the extrapolation is to be made. Several optional secondary inputs may be supplied in the transfer time case in order to speed the computation. The principal real-time output of both cases is the extrapolated inertial state vector.

3.1 Conic State Extrapolation As A Function of Transfer Time Interval (Kepler Routine)

Input Parameters

System

μ_E : Gravitational parameter of the earth (Product of earth's mass and universal gravitational constant).

Real-Time (Required)

$(\underline{r}_0, \underline{v}_0)$: Inertial state vector which is to be extrapolated (corresponds to time t_0).

$(t - t_0)$: Transfer time interval through which the extrapolation is to be made.

Real-Time (Optional)

x : Guess of independent variable corresponding to solution in Kepler iteration scheme. (Used to speed convergence).

$(t - t_0)'_c$: Value of dependent variable (the transfer time interval) in the Kepler iteration scheme, which was calculated in the last iteration of the previous call to Kepler.

x'_c : Value of the independent variable in the Kepler iteration scheme, to which the last iteration of the previous call to Kepler had converged.

Output Parameters

$(\underline{r}(t), \underline{v}(t))$: Extrapolated inertial state vector (corresponds to time t).

$(t - t_0)_c$: Value of the dependent variable (the transfer time interval) in the Kepler iteration scheme, which was calculated in the last iteration (should agree closely with $(t - t_0)$).

x_c : Value of the independent variable in the Kepler iteration scheme to which the last iteration converged.

3.2 Conic State Extrapolation As A Function Of Transfer Angle (Theta Routine)

Input Parameters

System

μ_E : Gravitational parameter of the earth (Product of earth's mass and universal gravitational constant).

9.6.1.1 Conic State Extrapolation (continued)

Real-Time

$(\underline{r}_0, \underline{v}_0)$: Inertial state vector which is to be extrapolated.

θ : Transfer angle through which the extrapolation is to be made.

Output Parameters

$(\underline{r}, \underline{v})$: Extrapolated inertial state vector.

$(t - t_0)_c$: Transfer Time Interval corresponding to the conic extrapolation through the transfer angle θ.

4. DESCRIPTION OF EQUATIONS

4.1 Conic State Extrapolation As A Function Of Time (Kepler Routine)

The universal formulation of Stumpff-Herrick-Battin in terms of the universal eccentric anomaly difference is used. This variable, usually denoted by x, is defined by the relations:

$$x = \begin{cases} \sqrt{a}\,(E - E_0) \text{ for ellipse} \\ \sqrt{p}\,(\tan f/2 - \tan f_0/2) \text{ for parabola} \\ \sqrt{-a}\,(H - H_0) \text{ for hyperbola} \end{cases}$$

where a is the semi-major axis, E and H are the eccentric anomaly and its hyperbolic analog, p is the semi-latus rectum and f the true anomaly. The expressions for the transfer time interval $(t - t_0)$ and the extrapolated position and velocity vectors $(\underline{r}, \underline{v})$ in terms of the initial position and velocity vectors $(\underline{r}_0, \underline{v}_0)$ as functions of x are:

(Universal Kepler Equation)

$$(t - t_0) = \frac{1}{\sqrt{\mu_E}} \left[\frac{\underline{r}_0 \cdot \underline{v}_0}{\sqrt{\mu_E}} x^2 C(\alpha_0 x^2) + (1 - r_0 \alpha_0) x^3 S(\alpha_0 x^2) + r_0 x \right]$$

$$\underline{r}(t) = \left[1 - \frac{x^2}{r_0} C(\alpha_0 x^2) \right] \underline{r}_0 + \left[(t - t_0) - \frac{x^3}{\sqrt{\mu_E}} S(\alpha_0 x^2) \right] \underline{v}_0$$

$$\underline{v}(t) = \frac{\sqrt{\mu_E}}{r\, r_0} \left[\alpha_0 x^3 S(\alpha_0 x^2) - x \right] \underline{r}_0 + \left[1 - \frac{x^2}{r} C(\alpha_0 x^2) \right] \underline{v}_0$$

where

$$\alpha_0 = \frac{1}{a_0} = \frac{2}{r_0} - \frac{v_0^2}{\mu_E}$$

and

$$S(\xi) = \frac{1}{3!} - \frac{\xi}{5!} + \frac{\xi^2}{7!} - \ldots$$

$$C(\xi) = \frac{1}{2!} - \frac{\xi}{4!} + \frac{\xi^2}{6!} - \ldots$$

Since the transfer time interval $(t - t_0)$ is given, it is desired to find the x corresponding to it in the Universal Kepler Equation, and then to evaluate the extrapolated state vector $(\underline{r}, \underline{v})$ expression using that value of x. Unfortunately, the Universal Kepler Equation expresses $(t - t_0)$ as a transcendental function of x rather than conversely, and no power series inversion of the equation is known which has good convergence properties for all orbits, so it is necessary to solve the equation iteratively for the variable x.

For this purpose, the secant method (linear inverse interpolation/ extrapolation) is used. It merely finds the increment in the independent variable x which is required in order to adjust the dependent variable $(t - t_0)_c$ to the desired value $(t - t_0)$ based on a linear interpolation/ extrapolation of the last two points calculated on the $(t - t_0)_c$ vs x curve. The method uses the formula

$$(x_{n+1} - x_n) = - \frac{(t - t_0)_c^{(n)} - (t - t_0)}{(t - t_0)_c^{(n)} - (t - t_0)_c^{(n-1)}} (x_n - x_{n-1})$$

where $(t - t_0)_c^{(i)}$ denotes the evaluation of the Universal Kepler Equation using the value x_i. In order to prevent the scheme from taking an increment back into regions in which it is known from past iterations that the solution does not lie, it has been found convenient to establish upper and lower bounds on the independent variable x which are continually reset during the course of the iteration as more and more values of x are found to be too large or too small. In addition, it has also been found expedient to damp by 10% any increment in the independent variable which would (if applied) take the value of the independent variable past a bound.

To start the iteration scheme, some initial guess x_0 of the independent variable is required as well as a previous point $(x_{-1}, (t - t_0)_c^{(-1)})$ on the $(t - t_0)_c$ vs x curve. If no previous point is available the point (0, 0) may be used as it lies on all $(t - t_0)_c$ vs. x curves. The closer the initial guess x_0 is to the value of x corresponding to the solution, the faster the convergence will be. One method of obtaining such a guess x_0 is to use a truncation of the infinite series obtained by direct inversion of the Kepler Equation (expressing x as a power series in $(t - t_0)$). It must be pointed out that this series diverges even for "moderate" transfer time intervals $(t - t_0)$; hence an iterative solution must be used to solve the Kepler equation for x in the general case. A third order truncation of the inversion of the Universal Kepler Equation is:

$$x = \sum_{n=0}^{3} \chi_n (t - t_0)^n$$

where

$$\chi_0 = 0, \quad \chi_1 = \sqrt{\mu_E}/r_0,$$

$$\chi_2 = -\frac{1}{2}\frac{\mu_E}{r_0^3}\left(\frac{\underline{r}_0 \cdot \underline{v}_0}{\sqrt{\mu_E}}\right),$$

$$\chi_3 = \frac{1}{6r_0}\left(\frac{\sqrt{\mu_E}}{r_0}\right)^3 \left[\frac{3}{r_0}\left(\frac{\underline{r}_0 \cdot \underline{v}_0}{\sqrt{\mu_E}}\right)^2 - (1 - r_0\alpha_0)\right],$$

with $\alpha_0 = 2/r_0 - v_0^2/\mu_E$.

4.2 Conic State Extrapolation As A Function of Transfer Angle (Theta Routine)

As with the Kepler Routine, the universal formulation of Stumpff-Herrick-Battin in terms of the universal eccentric anomaly difference x is used in the Theta Routine. A completely analogous iteration scheme could have been formulated with x again as the independent variable and the transfer angle θ as the dependent variable using Marscher's universally valid equation:

$$\cot \frac{\theta}{2} = \frac{r_0 \left[1 - \alpha_0 x^2 S(\alpha_0 x^2) \right]}{\sqrt{p}\, x \; C(\alpha_0 x^2)} + \cot \gamma_0$$

where

$$p = \left(\frac{r_0 v_0}{\sqrt{\mu_E}}\right)^2 \sin^2 \gamma_0$$

and

$$\gamma_0 = \text{angle from } \underline{r}_0 \text{ to } \underline{v}_0 .$$

However, in contrast to the Kepler equation, it is possible to invert the Marscher equation into a power series which can be made to converge as rapidly as desired, by means of which x may be calculated as a universal function of the transfer angle θ. Knowing x, we can directly calculate the transfer time interval $(t - t_0)_c$ and subsequently the extrapolated state vectors using the standard formulae.

The sequence of computations in the inversion of the Marscher Equation is asfollows:

Let

$$p_N = p/r_0, \; \alpha_N = \alpha \, r_0$$

and

9.6.1.1 Conic State Extrapolation (continued)

$$W_1 = \sqrt{p_N}\left(\frac{\sin\theta}{1 - \cos\theta} - \cot\gamma_0\right).$$

If

$$\left|W_1\right| > 1, \text{ let } V_1 = 1.$$

Let

$$W_{n+1} = +\sqrt{W_n^{\,2} + \alpha_N} + \left|W_n\right| \qquad \left(\left|W_1\right| \leq 1\right)$$

or

$$V_{n+1} = +\sqrt{V_n^{\,2} + \alpha_N \left(\left|1/W_1\right|\right)^2} + V_n \qquad \left(\left|W_1\right| > 1\right).$$

Let

$$\omega_n = W_n \qquad \left(\left|W_1\right| \leq 1\right)$$

or

$$1/\omega_n = \left(\left|1/W_1\right|\right)/V_n \qquad \left(\left|W_1\right| > 1\right).$$

Let

$$\Sigma = \frac{2^n}{\omega_n} \sum_{j=0}^{\infty} \frac{(-1)^j}{2j+1} \left(\frac{\alpha_N}{\omega_n^{\,2}}\right)^j$$

where n is an integer ≥ 4. Then

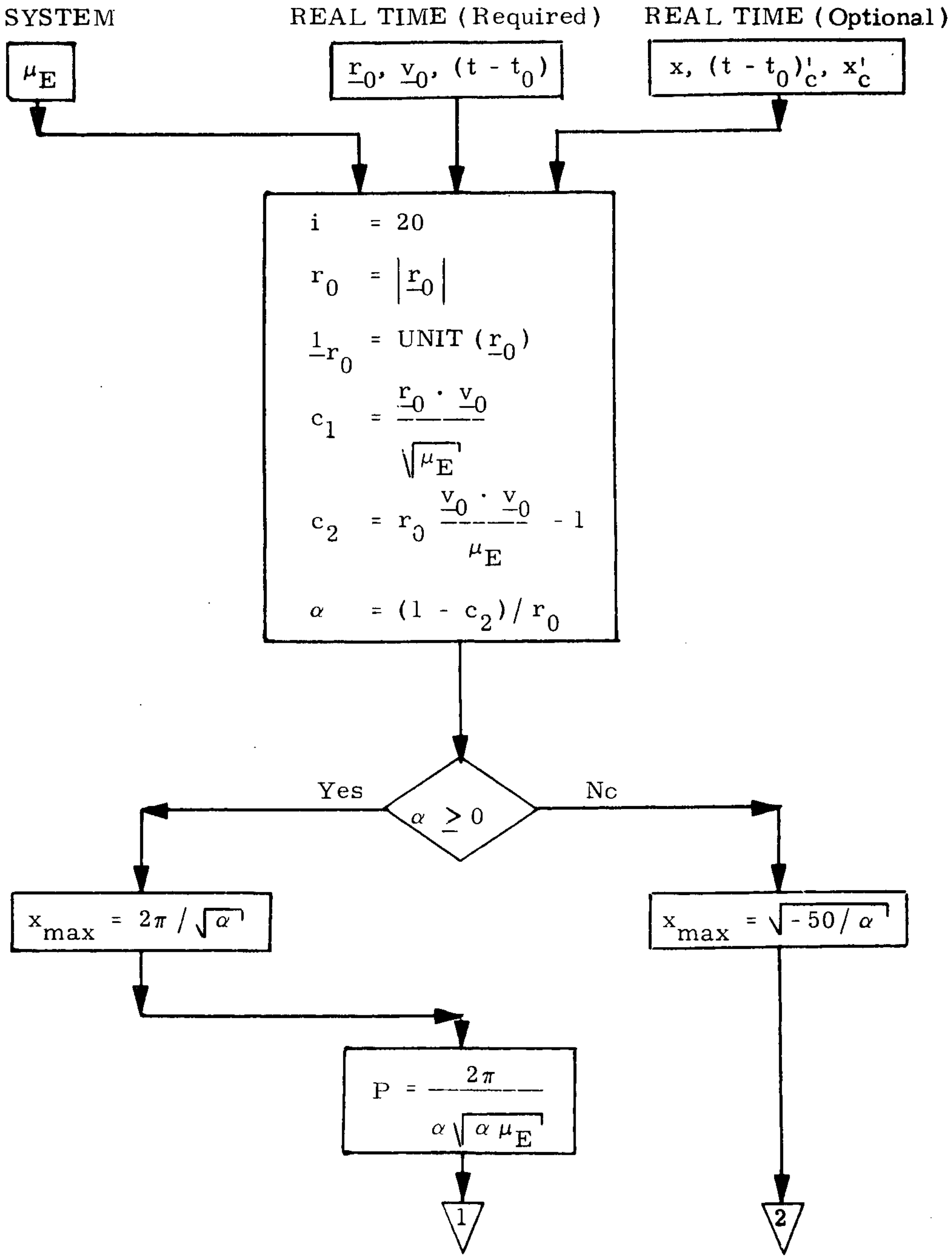
$$x/\sqrt{r_0} = \begin{cases} \Sigma & (W_1 > 0) \\ 2\pi/\sqrt{\alpha_N} - \Sigma & (W_1 < 0) \end{cases}$$

The above equations have been specifically formulated to avoid certain numerical difficulties.

9.6.1.1 Conic State Extrapolation (continued)

5. DETAILED FLOW DIAGRAMS

5.1 Conic State Extrapolation As A Function of Time (Kepler Routine)



SYSTEM
μ_E
REAL TIME (Required)
$\underline{r}_0$, $\underline{v}_0$, $(t - t_0)$
REAL TIME (Optional)
x, $(t - t_0)'_c$, x'_c
$i = 20$
$r_0 = |\underline{r}_0|$
$\underline{1}_{r_0} = \text{UNIT}(\underline{r}_0)$
$c_1 = \frac{\underline{r}_0 \cdot \underline{v}_0}{\sqrt{\mu_E}}$
$c_2 = r_0 \frac{\underline{v}_0 \cdot \underline{v}_0}{\mu_E} - 1$
$\alpha = (1 - c_2)/r_0$
$\alpha \geq 0$
Yes
Nc
$x_{max} = 2\pi / \sqrt{\alpha}$
$x_{max} = \sqrt{-50/\alpha}$
$P = \frac{2\pi}{\alpha\sqrt{\alpha\,\mu_E}}$
1
2

Figure 3a KEPLER ROUTINE DETAILED FLOW DIAGRAM

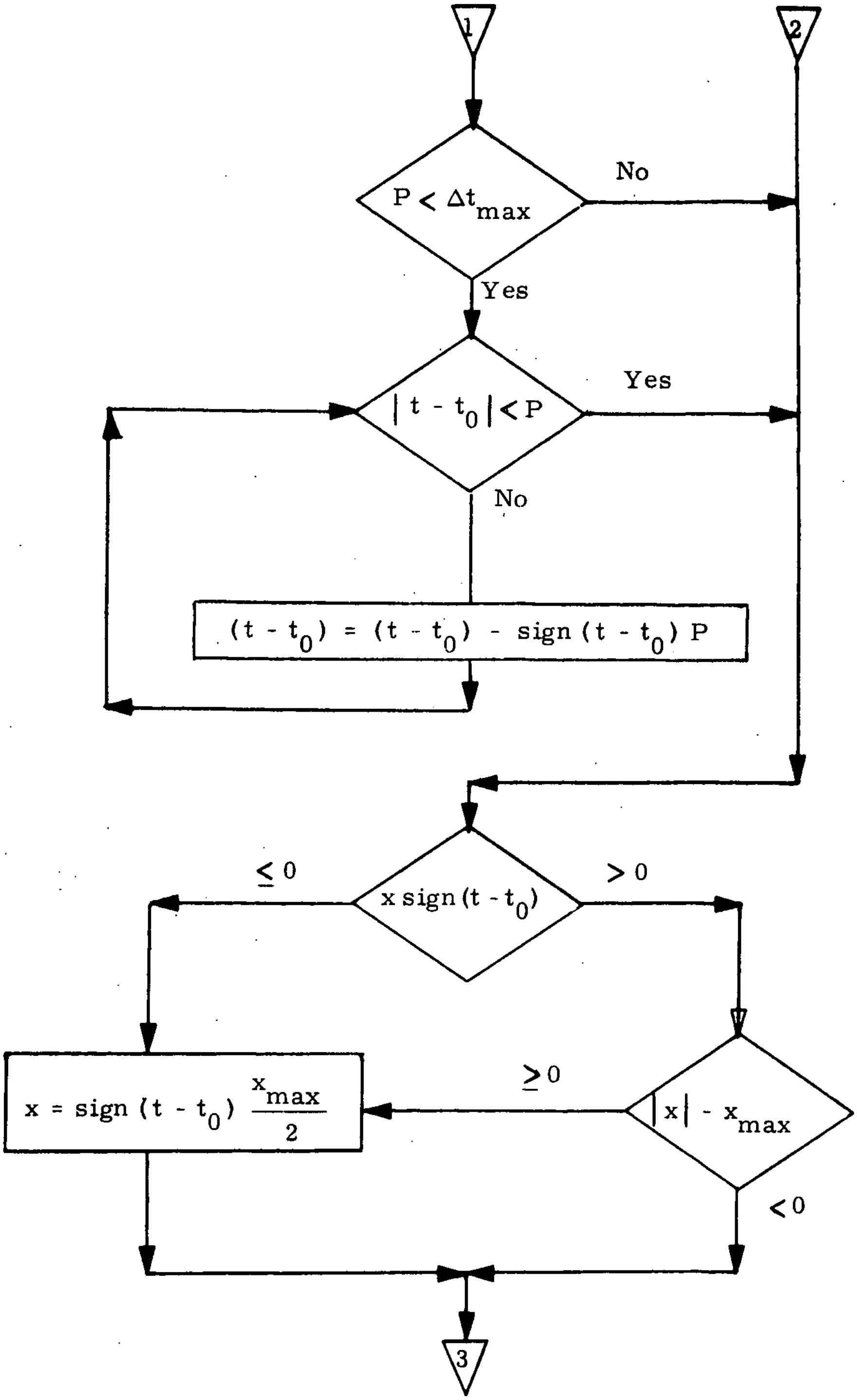

1
2
$P < \Delta t_{max}$
No
Yes
$|t - t_0| < P$
Yes
No
$(t - t_0) = (t - t_0) - \text{sign}(t - t_0)\,P$
$x\,\text{sign}(t - t_0)$
≤ 0
> 0
$x = \text{sign}(t - t_0)\,\frac{x_{max}}{2}$
≥ 0
$|x| - x_{max}$
< 0
3

Figure 3b KEPLER ROUTINE DETAILED FLOW DIAGRAM

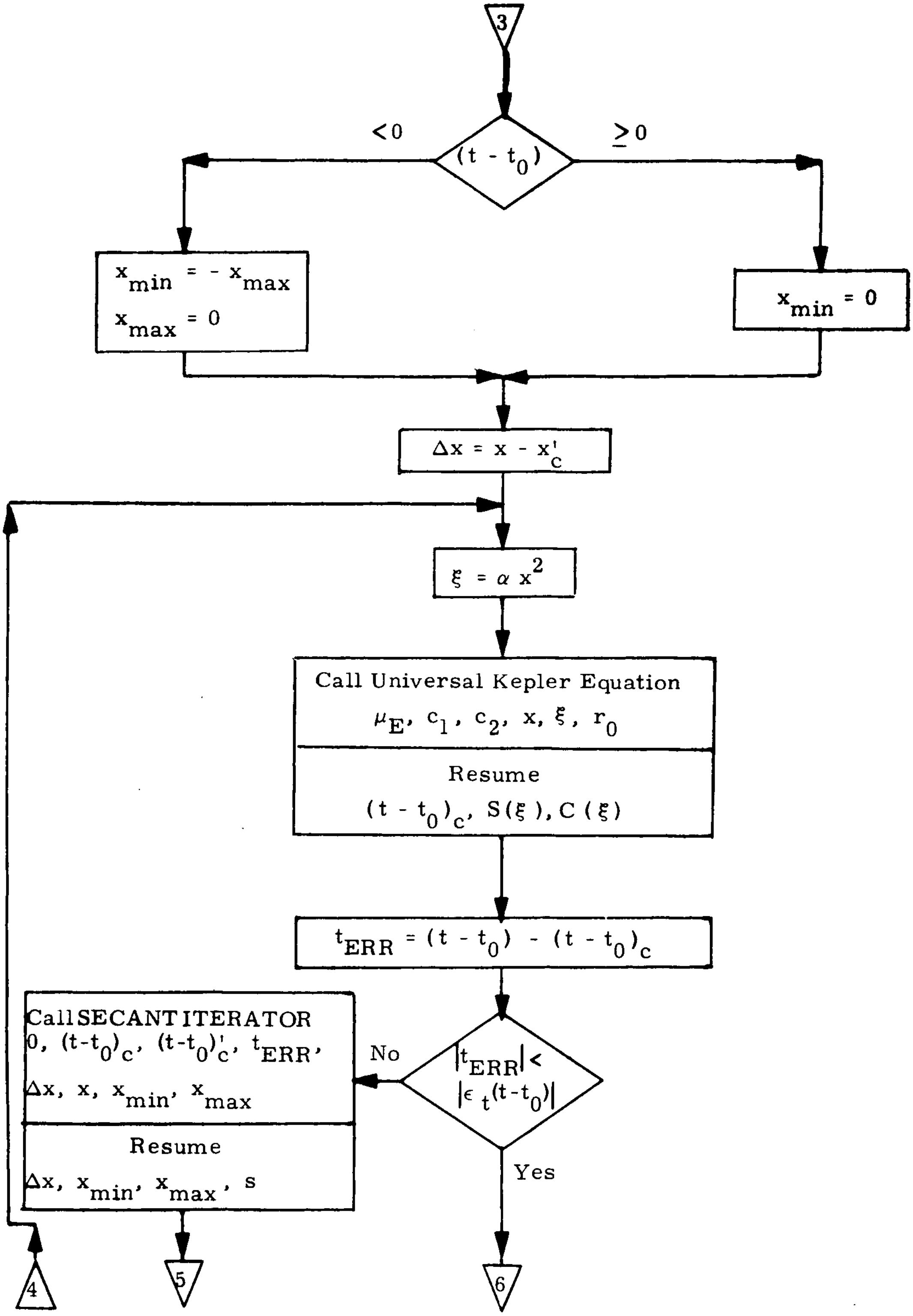

3
$(t - t_0)$
<0
≥0
$x_{min} = - x_{max}$
$x_{max} = 0$
$x_{min} = 0$
$\Delta x = x - x'_c$
$\xi = \alpha x^2$
Call Universal Kepler Equation
$\mu_E, c_1, c_2, x, \xi, r_0$
Resume
$(t - t_0)_c, S(\xi), C(\xi)$
$t_{ERR} = (t - t_0) - (t - t_0)_c$
$|t_{ERR}| <$
$|\epsilon_t (t-t_0)|$
No
Yes
Call SECANT ITERATOR
$0, (t-t_0)_c, (t-t_0)'_c, t_{ERR},$
$\Delta x, x, x_{min}, x_{max}$
Resume
$\Delta x, x_{min}, x_{max}, s$
4
5
6

Figure 3c KEPLER ROUTINE DETAILED FLOW DIAGRAM

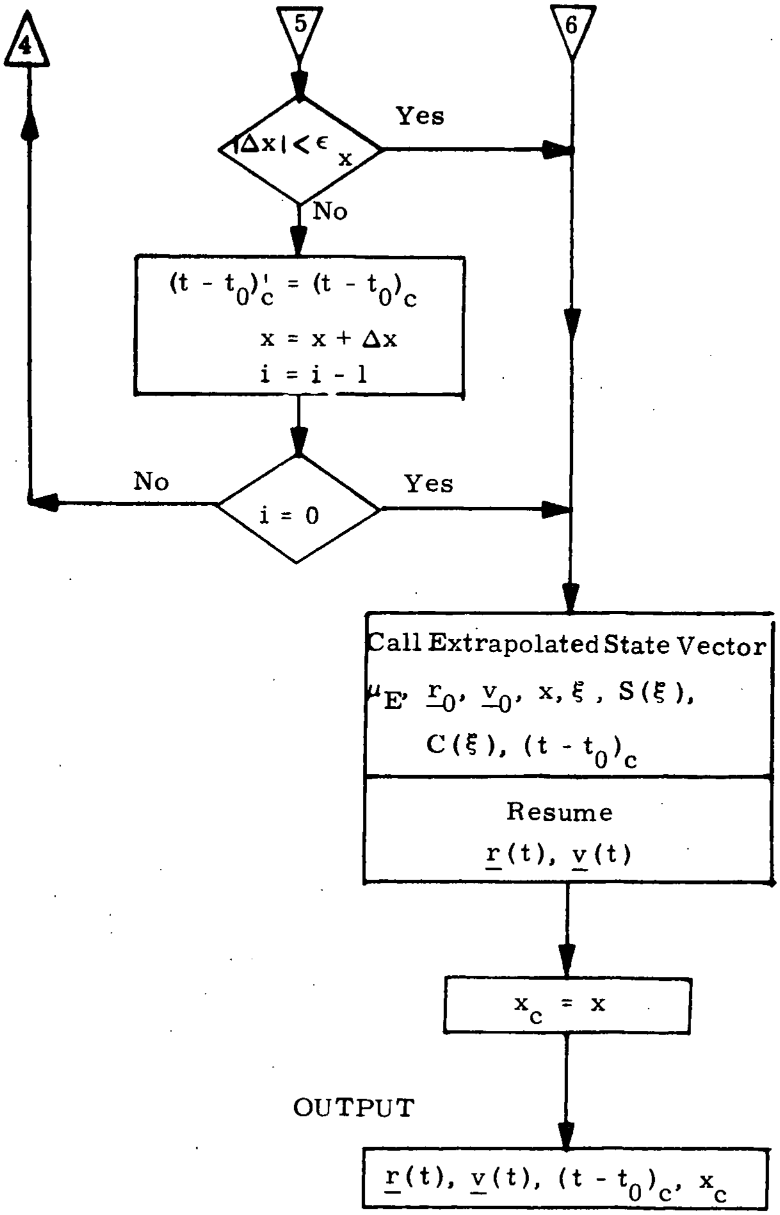

4
5
6
$|\Delta x| < \epsilon_x$
Yes
No
$(t - t_0)'_c = (t - t_0)_c$
$x = x + \Delta x$
$i = i - 1$
No
$i = 0$
Yes
Call Extrapolated State Vector
μ_E, $\underline{r}_0$, $\underline{v}_0$, x, ξ, S(ξ),
C(ξ), $(t - t_0)_c$
Resume
$\underline{r}(t)$, $\underline{v}(t)$
$x_c = x$
OUTPUT
$\underline{r}(t)$, $\underline{v}(t)$, $(t - t_0)_c$, x_c

Figure 3d KEPLER ROUTINE DETAILED FLOW DIAGRAM

5.2 Conic State Extrapolation As A Function of Transfer Angle

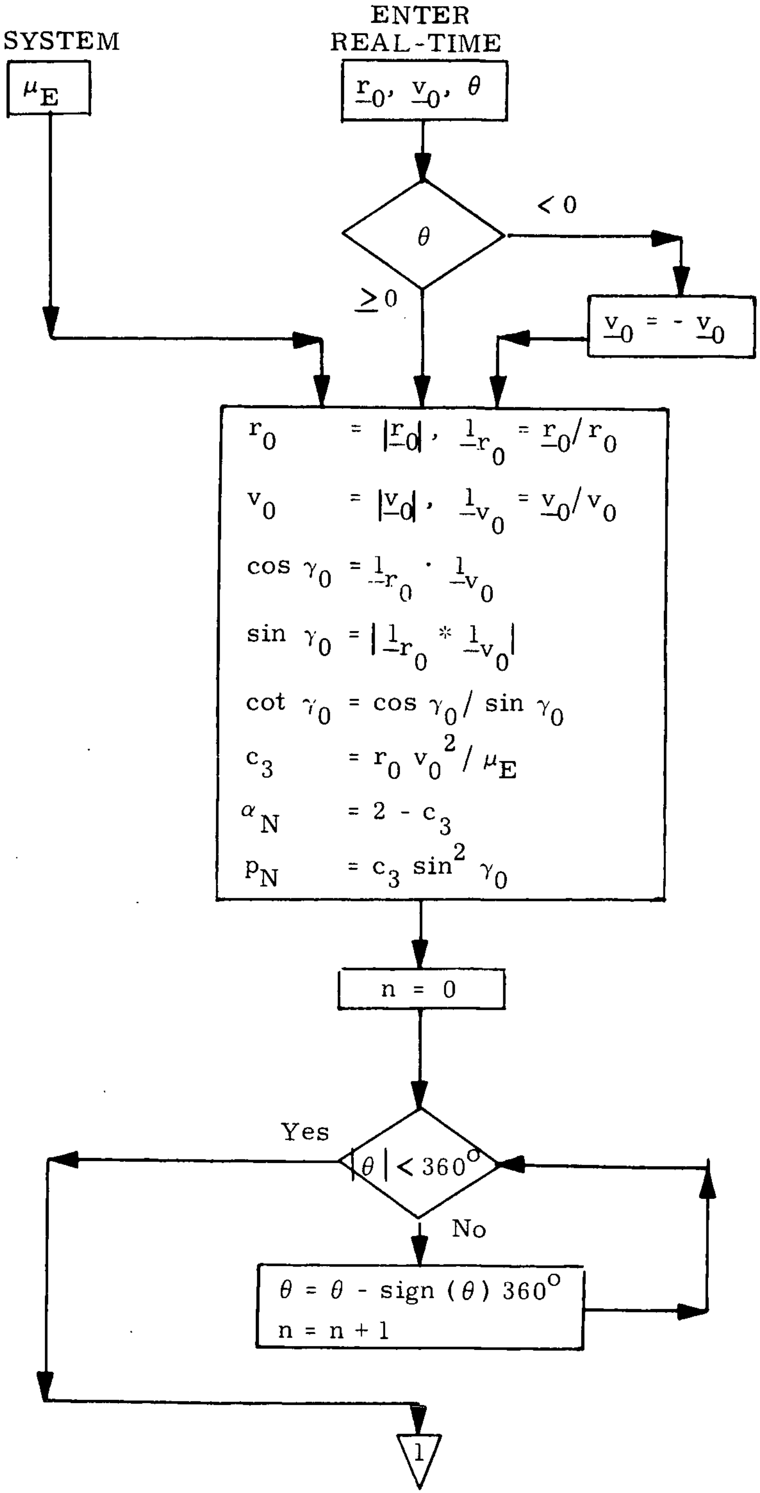

SYSTEM
μ_E
ENTER REAL-TIME
$\underline{r}_0, \underline{v}_0, \theta$
θ
< 0
≥ 0
$\underline{v}_0 = - \underline{v}_0$
$r_0 = |\underline{r}_0|, \underline{1}_{r_0} = \underline{r}_0 / r_0$
$v_0 = |\underline{v}_0|, \underline{1}_{v_0} = \underline{v}_0 / v_0$
$\cos \gamma_0 = \underline{1}_{r_0} \cdot \underline{1}_{v_0}$
$\sin \gamma_0 = | \underline{1}_{r_0} * \underline{1}_{v_0} |$
$\cot \gamma_0 = \cos \gamma_0 / \sin \gamma_0$
$c_3 = r_0 v_0^2 / \mu_E$
$\alpha_N = 2 - c_3$
$p_N = c_3 \sin^2 \gamma_0$
$n = 0$
Yes
$|\theta| < 360^\circ$
No
$\theta = \theta - \text{sign}(\theta)\, 360^\circ$
$n = n + 1$
1

Figure 4a THETA ROUTINE DETAILED FLOW DIAGRAM

9.6.1.1 Conic State Extrapolation (continued)

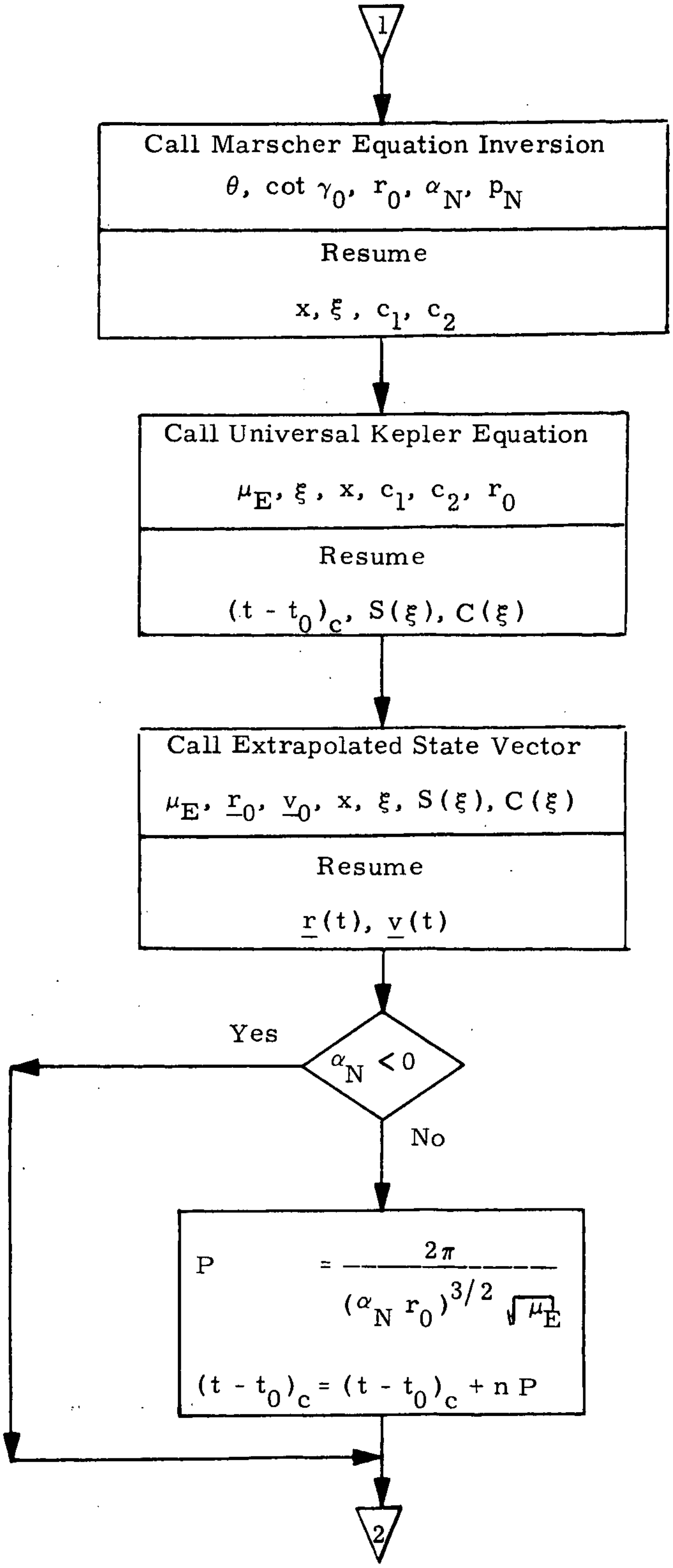

1
Call Marscher Equation Inversion
θ, cot γ_0, r_0, α_N, p_N
Resume
x, ξ , c_1, c_2
Call Universal Kepler Equation
μ_E, ξ , x, c_1, c_2, r_0
Resume
$(t - t_0)_c$, $S(\xi)$, $C(\xi)$
Call Extrapolated State Vector
μ_E, $\underline{r}_0$, $\underline{v}_0$, x, ξ, $S(\xi)$, $C(\xi)$
Resume
$\underline{r}(t)$, $\underline{v}(t)$
Yes
$\alpha_N < 0$
No
$P = \frac{2\pi}{(\alpha_N r_0)^{3/2} \sqrt{\mu_E}}$
$(t - t_0)_c = (t - t_0)_c + n P$
2

Figure 4b THETA ROUTINE DETAILED FLOW DIAGRAM

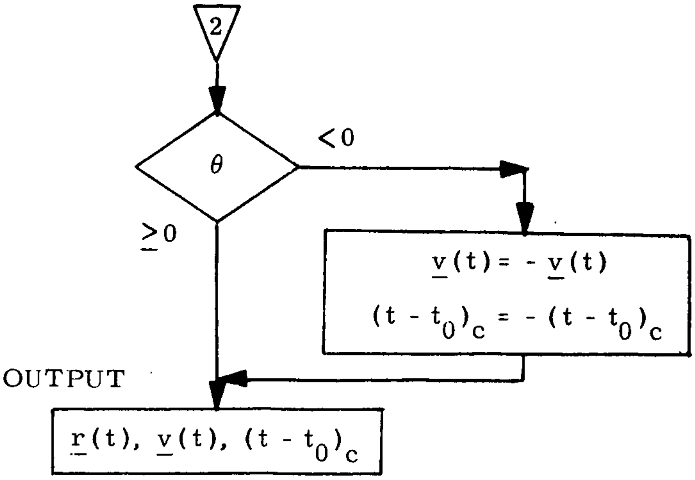

2
θ
<0
≥0
$\underline{v}(t) = -\underline{v}(t)$
$(t - t_0)_c = -(t - t_0)_c$
OUTPUT
$\underline{r}(t), \underline{v}(t), (t - t_0)_c$

Figure 4c THETA ROUTINE DETAILED FLOW DIAGRAM

5.3 Subroutines Used By The Transfer Time or Transfer Angle Conic Extrapolation Routines

5.3.1 Universal Kepler Equation

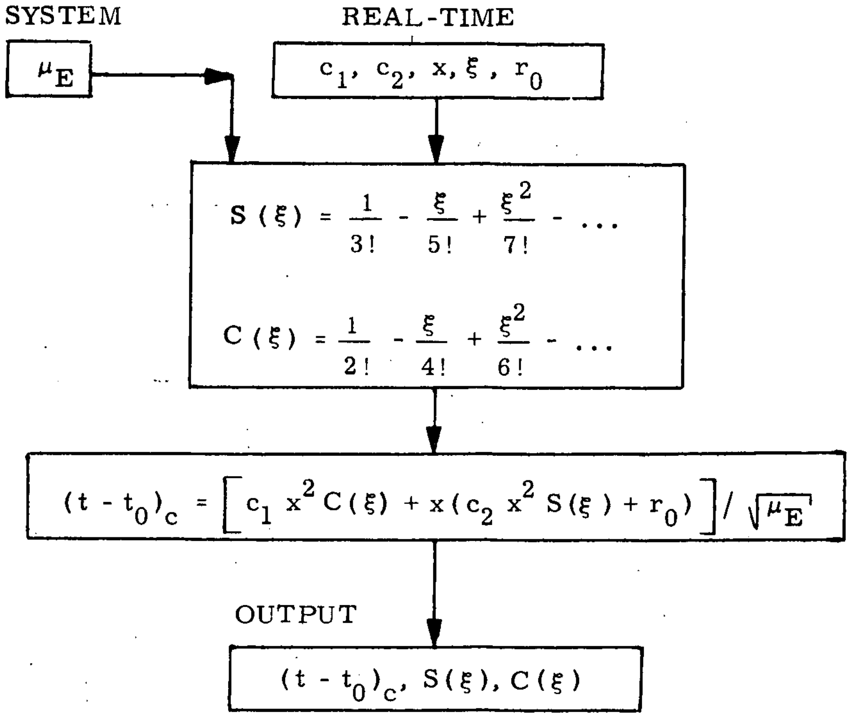

SYSTEM
μ_E
REAL-TIME
c_1, c_2, x, ξ, r_0
$S(\xi) = \frac{1}{3!} - \frac{\xi}{5!} + \frac{\xi^2}{7!} - \dots$
$C(\xi) = \frac{1}{2!} - \frac{\xi}{4!} + \frac{\xi^2}{6!} - \dots$
$(t - t_0)_c = \left[c_1 x^2 C(\xi) + x(c_2 x^2 S(\xi) + r_0)\right] / \sqrt{\mu_E}$
OUTPUT
$(t - t_0)_c, S(\xi), C(\xi)$

Figure 5 UNIVERSAL KEPLER EQUATION DETAILED FLOW DIAGRAM

5.3.2 Extrapolated State Vector

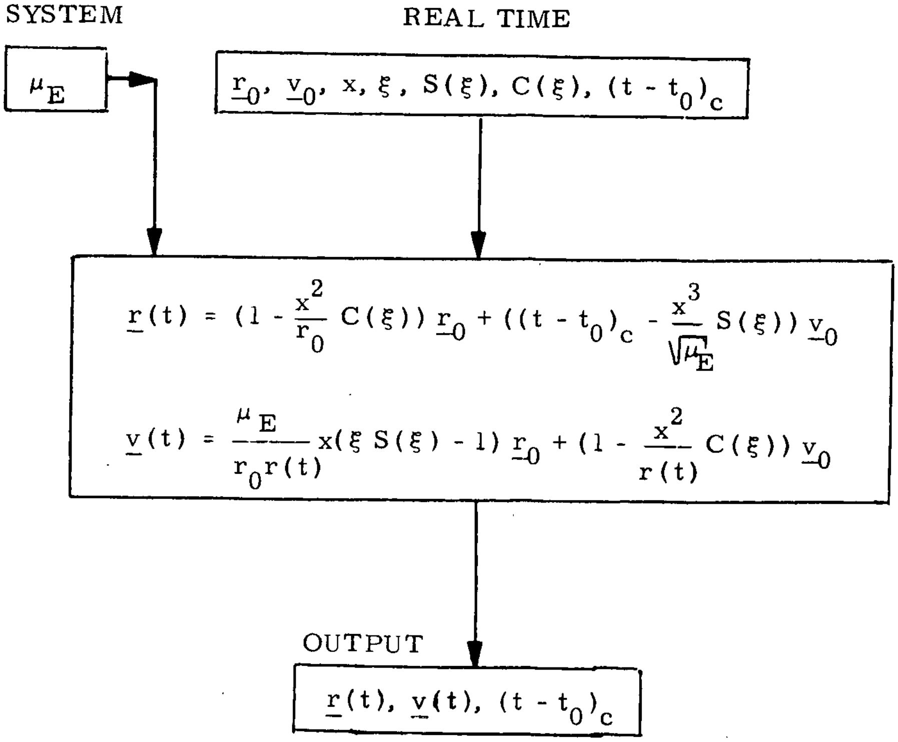

SYSTEM
REAL TIME
μ_E
$\underline{r}_0, \underline{v}_0, x, \xi, S(\xi), C(\xi), (t - t_0)_c$
$\underline{r}(t) = (1 - \frac{x^2}{r_0} C(\xi))\underline{r}_0 + ((t - t_0)_c - \frac{x^3}{\sqrt{\mu_E}} S(\xi))\underline{v}_0$
$\underline{v}(t) = \frac{\mu_E}{r_0 r(t)} x(\xi S(\xi) - 1)\underline{r}_0 + (1 - \frac{x^2}{r(t)} C(\xi))\underline{v}_0$
OUTPUT
$\underline{r}(t), \underline{v}(t), (t - t_0)_c$

Figure 6 EXTRAPOLATED STATE VECTOR EQUATION DETAILED FLOW DIAGRAM

5.3.3 Secant Iterator

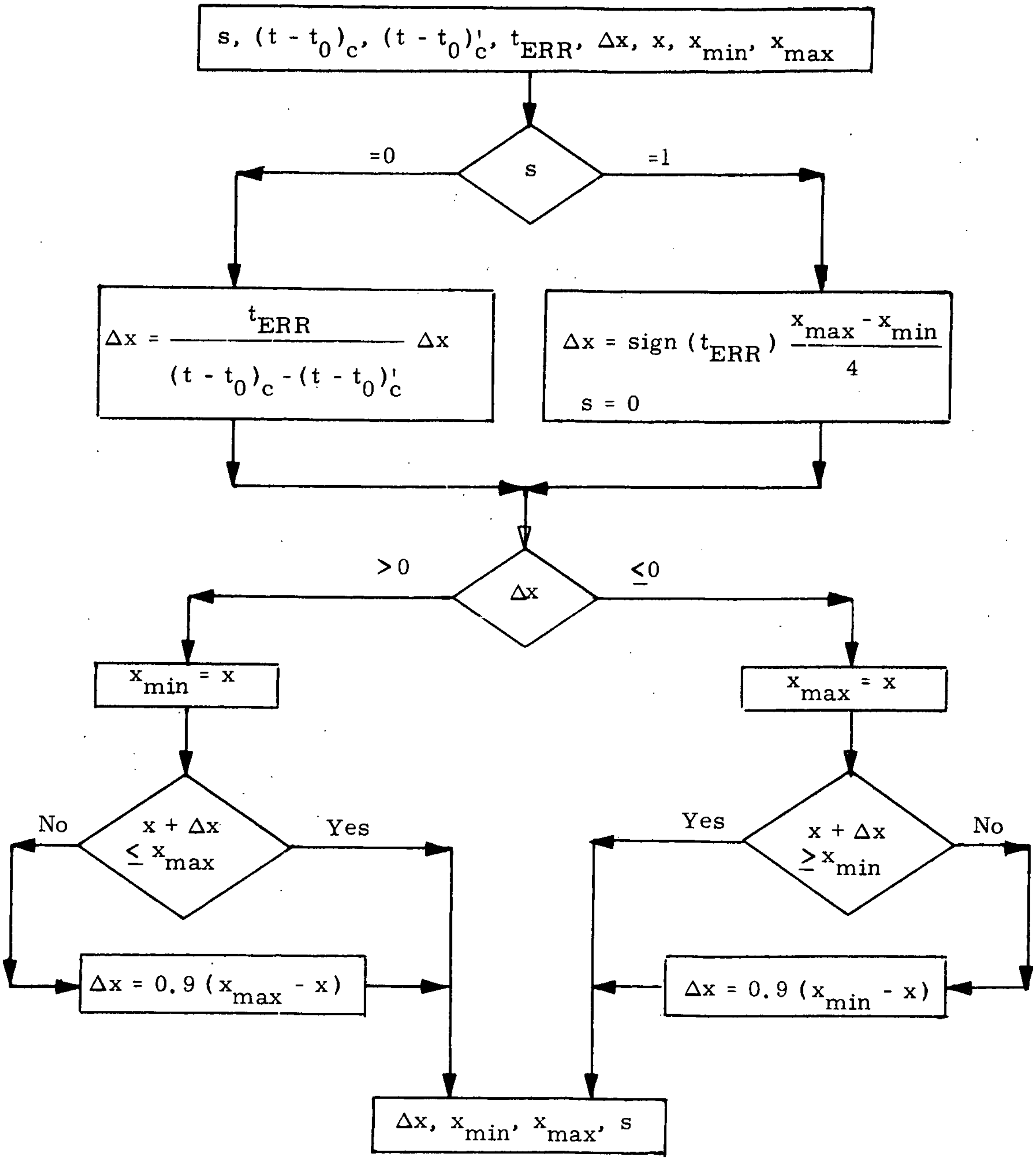
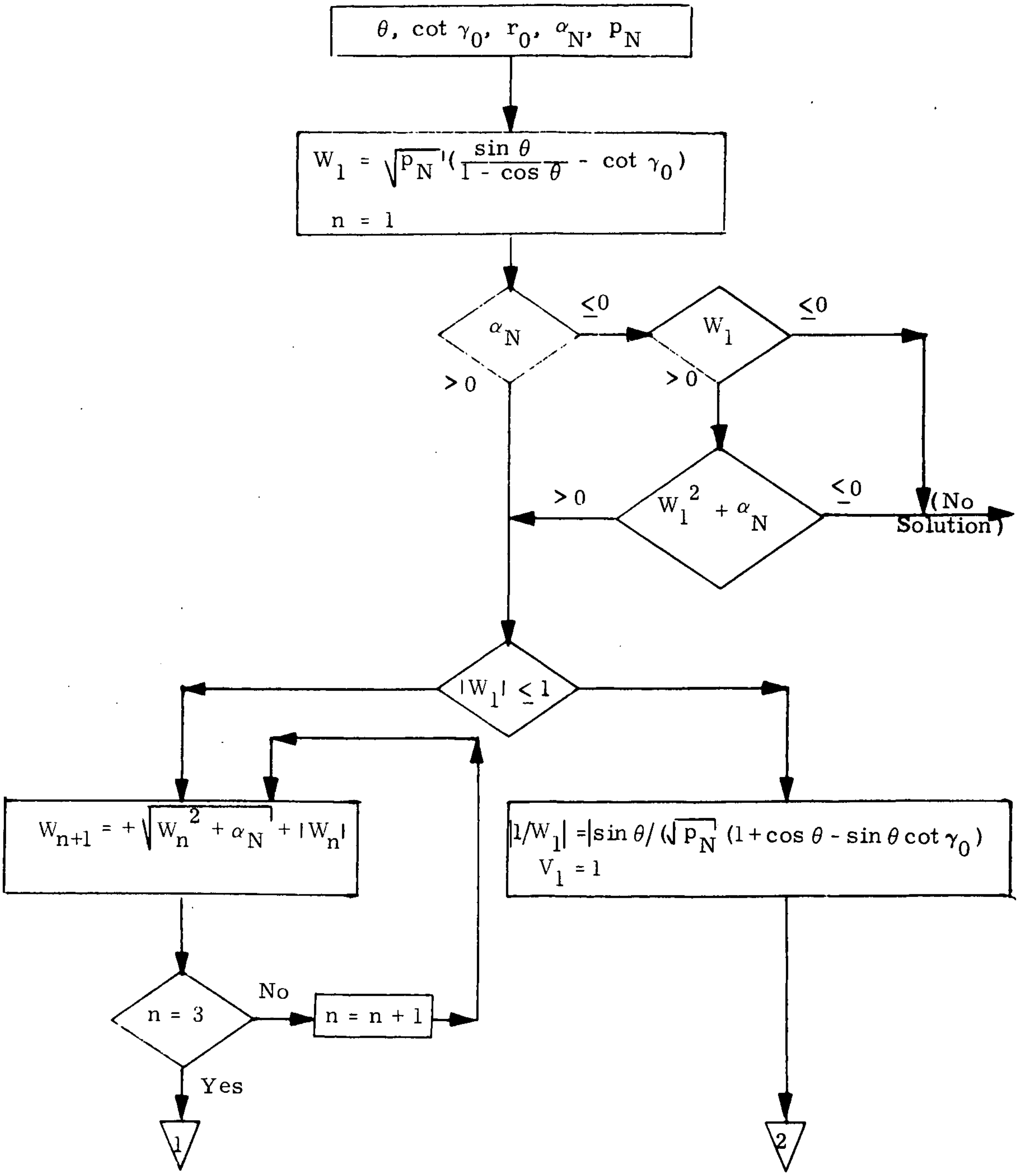

$s, (t - t_0)_c, (t - t_0)'_c, t_{ERR}, \Delta x, x, x_{min}, x_{max}$
s
=0
=1
$\Delta x = \frac{t_{ERR}}{(t - t_0)_c - (t - t_0)'_c} \Delta x$
$\Delta x = \text{sign}(t_{ERR}) \frac{x_{max} - x_{min}}{4}$
$s = 0$
Δx
> 0
≤ 0
$x_{min} = x$
$x_{max} = x$
No
$x + \Delta x \leq x_{max}$
Yes
Yes
$x + \Delta x \geq x_{min}$
No
$\Delta x = 0.9 (x_{max} - x)$
$\Delta x = 0.9 (x_{min} - x)$
$\Delta x, x_{min}, x_{max}, s$

Figure 7 SECANT ITERATOR DETAILED FLOW DIAGRAM

5.3.4 Marscher Equation Inversion



θ, cot γ_0, r_0, α_N, p_N
$W_1 = \sqrt{p_N}\left(\frac{\sin\theta}{1-\cos\theta} - \cot\gamma_0\right)$
n = 1
α_N
≤0
> 0
W_1
≤0
> 0
$W_1^2 + \alpha_N$
> 0
≤0
(No Solution)
$|W_1| \leq 1$
$W_{n+1} = + \sqrt{W_n^2 + \alpha_N} + |W_n|$
$|1/W_1| = |\sin\theta / (\sqrt{p_N}\,(1 + \cos\theta - \sin\theta\cot\gamma_0)$
$V_1 = 1$
n = 3
No
n = n + 1
Yes
1
2

Figure 8a MARSCHER EQUATION INVERSION DETAILED FLOW DIAGRAM

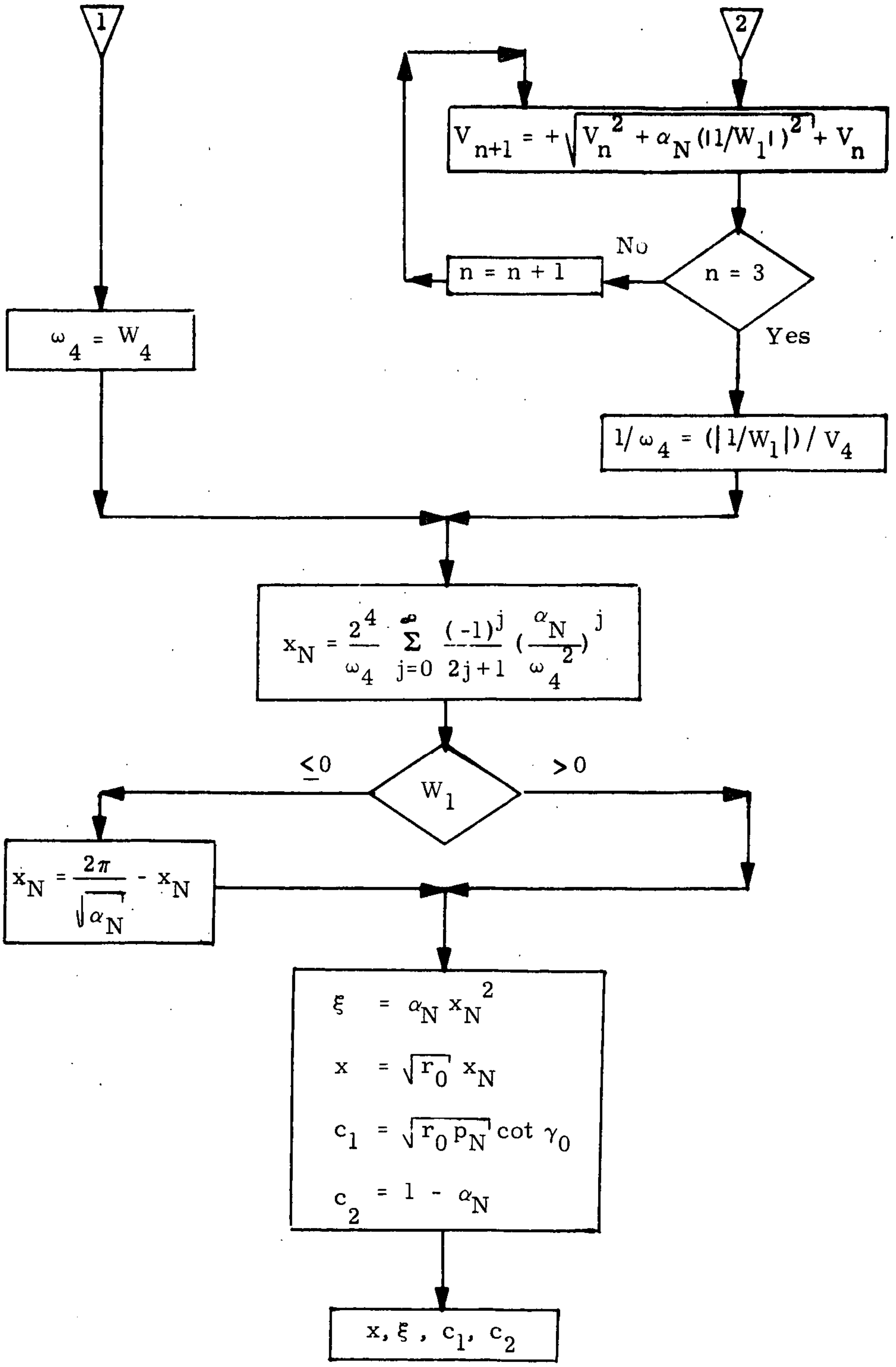

1
2
$V_{n+1} = +\sqrt{V_n^2 + \alpha_N (|1/W_1|)^2} + V_n$
No
$n = 3$
$n = n + 1$
Yes
$\omega_4 = W_4$
$1/\omega_4 = (|1/W_1|)/V_4$
$x_N = \frac{2^4}{\omega_4} \sum_{j=0}^{\infty} \frac{(-1)^j}{2j+1} (\frac{\alpha_N}{\omega_4^2})^j$
≤ 0
W_1
> 0
$x_N = \frac{2\pi}{\sqrt{\alpha_N}} - x_N$
$\xi = \alpha_N x_N^2$
$x = \sqrt{r_0}\, x_N$
$c_1 = \sqrt{r_0 p_N} \cot \gamma_0$
$c_2 = 1 - \alpha_N$
x, ξ, c_1, c_2

Figure 8b MARSCHER EQUATION INVERSION DETAILED FLOW DIAGRAM

6. SUPPLEMENTARY INFORMATION

The analytic expressions for the Universal Kepler Equation and the extrapolated position and velocity vectors are well known and are given by Battin (1964). Battin also outlines a Newton iteration technique for the solution of the Universal Kepler Equation; this technique converges somewhat faster than the secant technique but requires the evaluation of the derivative. It may be shown that if the derivative evaluation by itself takes more than 44% of the computation time used by the other calculations in one pass through the loop, then it is more efficient timewise to use the secant method.

Marscher's universal equation for cot $\theta/2$ was derived by him in his report (Marscher, 1965), and is the generalization of his "Three-Cotangent" equation:

$$\cot \frac{\theta}{2} = \frac{r_0}{\sqrt{p\,a}} \cot \frac{(E - E_0)}{2} + \cot \gamma_0$$

Marscher has also outlined in the report an iterative method of extrapolating the state based on his universal equation. The inversion of Marscher's universal equation was derived by Robertson (1967a).

Krause organized the details of the computation in both routines.

A derivation of the coefficients in the inversion of the Universal Kepler Equation is given in Robertson (1967 b) and Newman (1967).

References

1. Battin, R.H., 1964, Astronautical Guidance, McGraw-Hill.

2. Krause, K.W., 1968, A Unified Method of Solving Initial Value and Boundary Value Conic Trajectory Problems, TRW Interoffice Correspondence #3424.9-15 (January 1968).

3. Marscher, W., 1965, A Unified Method of Generating Conic Sections, MIT/IL Report R-479, February 1965.

4. Newman, C.M., 1967, The Inversion of Kepler's Equation, MIT/IL Space Guidance Analysis Memo #14-67.

5. Robertson, W.M., 1967a, Explicit Universal Series Solutions for the Universal Variable x, MIT/IL Space Guidance Analysis Memo #8-67.

6. Robertson, W.M., 1967b, Time-Series Expansions of the Universal Variable x, MIT/IL Space Guidance Analysis Memo #13-67.

SPACE SHUTTLE

GN&C SOFTWARE EQUATION SUBMITTAL

Software Equation Section: Precision State and Filter Weighting Matrix Extrapolation

Submittal No. 7 B

Function: Provide precision advancement of the Inertial State and Weighting Matrix

Module No. ON-2 Function No. 1 (MSC 03690 Rev. A)

Submitted by: W. M. Robertson Co. MIT No. 4 (Rev. 1)

Date: 21 Oct. 1971 (B Rev.)

NASA Contract: J. Suddath Organization: EG2

Approved by Panel III: K.J. Cox Date: 10/21/71

Summary Description: Provides the capability to extrapolate any spacecraft geocentric state vector either backwards or forwards in time through a force field in which all significant perturbation effects have been included. Also provided is the capability of extrapolating the filter-weighting matrix along the precision trajectory. The filter-weighting matrix extrapolation equations have been modified and expanded to account for the cross-correlation terms when a weighting matrix involving two state vectors is being extrapolated. The concept of a (6 x d) filter-weighting sub-matrix corresponding to a single state vector has been introduced. The earth's J_2 gravity field has been elevated to tha status of the standard option for the perturbing acceleration.

Shuttle Configuration: These equations are independent of the Shuttle Configuration.

Comments:

(Design Status) ________

(Verification Status) ________

Panel Comments:

Revision: B. Prior submittals Feb. and Aug. 1971.

9.6.1.2 Precision State Extrapolation

1. INTRODUCTION

The Precision State and Filter Weighting Matrix Extrapolation Routine provides the capability to extrapolate any spacecraft geocentric state vector either backwards or forwards in time through a force field consisting of the earth's primary central-force gravitational attraction and a superimposed perturbing acceleration. The perturbing acceleration may be either the single dominant term (J_2) of the earth's oblateness or a more complete expression involving all significant perturbation effects. The Routine also provides the capability of extrapolating the filter-weighting matrix along the precision trajectory. This matrix, also known as the "W-matrix", is a square root form of the error covariance matrix and contains statistical information relative to the accuracies of the state vectors and certain other optionally estimated quantities.

On any one call, the routine extrapolates only one state vector and only those six rows of the filter-weighting matrix relating to this state vector. Two calls are required to extrapolate two separate state vectors and a complete filter-weighting matrix pertaining to two state vectors. The complete extrapolated filter-weighting matrix is obtained by properly adjoining the two separately extrapolated submatrices of six rows each.

The routine is merely a coded algorithm for the numerical solution of modified forms of the basic differential equations which are satisfied by the geocentric state vector of the spacecraft's center of mass and by the filter-weighting matrix, namely:

$$\frac{d^2}{dt^2}\underline{r}(t) + \frac{\mu}{r^3(t)}\underline{r}(t) = \underline{a}_d(t)$$

and

$$\frac{d}{dt}W(t) = F(t)\,W(t),$$

where $\underline{a}_d(t)$ is the vector sum of all the desired perturbing accelerations, and $F(t)$ is a matrix containing the gravity gradient matrix and the identity matrix in its off-diagonal sub-blocks.

Because of its high accuracy and its capability of extrapolaing the filter-weighting matrix, this routine serves as the computational foundation for precise space navigation. It suffers from a relatively slow computation speed in comparison with the Conic State Extrapolation Routine.

NOMENCLATURE

$\underline{a}_d(t)$	Perturbing acceleration at time t
c_{nom}	Constant for adjustment of nominal step-size
d	Number of columns in the filter weighting submatrix W
$E_d(t)$	Covariance matrix of dimension d
$f(q)$	Special function of q defined in text
$G(t)$	Gravity gradient matrix
$\underline{i}_{pole}$	Unit vector of earth's north polar axis expressed in reference coordinates
$\underline{i}_r$	Unit vector in the direction of the position vector $\underline{r}$
I_3	Three-dimensional identity matrix
j	Number of additional quantities, such as landmark locations or instrument biases, being estimated.
J_2	Constant describing dominant term of earth's oblateness
q	Special function of $\underline{r}$ and $\underline{\delta}$ defined in text

9.6.1.2 Precision State Extrapolation (continued)

$\underline{r}_0$	Geocentric position vector at time t_0
$\underline{r}(t)$	Geocentric position vector at time t
$r(t)$	Magnitude of geocentric position vector
$\underline{r}_{con}(t)$	Reference conic position vector at time t
$r_{con}(t)$	Magnitude of reference conic position vector at time t
r_E	Mean equatorial radius of the earth
$\underline{r}_F$	Geocentric position vector at time t_F
$\underline{r}_j$	Intermediate values of $\underline{r}$
s_{pert}	Switch indicating the perturbing accelerations to be included
s_W	Switch controlling whether state or filter-weighting matrix integration is being performed (used only internally in routine)
t_0	Initial time point
t_F	Time to which it is desired to extrapolate $(\underline{r}_0, \underline{v}_0)$ and optionally W_0
$\underline{v}_0$	Geocentric velocity vector at time t_0
$\underline{v}_F$	Geocentric velocity vector at time t_F
$\underline{v}_{con}(t)$	Reference conic velocity vector at time t
W_0	Filter-weighting matrix at time t_0
W_F	Filter-weighting matrix at time t_F
$\underline{w}_{k,i}$	Three-dimensional column vectors into which the filter-weighting matrix is partitioned
x	Independent variable in Kepler routine

9.6.1.2 Precision State Extrapolation (continued)

x' Previous value of x

$\underline{\gamma}(t)$ Vector random variable of dimension j representing errors in the additionally estimated quantities such as landmark locations or instrument biases

$\underline{\delta}(t)$ Position deviation vector of true position from reference conic position at time t

δ_{max} Maximum value of $\left|\underline{\delta}\right|$ permitted (used as rectification criterion)

Δt Time-step size in numerical integration of differential equation

Δt_{max} Maximum permissible time-step size

Δt_{nom} Nominal integration time-step size

ϵ_t Time convergence tolerance criterion

$\underline{\epsilon}(t)$ Random variable representing error in estimate of position vector at time t

$\underline{\eta}(t)$ Random variable representing error in estimate of velocity vector at time t

μ Earth's gravitational parameter

$\underline{\nu}(t)$ Velocity deviation vector of true velocity from reference conic velocity at time t

ν_{max} Maximum value of $\left|\underline{\nu}\right|$ permitted (used as rectification criterion)

τ Time interval since last rectification

τ' Previous value of τ

ϕ Geocentric latitude

2. FUNCTIONAL FLOW DIAGRAM

The Precision State and Filter Weighting Matrix Extrapolation Routine performs its functions by integrating modified forms of the basic differential equations at a sequence of points separated by intervals known as time-steps, which are not necessarily of the same size. The routine automatically determines the size to be taken at each step.

As shown in Fig. 1, the state vector and (optionally) the filter-weighting sub-matrix are updated one step at a time along the precision trajectory until the specified overall transfer time interval is exactly attained. (The size of the last time-step is adjusted as necessary to make this possible.)

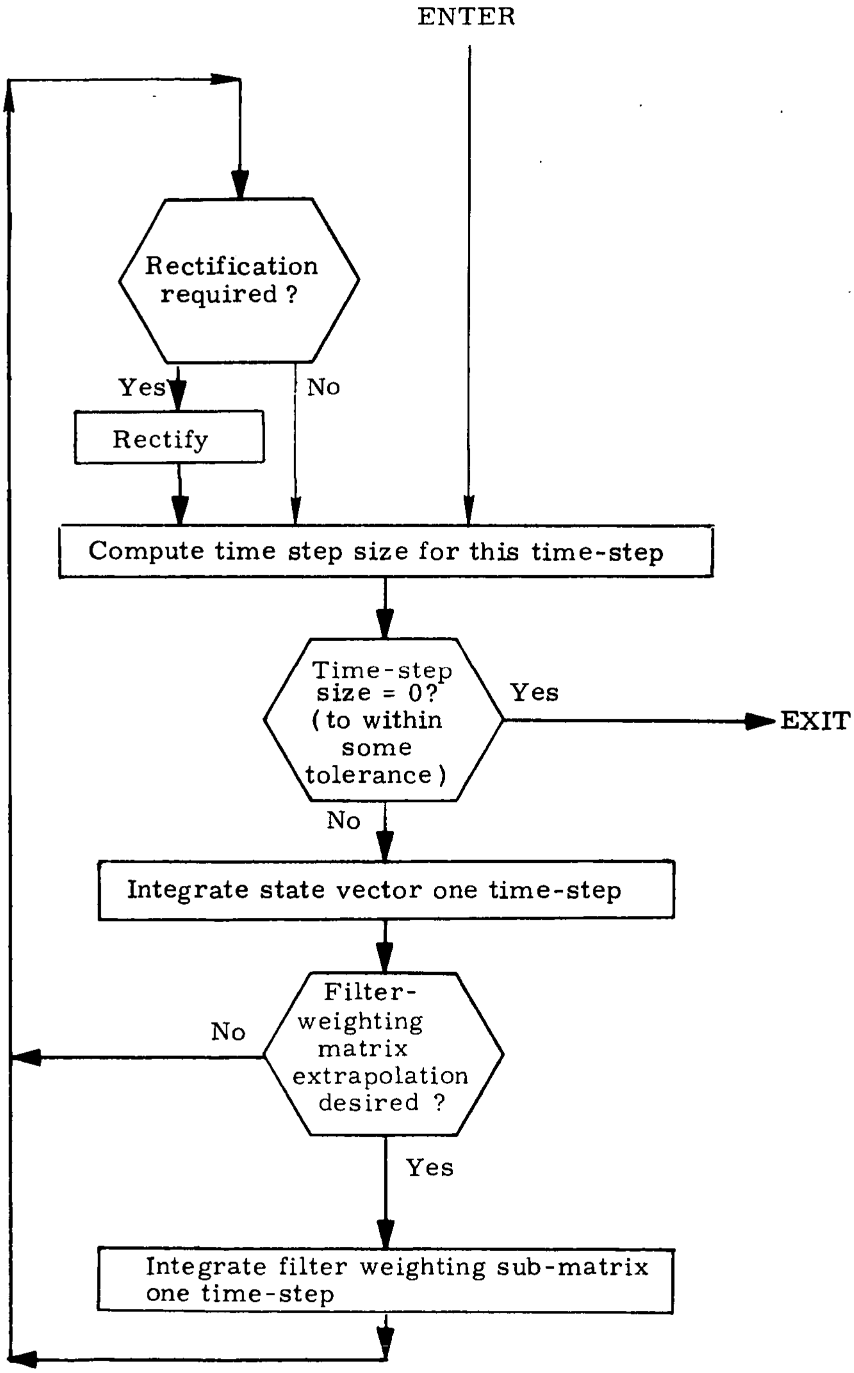

ENTER
Rectification required ?
Yes
No
Rectify
Compute time step size for this time-step
Time-step size = 0? (to within some tolerance)
Yes
EXIT
No
Integrate state vector one time-step
Filter-weighting matrix extrapolation desired ?
No
Yes
Integrate filter weighting sub-matrix one time-step

Figure 1. Functional Flow Diagram Precision State and Filter Weighting Matrix Extrapolation Routine

3. INPUT AND OUTPUT VARIABLES

The Precision State and Filter Weighting Matrix Extrapolation Routine has the following input and output variables;

Input Variables

$(\underline{r}_0, \underline{v}_0)$ Geocentric state vector to be extrapolated

t_0 Time associated with $(\underline{r}_0, \underline{v}_0)$ and W_0

t_F Time to which it is desired to extrapolate $(\underline{r}_0, \underline{v}_0)$ and optionally W_0

W_0 Filter-weighting sub-matrix to be extrapolated (optional) (W_0 has dimension 6 x d)

d Number of columns in filter-weighting sub-matrix (d = 0, 6, 7, ..., where 0 indicates no W-matrix extrapolation)

s_{pert} Switch indicating the perturbing accelerations to be included. ($s_{pert} = 1$ implies J_2 oblateness term only; $s_{pert} > 1$ implies a more complete perturbing acceleration model (or models).)

Output Variables

$(\underline{r}_F, \underline{v}_F)$ Extrapolated geocentric state vector

W_F Extrapolated filter-weighting sub-matrix of dimension 6 x d

4. DESCRIPTION OF EQUATIONS

4.1 Precision State Extrapolation Equations

Since the perturbing acceleration is small compared with the central force field, direct numerical integration of the basic differential equations of motion of the spacecraft state vector is inefficient. Instead, a technique due to Encke is utilized in which only the deviations of the state from a reference conic orbit are numerically integrated. The positions and velocities along the reference conic are obtained from the Kepler routine.

At time t_0 the position and velocity vectors, $\underline{r}_0$ and $\underline{v}_0$, define an osculating conic orbit. Because of the perturbing accelerations, the true position and velocity vectors $\underline{r}(t)$ and $\underline{v}(t)$ will deviate as time progresses from the conic position and velocity vectors $\underline{r}_{con}(t)$ and $\underline{v}_{con}(t)$ which have been conically extrapolated from $\underline{r}_0$ and $\underline{v}_0$. Let

$$\underline{\delta}(t) = \underline{r}(t) - \underline{r}_{con}(t)$$

$$\underline{\nu}(t) = \underline{v}(t) - \underline{v}_{con}(t)$$

be the vector deviations. It can be shown that the position deviation $\underline{\delta}(t)$ satisfies the differential equation

$$\frac{d^2}{dt^2}\underline{\delta}(t) + \frac{\mu}{r_{con}^{3}(t)}\left[f(q)\,\underline{r}(t) + \underline{\delta}(t) \right] = \underline{a}_d(t)$$

with the initial conditions

$$\underline{\delta}(t_0) = \underline{0}, \quad \underline{\nu}(t_0) = \underline{0}$$

where

$$q = \frac{(\underline{\delta} - 2\underline{r}) \cdot \underline{\delta}}{r^2}, \quad f(q) = q \, \frac{3 + 3q + q^2}{1 + (1 + q)^{3/2}},$$

and $\underline{a}_d(t)$ is the total perturbing acceleration. The above second order differential equation in the deviation vector $\underline{\delta}(t)$ is numerically integrated by a method described in a later subsection.

The term

$$\frac{\mu}{r_{con}^{3}} \left[f(q) \, \underline{r}(t) + \underline{\delta}(t) \right]$$

must remain small, i.e. of the same order as $\underline{a}_d(t)$, if the method is to be efficient. As the deviation vector $\underline{\delta}(t)$ grows in magnitude, this term will eventually increase in size. When

$$|\underline{\delta}(t)| > 0.01 \, |\underline{r}_{con}(t)| \text{ or } |\underline{\nu}(t)| > 0.01 \, |\underline{v}_{con}(t)|$$

or when

$$|\underline{\delta}(t)| > \delta_{max} \text{ or } |\underline{\nu}(t)| > \nu_{max},$$

a new osculating conic orbit is established based on the latest precision position and velocity vectors $\underline{r}(t)$ and $\underline{v}(t)$, the deviations $\underline{\delta}(t)$ and $\underline{\nu}(t)$ are zeroed, and the numerical integration of $\underline{\delta}(t)$ and $\underline{\nu}(t)$ continues. The process of establishing a new conic orbit is called rectification.

The total perturbing acceleration $a_d(t)$ is in general the vector sum of all the desired individual perturbing accelerations comprising the total force field, such as those due to the earth's oblateness, the gravitational attractions of the sun and moon, and the earth's atmospheric drag. Since many Shuttle applications will require only the perturbing effect of the dominant term J_2 of the earth's oblateness, the use of only this term has been made a standard option in the routine diagrammed in Section 5. However, provision has been made for handling a completely general perturbing acceleration. The form of this perturbing acceleration will depend primarily upon the requirements of the Orbit Navigation function.

The explicit expression for the earth's J_2 oblateness acceleration alone is:

$$\underline{a}_d = - \frac{\mu}{r^2} \left\{ \frac{3}{2} J_2 \left[\frac{r_E}{r} \right]^2 \left[(1 - 5 \sin^2 \phi)\, \underline{i}_r + 2 \sin\phi\, \underline{i}_{pole} \right] \right\}$$

where

$\underline{i}_r$ is the unit position vector in reference coordinates,

$\underline{i}_{pole}$ is the unit vector of the earth's north polar axis expressed in reference coordinates,

$\sin\phi = \underline{i}_r \cdot \underline{i}_{pole}$,

and

r_E is the mean equatorial radius of the earth.

4.2 Filter-Weighting (W) Matrix Extrapolation Equations

The position and velocity vectors which are maintained by the spacecraft's computer are only estimates of the actual values of these vectors. As part of the navigation technique it is also necessary for the computer to maintain statistical information about the position and velocity vectors. Furthermore, in particular applications it is necessary to include statistical data on various other quantities, such as landmark locations during Orbit Navigation and certain instrument biases during Co-orbiting Vehicle Navigation. The filter-weighting W-matrix is used for all these purposes.

If $\underline{\epsilon}(t)$ and $\underline{\eta}(t)$ are three dimensional vector random variables with zero mean which represent the errors in the estimates of a spacecraft's position and velocity at time t, then the six-dimensional state error covariance matrix $E_6(t)$ at time t is defined by:

$$E_6(t) = \begin{bmatrix} \overline{\underline{\epsilon}(t)\,\underline{\epsilon}(t)^T} & \overline{\underline{\epsilon}(t)\,\underline{\eta}(t)^T} \\ \overline{\underline{\eta}(t)\,\underline{\epsilon}(t)^T} & \overline{\underline{\eta}(t)\,\underline{\eta}(t)^T} \end{bmatrix},$$

where the bar represents the expected value or ensemble average at the fixed time t of each element of the matrix over which it appears.

If $\underline{\gamma}(t)$ is a j-dimensional vector random variable with zero mean which represents the errors in the estimates of the j additionally estimated quantities such as landmark locations or instrument biases, then a $(6+j)$ - dimensional state and other parameter covariance matrix $E_{(6+j)}(t)$ is defined by:

$$E_{(6+j)}(t) = \left[\begin{array}{ccc} \multicolumn{2}{c}{} & \overline{\underline{\epsilon}(t)\,\underline{\gamma}(t)^T} \\ \multicolumn{2}{c}{E_6(t)} & \overline{\underline{\eta}(t)\,\underline{\gamma}(t)^T} \\ \overline{\underline{\gamma}(t)\,\underline{\epsilon}(t)^T} & \overline{\underline{\gamma}(t)\,\underline{\eta}(t)^T} & \overline{\underline{\gamma}(t)\,\underline{\gamma}(t)^T} \end{array} \right]$$

Further, if the statistical properties of the positions and velocities of two separate spacecraft are to be maintained, a twelve-dimensional state covariance matrix is defined by:

$$E_{12}(t) = \left[\begin{array}{cccc} \overline{\underline{\epsilon}_P\,\underline{\epsilon}_P^T} & \overline{\underline{\epsilon}_P\,\underline{\eta}_P^T} & \overline{\underline{\epsilon}_P\,\underline{\epsilon}_T^T} & \overline{\underline{\epsilon}_P\,\underline{\eta}_T^T} \\ \overline{\underline{\eta}_P\,\underline{\epsilon}_P^T} & \overline{\underline{\eta}_P\,\underline{\eta}_P^T} & \overline{\underline{\eta}_P\,\underline{\epsilon}_T^T} & \overline{\underline{\eta}_P\,\underline{\eta}_T^T} \\ \overline{\underline{\epsilon}_T\,\underline{\epsilon}_P^T} & \overline{\underline{\epsilon}_T\,\underline{\eta}_P^T} & \overline{\underline{\epsilon}_T\,\underline{\epsilon}_T^T} & \overline{\underline{\epsilon}_T\,\underline{\eta}_T^T} \\ \overline{\underline{\eta}_T\,\underline{\epsilon}_P^T} & \overline{\underline{\eta}_T\,\underline{\eta}_P^T} & \overline{\underline{\eta}_T\,\underline{\epsilon}_T^T} & \overline{\underline{\eta}_T\,\underline{\eta}_T^T} \end{array} \right]$$

where the subscripts P and T refer to the primary and target vehicles, respectively.

And finally, if the statistical properties of the j additionally estimated quantities are also to be maintained along with the two state vectors, a $(12+j)$ state and other parameter covariance matrix $E_{(12+j)}(t)$ is defined by:

9.6.1.2 Precision State Extrapolation (continued)

$$E_{(12+j)}(t) = \left[\begin{array}{cccc|c} \multicolumn{4}{c|}{} & \overline{\underline{\epsilon}_P \underline{\gamma}^T} \\ \multicolumn{4}{c|}{} & \overline{\underline{\eta}_P \underline{\gamma}^T} \\ \multicolumn{4}{c|}{E_{12}(t)} & \overline{\underline{\epsilon}_T \underline{\gamma}^T} \\ \multicolumn{4}{c|}{} & \overline{\underline{\eta}_T \underline{\gamma}^T} \\ \hline \overline{\underline{\gamma}\, \underline{\epsilon}_P^T} & \overline{\underline{\gamma}\, \underline{\eta}_P^T} & \overline{\underline{\gamma}\, \underline{\epsilon}_T^T} & \overline{\underline{\gamma}\, \underline{\eta}_T^T} & \overline{\underline{\gamma}\ \underline{\gamma}^T} \end{array}\right]$$

Rather than use one of the above covariance matrices in the navigation procedure, it is more convenient to use a matrix $W_d(t)$ having the same dimension d as the covariance matrix $E_d(t)$ and defined by:

$$E_d(t) = W_d(t)\, W_d(t)^T$$

The matrix $W_d(t)$ is called the filter-weighting matrix, and is in a sense a square root of the covariance matrix.

Extrapolation of the $W_d(t)$ matrix in time may be made by direct numerical integration of the differential equation which it satisfies: (where $j = 0, 1, 2, \ldots$ is the number of additionally estimated quantities)

$$\frac{d}{dt} W_{(6+j)}(t) = \left[\begin{array}{cc|c} O & I_3 & \\ G(t) & O & O_{(6xj)} \\ \hline \multicolumn{2}{c|}{O_{(j\,x\,6)}} & O_{(j\,x\,j)} \end{array}\right] W_{(6+j)}(t)$$

$$\frac{d}{dt} W_{(12+j)}(t) = \left[\begin{array}{cc|cc|c} O & I_3 & O & O & O_{(6xj)} \\ G_P(t) & O & O & O & \\ \hline O & O & O & I_3 & O_{(6xj)} \\ O & O & G_T(t) & O & \\ \hline \multicolumn{2}{c|}{O_{(jx6)}} & \multicolumn{2}{c|}{O_{(j\,x6)}} & O_{(j\,x\,j)} \end{array}\right] W_{(12+j)}(t)$$

where I_3 is the 3 x 3 identity matrix, the O's are zero matrices of the required dimensions, and the $G(t)$ are the 3 x 3 conic gravity gradient matrices

$$G(t) = \frac{\mu}{r^5(t)} \left[3\,\underline{r}(t)\,\underline{r}(t)^T - r^2(t)\,I_3 \right]$$

associated with the vehicle under consideration or with the primary (P) or target (T) vehicle.

Extrapolation of the W_d matrix may also be made by the following technique, which is somewhat simpler to implement in an on-board computer since matrix manipulations are reduced to more tractable vector manipulations.

If the d x d filter-weighting matrix $W_d = \left[w_{k,i} \right]$ is partitioned into three-dimensional column vectors $\underline{w}_{k,i}$ which bear the subscripts of their first component:

$$W_d = \begin{bmatrix} \underline{w}_{0,0} & \underline{w}_{0,1} & \cdots\cdots & \underline{w}_{0,(d-1)} \\ \underline{w}_{3,0} & \underline{w}_{3,1} & \cdots\cdots & \underline{w}_{3,(d-1)} \\ \ldots\ldots\ldots & \text{etc.} & \ldots\ldots\ldots & \ldots\ldots\ldots \end{bmatrix}$$

except for the last row where the $\underline{w}_{k,i}$ vectors may be one or two-dimensional if d is not divisible by three, then the previous first order differential equations are equivalent to:

$$\left.\begin{aligned} \frac{d^2}{dt^2}\,\underline{w}_{0,i} &= G(t)\,\underline{w}_{0,i} \\ &\text{with} \\ \underline{w}_{3,i} &= \frac{d}{dt}\,\underline{w}_{0,i} \\ \underline{w}_{k,i} &= \text{constant for } k \geq 6 \end{aligned}\right\} \quad i = 0, 1, \ldots (d-1)$$

and

$$\left.\begin{aligned} \frac{d^2}{dt^2}\,\underline{w}_{0,i} &= G_P(t)\,\underline{w}_{0,i} \\ \frac{d^2}{dt^2}\,\underline{w}_{6,i} &= G_T(t)\,\underline{w}_{6,i} \\ &\text{with} \\ \underline{w}_{3,i} &= \frac{d}{dt}\,\underline{w}_{0,i} \\ \underline{w}_{9,i} &= \frac{d}{dt}\,\underline{w}_{6,i} \\ \underline{w}_{k,i} &= \text{constant for } k \geq 12 \end{aligned}\right\} \quad i = 0, 1, \ldots, (d-1)$$

9.6.1.2 Precision State Extrapolation (continued)

When written out in full, the above equations are:

$$\left\{\begin{aligned}
&\frac{d^2}{d\,t^2}\,\underline{w}_{0,i} = \frac{\mu}{\underline{r}_P{}^5(t)}\left\{3\left[\underline{r}(t)\cdot\underline{w}_{0,i}(t)\right]\underline{r}(t) - r^2(t)\,\underline{w}_{0,i}(t)\right\}\\
&\text{with} \qquad\qquad\qquad\qquad\qquad\qquad i = 0,\ 1,\ \ldots,\ (d-1)\\
&\underline{w}_{3,i} = \frac{d}{d\,t}\,\underline{w}_{3,i}\\
&\underline{w}_{k,i} = \text{constant for } k \geq 6
\end{aligned}\right.$$

and

$$\left\{\begin{aligned}
&\frac{d^2}{d\,t^2}\,\underline{w}_{0,i} = \frac{\mu}{r_P{}^5(t)}\left\{3\left[\underline{r}_P(t)\cdot\underline{w}_{0,i}(t)\right]\underline{r}_P(t) - r_P{}^2(t)\,\underline{w}_{0,i}(t)\right\}\\
&\frac{d^2}{d\,t^2}\,\underline{w}_{6,i} = \frac{\mu}{r_T{}^5(t)}\left\{3\left[\underline{r}_T(t)\cdot\underline{w}_{6,i}(t)\right]\underline{r}_T(t) - r_T{}^2(t)\,\underline{w}_{6,i}(t)\right\}\\
&\text{with} \qquad\qquad\qquad\qquad\qquad\qquad i = 0,\ 1,\ \ldots,\ (d-1)\\
&\underline{w}_{3,i} = \frac{d}{d\,t}\,\underline{w}_{0,i}\\
&\underline{w}_{9,i} = \frac{d}{d\,t}\,\underline{w}_{6,i}\\
&\underline{w}_{k,i} = \text{constant for } k \geq 12
\end{aligned}\right.$$

These second-order differential equations may be integrated using the same numerical integration technique as is used for the spacecraft position vector. The vectors $\underline{w}_{3,i}$ and $\underline{w}_{9,i}$ bear the same relationship to the spacecraft velocity vector as the vectors $\underline{w}_{0,i}$ and $\underline{w}_{6,i}$ bear to the spacecraft position vector, and $\underline{w}_{3,i}$ and $\underline{w}_{9,i}$ are a by-product of the numerical integration of $\underline{w}_{0,i}$ and $\underline{w}_{6,i}$ just as the velocity vector is a by-product of the numerical integration of the position vector.

4.3 Numerical Integration Method

The extrapolation of inertial state vectors and filter weighting matrices requires the numerical solution of two second-order vector differential equations, which are special cases of the general form

$$\frac{d^2}{dt^2}\,\underline{y}(t) = \underline{f}(t,\ \underline{y}(t),\ \underline{z}(t))$$

where

$$\underline{z} = \frac{d}{dt}\,\underline{y}\,.$$

Nystrom's standard fourth-order method is utilized to numerically solve this equation. The algorithm for this method is:

$$\underline{y}_{n+1} = \underline{y}_n + \underline{z}_n\,\Delta t + \frac{1}{6}(\underline{k}_1 + \underline{k}_2 + \underline{k}_3)(\Delta t)^2$$

$$\underline{z}_{n+1} = \underline{z}_n + \frac{1}{6}(\underline{k}_1 + 2\underline{k}_2 + 2\underline{k}_3 + \underline{k}_4)\,\Delta t$$

$$\underline{k}_1 = \underline{f}\,(t_n,\ \underline{y}_n,\ \underline{z}_n)$$

$$\underline{k}_2 = \underline{f}\,(t_n + \frac{1}{2}\Delta t,\ \underline{y}_n + \frac{1}{2}\underline{z}_n\,\Delta t + \frac{1}{8}\underline{k}_1(\Delta t)^2,\ \underline{z}_n + \frac{1}{2}\underline{k}_1\,\Delta t)$$

$$\underline{k}_3 = \underline{f}\,(t_n + \frac{1}{2}\Delta t,\ \underline{y}_n + \frac{1}{2}\underline{z}_n\,\Delta t + \frac{1}{8}\underline{k}_1(\Delta t)^2,\ \underline{z}_n + \frac{1}{2}\underline{k}_2\,\Delta t)$$

$$\underline{k}_4 = \underline{f}\,(t_n + \Delta t,\ \underline{y}_n + \underline{z}_n\,\Delta t + \frac{1}{2}\underline{k}_3(\Delta t)^2,\ \underline{z}_n + \underline{k}_3\,\Delta t)$$

where

$$\underline{y}_n = \underline{y}(t_n),\ \underline{z}_n = \underline{z}(t_n)$$

and

$$t_{n+1} = t_n + \Delta t$$

As can be seen, the method requires four evaluations of $\underline{f}(t, \underline{y}, \underline{z})$ per integration step Δt as does the classical fourth-order Runge-Kutta method when it is extended to second-order equations. However, if $\underline{f}$ is independent of $\underline{z}$, then Nystrom's method above only requires three evaluations per step since $\underline{k}_3 = \underline{k}_2$. (Runge-Kutta's method will still require four).

The integration time step Δt may be varied from step to step. The nominal integration step size is

$$\Delta t_{nom} = c_{nom} \; r_{con}{}^{3/2}/\sqrt{\mu}$$

where c_{nom} is a program constant. (The value $c_{nom} = 0.3$ is recommended and implies that about 21 steps will be taken per trajectory revolution). The actual step-size is however limited to a maximum of Δt_{max}, which is also a program constant. (A value of about 4000 seconds is suggested.) Also, in the last step, the actual step size is taken to be the interval between the end of the previous step and the desired integration endpoint, so that the extrapolated values of the state or W-matrix are immediately available. Thus the integration step-size Δt is given by the formula

$$\Delta t = \pm \text{ minimum } (|t_F - t|, \Delta t_{nom}, \Delta t_{max})$$

where t_F is the desired integration end-point and t is the time at the end of the previous step. The plus sign is used if forward extrapolation is being performed, while the negative sign is used in the backdating case.

C.2

5. DETAILED FLOW DIAGRAMS

This section contains detailed flow diagrams of the Precision State and Filter Weighting Matrix Extrapolation Routine.

Each input and output variable in the routine and subroutine call statements can be followed by a symbol in brackets. This symbol identifies the notation for the corresponding variable in the detailed description and flow diagrams of the called routine. When identical notation is used, the bracketed symbol is omitted.

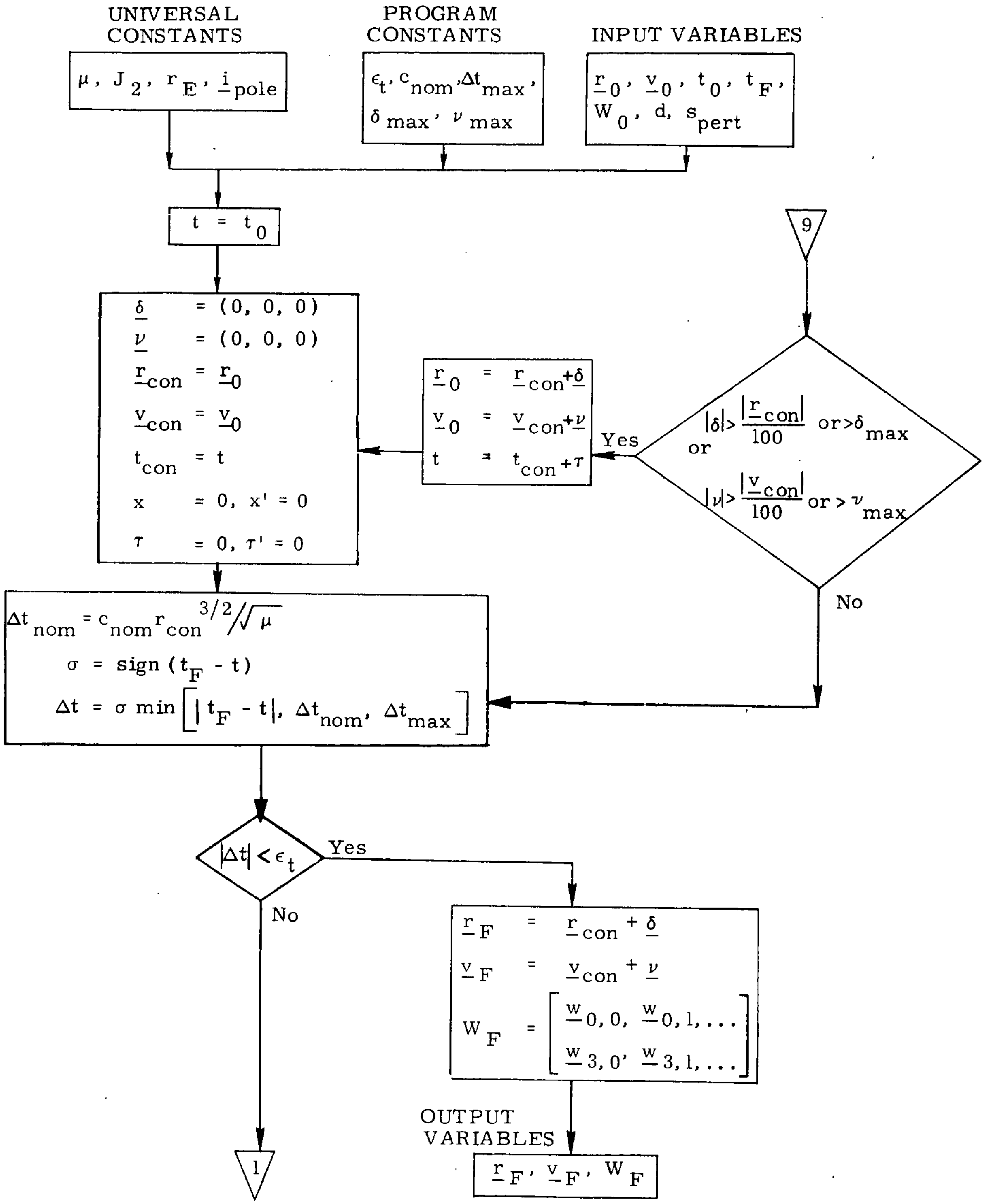

UNIVERSAL CONSTANTS
μ, J_2, r_E, $\underline{i}_{pole}$
PROGRAM CONSTANTS
ϵ_t, c_{nom}, Δt_{max}, δ_{max}, ν_{max}
INPUT VARIABLES
$\underline{r}_0$, $\underline{v}_0$, t_0, t_F, W_0, d, s_{pert}
$t = t_0$
$\underline{\delta} = (0, 0, 0)$
$\underline{\nu} = (0, 0, 0)$
$\underline{r}_{con} = \underline{r}_0$
$\underline{v}_{con} = \underline{v}_0$
$t_{con} = t$
$x = 0$, $x' = 0$
$\tau = 0$, $\tau' = 0$
9
$\underline{r}_0 = \underline{r}_{con} + \underline{\delta}$
$\underline{v}_0 = \underline{v}_{con} + \underline{\nu}$
$t = t_{con} + \tau$
Yes
$|\delta| > \frac{|\underline{r}_{con}|}{100}$ or $> \delta_{max}$
or
$|\nu| > \frac{|\underline{v}_{con}|}{100}$ or $> \nu_{max}$
No
$\Delta t_{nom} = c_{nom} r_{con}^{3/2} / \sqrt{\mu}$
$\sigma = \text{sign}(t_F - t)$
$\Delta t = \sigma \min\left[|t_F - t|, \Delta t_{nom}, \Delta t_{max}\right]$
$|\Delta t| < \epsilon_t$
Yes
No
$\underline{r}_F = \underline{r}_{con} + \underline{\delta}$
$\underline{v}_F = \underline{v}_{con} + \underline{\nu}$
$W_F = \begin{bmatrix} \underline{w}_{0,0}, & \underline{w}_{0,1}, \dots \\ \underline{w}_{3,0}, & \underline{w}_{3,1}, \dots \end{bmatrix}$
OUTPUT VARIABLES
$\underline{r}_F$, $\underline{v}_F$, W_F
1

Figure 2a. Detailed Flow Diagram

9.6.1.2 Precision State Extrapolation (continued)

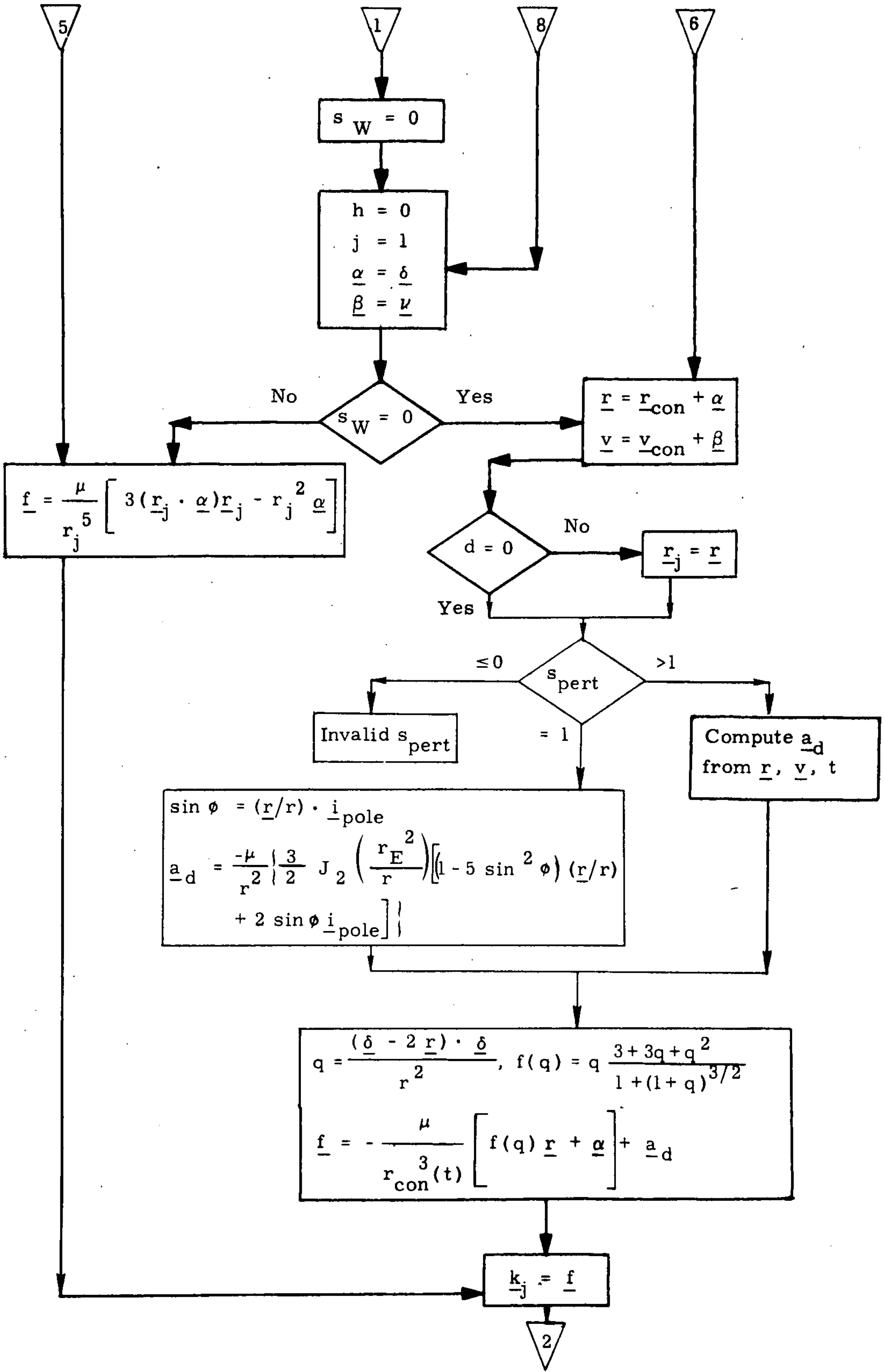

5
1
8
6
$s_W = 0$
$h = 0$
$j = 1$
$\underline{\alpha} = \underline{\delta}$
$\underline{\beta} = \underline{\nu}$
No
$s_W = 0$
Yes
$\underline{r} = \underline{r}_{con} + \underline{\alpha}$
$\underline{v} = \underline{v}_{con} + \underline{\beta}$
$\underline{f} = \frac{\mu}{r_j^5}\left[3(\underline{r}_j \cdot \underline{\alpha})\underline{r}_j - r_j^2\,\underline{\alpha}\right]$
$d = 0$
No
$\underline{r}_j = \underline{r}$
Yes
≤ 0
s_{pert}
>1
Invalid s_{pert}
$= 1$
Compute $\underline{a}_d$ from $\underline{r}$, $\underline{v}$, t
$\sin\phi = (\underline{r}/r) \cdot \underline{i}_{pole}$
$\underline{a}_d = \frac{-\mu}{r^2}\left\{\frac{3}{2} J_2 \left(\frac{r_E}{r}\right)^2 \left[\left(1 - 5\sin^2\phi\right)(\underline{r}/r) + 2\sin\phi\,\underline{i}_{pole}\right]\right\}$
$q = \frac{(\underline{\delta} - 2\,\underline{r}) \cdot \underline{\delta}}{r^2}$, $f(q) = q\,\frac{3 + 3q + q^2}{1 + (1+q)^{3/2}}$
$\underline{f} = -\frac{\mu}{r_{con}^3(t)}\left[f(q)\,\underline{r} + \underline{\alpha}\right] + \underline{a}_d$
$\underline{k}_j = \underline{f}$
2

Figure 2b. Detailed Flow Diagram

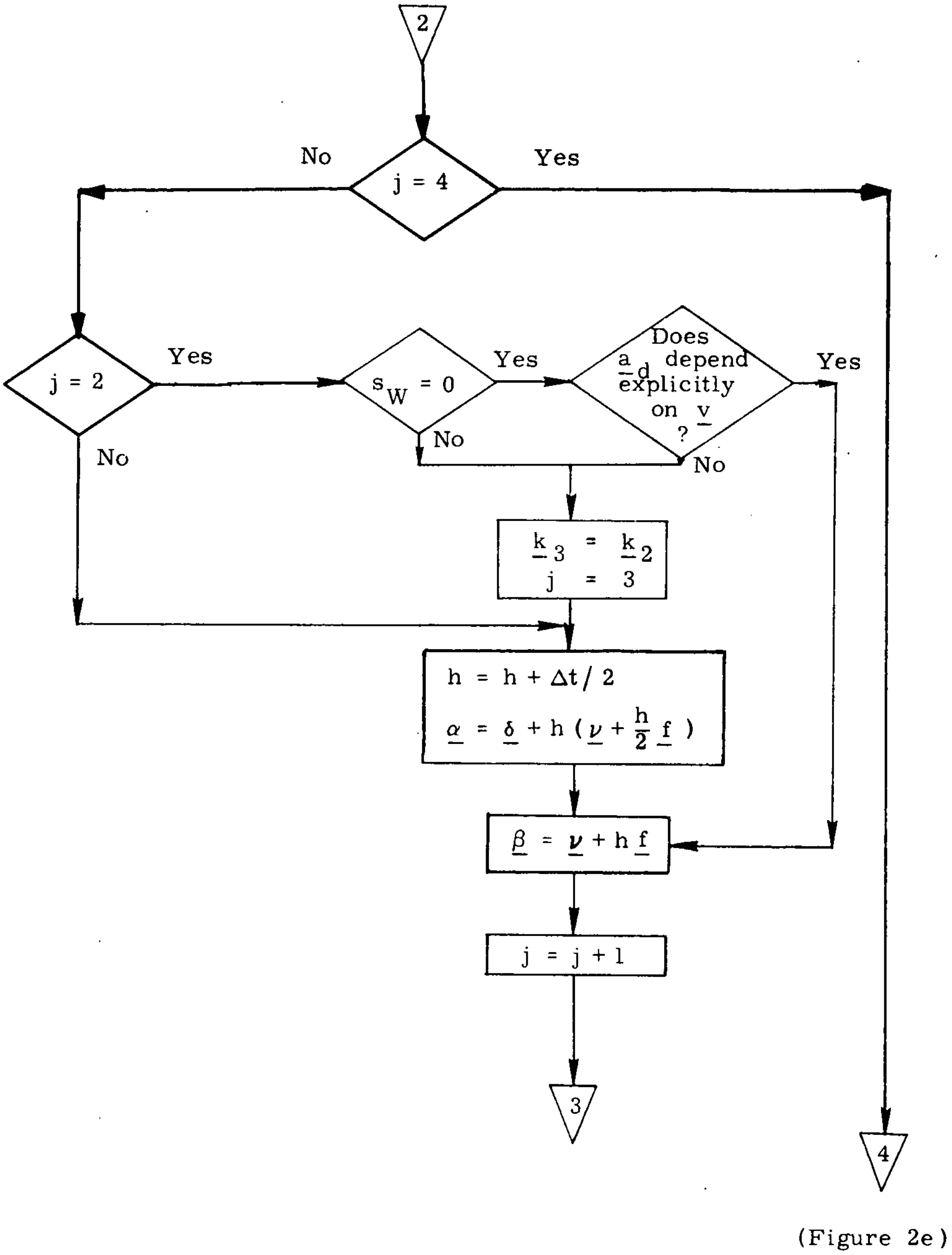

2
No
j = 4
Yes
j = 2
Yes
No
$s_W = 0$
Yes
No
Does $\underline{a}_d$ depend explicitly on $\underline{v}$?
Yes
No
$\underline{k}_3 = \underline{k}_2$
j = 3
$h = h + \Delta t/2$
$\underline{\alpha} = \underline{\delta} + h(\underline{\nu} + \frac{h}{2}\underline{f})$
$\underline{\beta} = \underline{\nu} + h\underline{f}$
j = j + 1
3
4
(Figure 2e)

Figure 2c. Detailed Flow Diagram

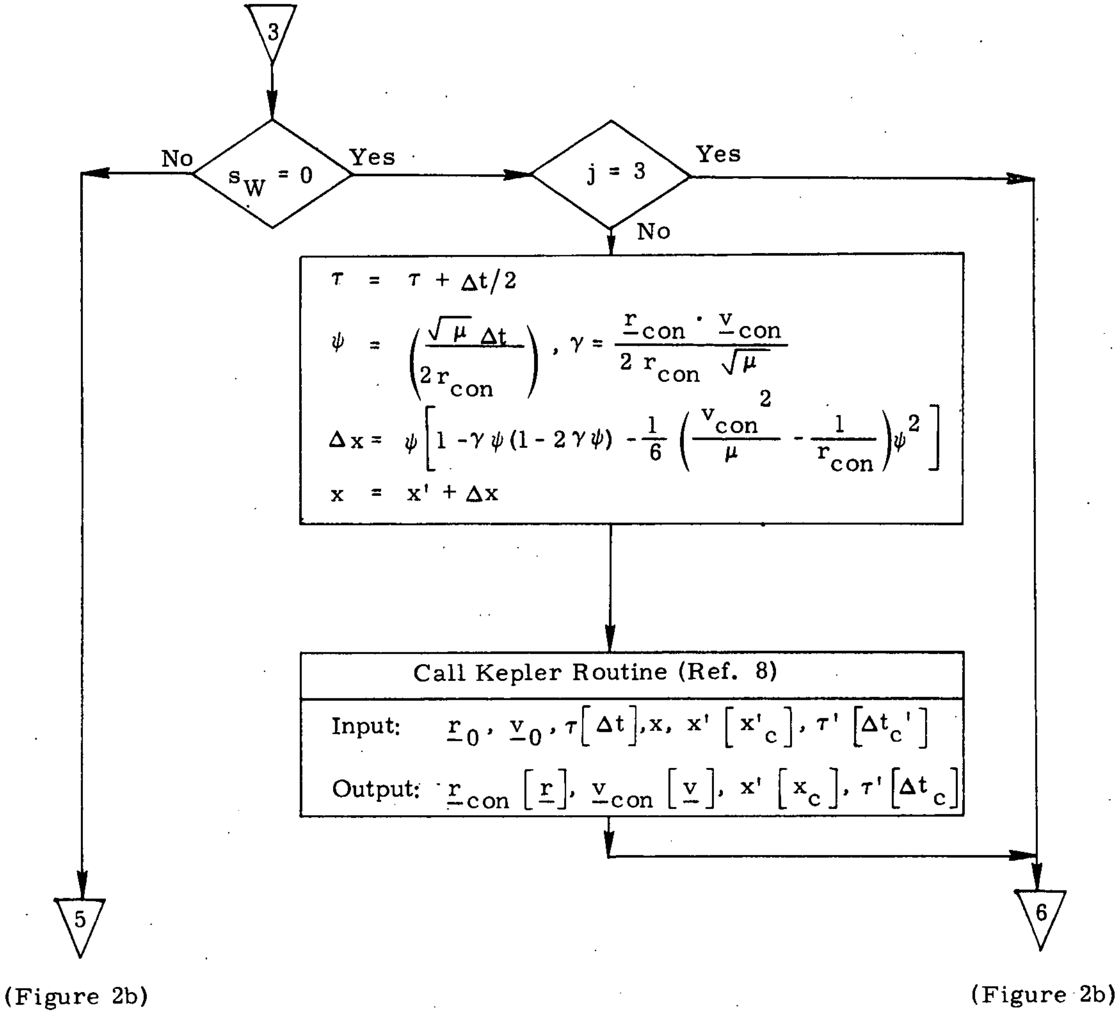

3
$s_W = 0$
No
Yes
$j = 3$
Yes
No
$\tau = \tau + \Delta t/2$
$\psi = \left(\frac{\sqrt{\mu}\,\Delta t}{2 r_{con}}\right), \gamma = \frac{\underline{r}_{con} \cdot \underline{v}_{con}}{2 r_{con} \sqrt{\mu}}$
$\Delta x = \psi \left[1 - \gamma \psi (1 - 2 \gamma \psi) - \frac{1}{6} \left(\frac{v_{con}^2}{\mu} - \frac{1}{r_{con}} \right) \psi^2 \right]$
$x = x' + \Delta x$
Call Kepler Routine (Ref. 8)
Input: $\underline{r}_0$, $\underline{v}_0$, $\tau [\Delta t]$, x, $x' [x'_c]$, $\tau' [\Delta t_c']$
Output: $\underline{r}_{con} [\underline{r}]$, $\underline{v}_{con} [\underline{v}]$, $x' [x_c]$, $\tau' [\Delta t_c]$
5
(Figure 2b)
6
(Figure 2b)

Figure 2d. Detailed Flow Diagram

9.6.1.2 Precision State Extrapolation (continued)

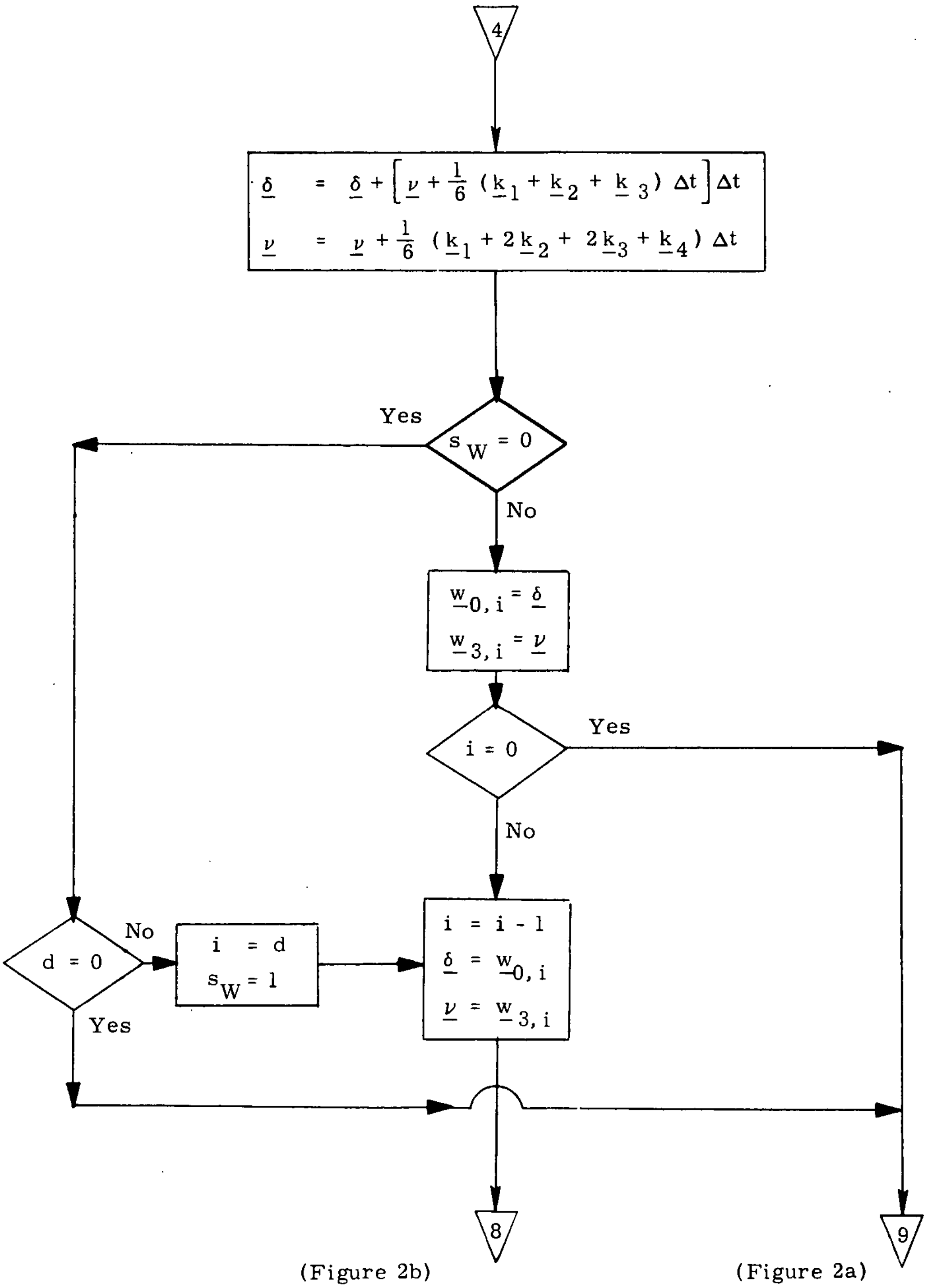

4
$\underline{\delta} = \underline{\delta} + \left[\underline{\nu} + \frac{1}{6}(\underline{k}_1 + \underline{k}_2 + \underline{k}_3)\,\Delta t\right]\Delta t$
$\underline{\nu} = \underline{\nu} + \frac{1}{6}(\underline{k}_1 + 2\underline{k}_2 + 2\underline{k}_3 + \underline{k}_4)\,\Delta t$
Yes
$s_W = 0$
No
$\underline{w}_{0,i} = \underline{\delta}$
$\underline{w}_{3,i} = \underline{\nu}$
Yes
$i = 0$
No
No
$d = 0$
$i = d$
$s_W = 1$
$i = i - 1$
$\underline{\delta} = \underline{w}_{0,i}$
$\underline{\nu} = \underline{w}_{3,i}$
Yes
8
9
(Figure 2b)
(Figure 2a)

Figure 2e. Detailed Flow Diagram

6. SUPPLEMENTARY INFORMATION

Encke's technique is a classical method in astrodynamics and is described in all standard texts, for example Battin (1964). The f(q) function used in Encke's technique (and in the lunar-solar perturbing acceleration computations) has generally been evaluated by a power series expansion; the closed form expression given here was derived by Potter, and is described in Battin (1964).

The oblateness acceleration in terms of a general spherical harmonic expansion may be calculated in a variety of ways; three different recursive algorithms are given in Gulick (1970). For low order expansions, especially those involving mostly zonal terms, an explicit formulation is generally superior computation-time-wise, as only the non-zero terms enter into the calculation. The general expression for the zonal terms is given by Battin (1964), while Zeldin and Robertson (1970) give explicit analytic expressions for each of the tesseral terms up through fifth order; hence all combinations of terms may easily be included in the oblateness acceleration by consulting the formulations in these references.

A full discussion of the use of covariance matrices in space navigation is given in Battin (1964). Potter (1963) suggested the use of the W-matrix and developed several of its properties. It should be noted that strictly the gravity gradient matrix **G(t)** should also include the gradient of the perturbing acceleration; however, these terms are so small that they may be neglected for our purposes. The use of only the conic gravity gradient, however, does not imply the W-matrix is being extrapolated conically. (Conic extrapolation of the W-matrix can be performed by premultiplying the W-matrix by the conic state transition matrix, which can be expressed in closed form). Rather the W-matrix is here extrapolated along the precision (perturbed) trajectory, as can be seen from the detailed flow diagram of Section 5.

9.5.1.2 Precision State Extrapolation (continued)

The Nystrom numerical integration technique was first conceived by Nystrom (1925), and is described in all standard texts on the numerical integration of ordinary differential equations, such as Henrici (1962). Parametric studies carried out by Robertson (1970) on the general fourth-order Runge-Kutta and Nystrom integration techniques indicate that the "classic" techniques are the best overall techniques for a variety of earth orbiting trajectories in the sense of minimizing the terminal position error for all the trajectories, although for any one trajectory a special technique can generally be found which decreases the position error after ten steps by one or two orders of magnitude for only that trajectory. The classical fourth-order Runge-Kutta and Nystrom techniques are approximately equally accurate, but the latter possesses the computational advantage of requiring one less perturbing acceleration evaluation per step when the perturbing acceleration is independent of the velocity. This fact has been taken into account in the detailed flow diagram of Section 5, in that the extra evaluation is performed only when the perturbing acceleration depends explicitly on the velocity. Some past Apollo experience has suggested that extra evaluation effect with drag is so small as to be negligible; further analysis will confirm or deny this for the Space Shuttle. In regard to step-size, the constants and the functional form of the nominal and maximum time-step expressions have been determined by Marscher (1965).

References

1. Battin, R.H., 1964, Astronautical Guidance, McGraw-Hill.

2. Gulick, L.J., 1970, A Comparison of Methods for Computing Gravitational Potential Derivatives, Environmental Science Services Administration Technical Report C & GS 40, Rockville Maryland, September 1970.

3. Henrici, P., 1962, Discrete Variable Methods in Ordinary Differential Equations, Wiley & Sons.

4. Marscher, W.F., 1965 , A Modified Encke, MIT/IL Space Guidance Analysis Memo #2-65, January 1965.

5. Nystrom, E.J., 1925, Uber die Numerische Integration von Differentialgleichungen, Acta Soc. Sci., Fenn, 50 (1925), No. 13, 1-55.

6. Potter, J.E., 1963, New Statistical Formulas, MIT/IL Space Guidance Analysis Memo #40, April 1963.

7. Robertson, W.M., 1970, Parametric Investigations of the Free Parameters in Fourth-Order Runge-Kutta and Nystrom Integration Techniques, Supplement to MIT/DL Report # DLMC-70-3, June 1970.

8. Robertson, W.M., 1971, Conic State Extrapolation, MIT/DL Space Shuttle GN & C Equation Document No. 3-71, February 1971.

9. Zeldin, S. and Robertson, W.M., 1970, Explicit Analytic Expressions for the Terms in a Spherical Harmonic Expansion of the Gravitational Acceleration due to Oblateness, MIT/DL Space Guidance Analysis Memo #4-70, May 1970.

9.6.1.3 ORBIT NAVIGATION USING NAVIGATION SENSORS

SPACE SHUTTLE

GN&C SOFTWARE EQUATION SUBMITTAL

Software Equation Section: Orbit Navigation using Navigation Sensors

Submittal No. 23

Function: Automatically use navigation sensor data to reduce uncertainties in on-board state estimate.

Module No. ON 2   Function No. 2 (MSC 03690)

Submitted by: E. S. Muller, Jr.   Co. MIT No. 9-71

Date: 1 July 1971

NASA Contact: I. Caro   Organization EG2

Approved by Panel III: K.J. Cox   Date: 7/1/71

Summary Description: Equations are provided required for horizon sensing systems and a ground beacon orbit navigation system. With minor modifications, the equations may be readily adapted to other orbit navigation systems such as known or unknown landmark tracking, or to systems using different sensors, e.g. radar altimeter.

Shuttle Configuration: This software is essentially independent of shuttle configuration.

Comments:

(Design Status) ______

(Verification Status) ______

Panel Comments:

1. INTRODUCTION AND FUNCTIONAL FLOW DIAGRAM

The purpose of the Orbit Navigation function is to provide a means of automatically reducing uncertainties in the on-board knowledge of the SSV (primary vehicle) inertial state by accepting and processing data from the navigation sensor(s). This knowledge is required to (a) accurately compute orbital maneuvers, (b) provide accurate initial conditions for other mission phases such as rendezvous, deorbit and landing.

There are several candidate orbit navigation systems for the shuttle mission e.g.:

1. horizon sensing
2. tracking ground based beacons
3. tracking navigation satellites
4. tracking satellites ejected from the primary vehicle

The navigation equations required for systems (3) or (4) fall in the category of relative state updating and are documented in Ref. 1. This document will present the equations required for horizon sensing systems and a ground beacon orbit navigation system. In the horizon sensing system, the direction of the line-of-sight to a horizon is measured with respect to inertially fixed coordinates provided by the inertial measurement unit (IMU). In the ground beacon system, transponders located at known positions on the earth are interrogated by the SSV navigation system. The return signals from the transponders provide range to the beacon and/ or range rate relative to the beacon.

With minor modifications, the equations presented for these two systems may be readily adapted to other orbit navigation systems, such as known or unknown landmark tracking, or to systems using different navigation sensors, e.g. radar altimeter.

A general flow diagram of this function is presented in Fig. 1. The inputs required by this function are:

1. On-board estimate of primary vehicle state ($\underline{x}_P$) with time tag.
2. Initial filter weighting matrix (W)
3. A priori sensor measurement variance
4. Navigation sensor measurements

   and if ground beacon navigation is used
5. Latitude and longitude of next ground beacon(s) encountered.

The output of this function is an updated estimate of the primary vehicle state. This output is available after each measurement incorporation.

The system depicted in Fig. 1 operates as follows: Navigation sensor data are accepted at discrete "measurement incorporation times". The estimate of the primary vehicle state is updated at each of these times by processing the sensor data in the measurement incorporation routine. If more than one piece of sensor data is to be incorporated at a given time (e.g. range and range rate relative to a ground beacon) each piece of data is incorporated independently in a sequential fashion.

A precision extrapolation routine extrapolates the primary vehicle state and filter weighting matrix from one "measurement incorporation time" to the next. This routine is described in Ref. 2. For the ground beacon navigation system, a prediction scheme is described which determines which of the ground beacons stored in a catalog will be encountered next by the primary vehicle, and at what time this will occur so that the on-board interrogator may be turned on a sufficient time prior to this encounter.

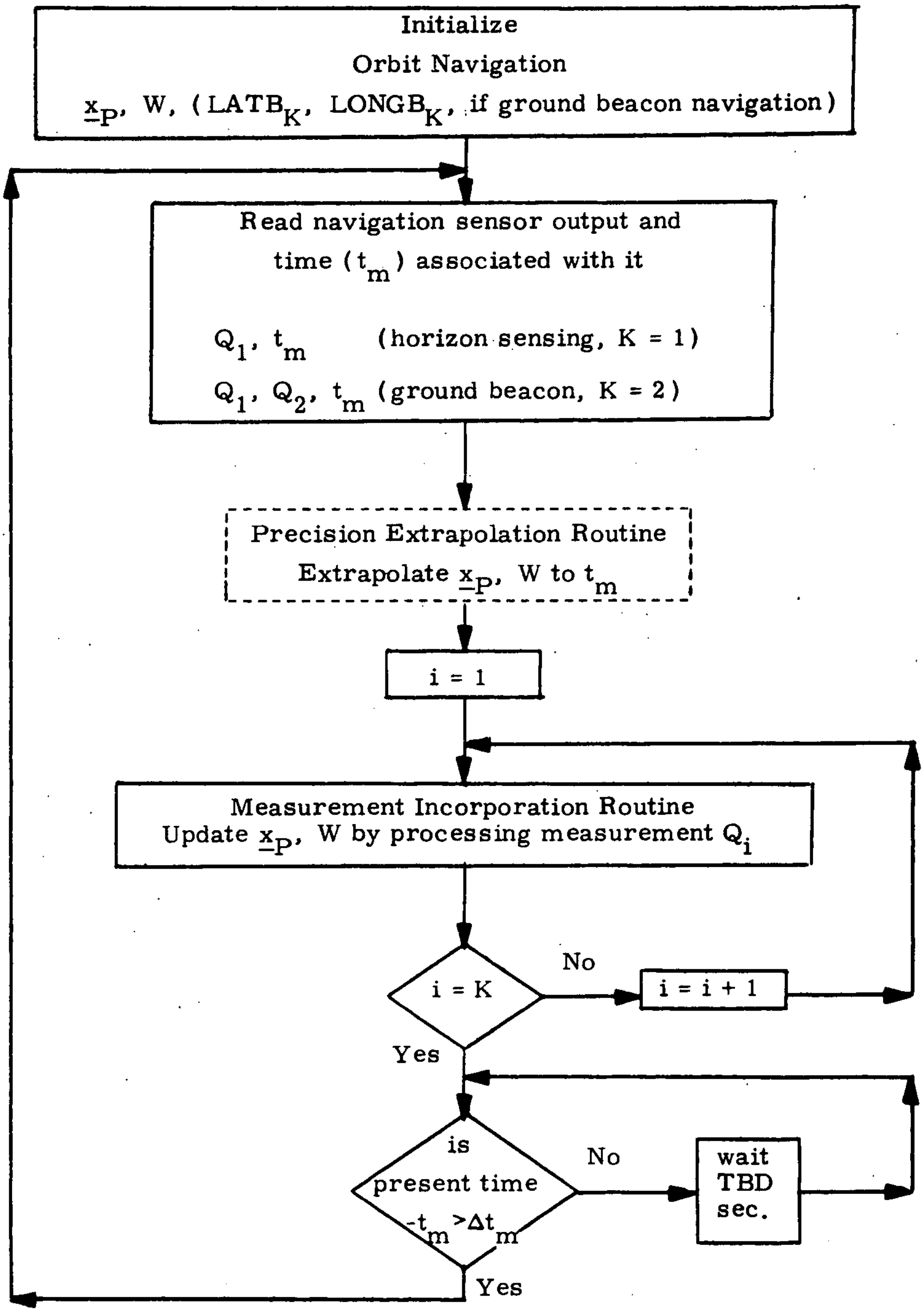

Initialize
Orbit Navigation
$\underline{x}_P$, W, (LATB$_K$, LONGB$_K$, if ground beacon navigation)
Read navigation sensor output and
time (t_m) associated with it
Q_1, t_m (horizon sensing, K = 1)
Q_1, Q_2, t_m (ground beacon, K = 2)
Precision Extrapolation Routine
Extrapolate $\underline{x}_P$, W to t_m
i = 1
Measurement Incorporation Routine
Update $\underline{x}_P$, W by processing measurement Q_i
i = K
No
i = i + 1
Yes
is present time $-t_m > \Delta t_m$
No
wait TBD sec.
Yes

Figure 1 ORBIT NAVIGATION FLOW DIAGRAM

9.6.1.3 Orbit Navigation Using Navigation Sensors (continued)

NOMENCLATURE

$\underline{a} = \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix}$ Vector

a Magnitude of vector $\underline{a}$

UNIT $(\underline{a})$ Unit vector $(\underline{a}/a)$

a_F Fischer ellipsoid semi-major axis in equatorial plane

A_Z Angle between reference frame x axis and earth fixed frame x axis (zero longitude in equatorial plane) at launch epoch (t_0)

$b_F(I)$ Radius of Fischer ellipsoid which corresponds to a latitude equal to I

$\underline{b} = \begin{pmatrix} \underline{b}_{P,0} \\ \underline{b}_{P,3} \end{pmatrix}$ 6-dimensional measurement geometry vector

$\underline{b}_{P,0}$, $\underline{b}_{P,3}$ 3-dimensional measurement geometry vectors associated with $\underline{r}_P$, $\underline{v}_P$

I Inclination angle between horizon measurement plane and equatorial plane

ID Beacon identification code of next beacon encountered

ID_j Beacon identification code of j th beacon encountered within 15 minutes of last beacon (j = 1,2)

$LATB_K$, $LONGB_K$ Latitude, longitude of Kth ground beacon of total of n beacons (K = 1, 2, 3, ... n)

M_{NB-m} Transformation matrix from navigation base axes to navigation sensor axes. M_{NB-m} is fixed according to spacecraft configuration.

M_{E-R} Transformation matrix from earth fixed frame (z-north pole, x-in equatorial plane at 0^{o} long., y-completes right hand system) to reference coordinate frame (in which initial state is expressed and computations are performed). M_{E-R} is determined from A_Z initially and is updated using earth spin rate and elapsed mission time.

M_{R-SM} Transformation matrix from reference coordinate frame to stable member axes. M_{R-SM} is given from specified platform alignment.

M_{SM-NB} Transformation matrix from stable member axes to navigation base axes on which IMU is mounted. M_{SM-NB} is determined from IMU gimbal angles.

n total number of ground beacons

nv number of ground beacons visible within 15 minutes of each other

Q_{EST} On-board estimate of measured parameter

Q_i ith measured parameter at t_m

$\underline{r}_B$ Ground beacon position vector

RB_K Altitude of Kth ground beacon

$\underline{R}_{BP}$ Position vector of $\underline{r}_B$ relative to $\underline{r}_P$

$\underline{r}_H$ Horizon position vector from earth center

$\underline{r}_{PH}$ Position vector of $\underline{r}_H$ relative to $\underline{r}_P$

$\underline{r}_P$ Primary vehicle position vector

ORBWFLAG "0" - W is left as extrapolated from Precision Integration routine (initially set to "0").

"1" - W is set to pre-loaded value given by W_F

9.6.1.3 Orbit Navigation Using Navigation Sensors (continued)

t_0	Launch epoch
t	Present time
t_m	Measurement Incorporation time
t_S	Time of initiation of ground beacon search
TI, TI_j	Ground beacon search initiation time of next beacon encountered, of j th beacon encountered within 15 minutes of last beacon (j = 1,2).
$\underline{V}_{BP}$	Velocity vector of ground beacon relative to $\underline{v}_P$
VAR	A priori filter measurement error variance
VAR_ψ	A priori random measurement error variance for horizon angle ψ
VAR_H	A priori random horizon threshold variance
W	6 × 6 filter weighting matrix associated with $\underline{x}_P$
W_I	Pre-loaded value of initial filter weighting matrix
W_F	Pre-loaded value to which W is reinitialized
$\underline{x}_P = \begin{pmatrix} \underline{r}_P \\ \underline{v}_P \end{pmatrix}$	6-dimensional primary vehicle state vector
γ_H	Pre-stored horizon threshold altitude
Δt_m	Time increment between measurement incorporation times
$\delta \underline{x}$	6-dimensional navigation update of $\underline{x}_P$
δ_B	Angle above horizontal at which ground beacon is "visible".

9.6.1.3 Orbit Navigation Using Navigation Sensors (continued)

μ	Gravitational constant
τ	Orbital period
ϕ	Pre-loaded horizon direction azimuth angles in vehicle local horizontal plane, measured from foward direction
ψ	Angle from navigation sensor boresight axis (x_m) to horizon
$\underline{\omega}_E$	Earth spin rate vector

2. DESCRIPTION OF EQUATIONS

The equations involved in the Measurement Incorporation Routine of the orbit navigation function are described in this section. In addition, equations are described for the prediction scheme required with a ground beacon orbit navigation system. Except where specifically noted, measurement incorporation equations are applicable to both horizon sensing and ground beacon orbit navigation systems.

2.1 Ground Beacon Prediction Routine

In a ground beacon orbit navigation system, a network of ground beacons (transponders) will be strategically located on the earth. The locations (latitude, longitude) of these beacons will be stored in the on-board computer. The function of the Ground Beacon Prediction Routine is to (a) provide an estimate of the time the primary vehicle will be in "viewing" range of the next ground beacon so that the on-board beacon interrogator may be activated prior to this time and (b) provide the measurement incorporation routine the coordinates of the next beacon encountered.

The beacon prediction scheme to be described consists mainly of logic statements with few equations. Thus, the detailed description will be left for the flow diagram section and a brief word description will be given here.

At a prescribed time (t_S), the beacon search is initiated. In order to save computer time, conic extrapolation of a dummy state vector (to preserve the permanent state vector) is utilized in the search routine. The search interval is constrained in order to minimize the error resulting from conic approximation. This constraint is achieved by specifying a maximum search interval of 1/2 orbit, and reinitiating the search 1/4 orbit later. In this manner, closely spaced or overlapping beacons are not missed.

The dummy state is extrapolated in 1 minute intervals. At each interval all the stored beacon locations are examined for visibility until one passes the visibility criterion (i.e. the dummy state is within a cone shown in Fig. 2). After finding a visible beacon, the search is continued until one of the following constraints is violated:

(a) another beacon has not been found visible within 15 minutes of the first beacon intercept;

(b) three beacons have been found visible within 15 minutes of each other (the number three is arbitrary, but in the final beacon network, there will most likely not be over three closely spaced beacons).

If no visible beacons are found within 1/2 orbit, the search is stopped and reinitiated 1/4 orbit later.

After finding a visible beacon (or 2 or 3 closely spaced beacons), the on-board interrogator is turned on 10 minutes prior to the predicted intercept of the first visible beacon. After navigation updating across this beacon, either of two options is executed:

(a) if another beacon is predicted within 15 minutes of the previous encounter, the interrogator is informed of the next beacon identification and interrogation is initiated.

(b) if a visible beacon is not predicted within 15 minutes of the last beacon, the search is reinitiated 1 minute after the cessation of navigation updating.

The above scheme may not be the one ultimately coded for the Orbital Coast Navigation Module when the final beacon network has been established. It does, however, represent a "brute force" approach which is not overly expensive in computer time. (A half orbit search should take approximately 10-15 seconds assuming computer comparable to the AGC is utilized). For

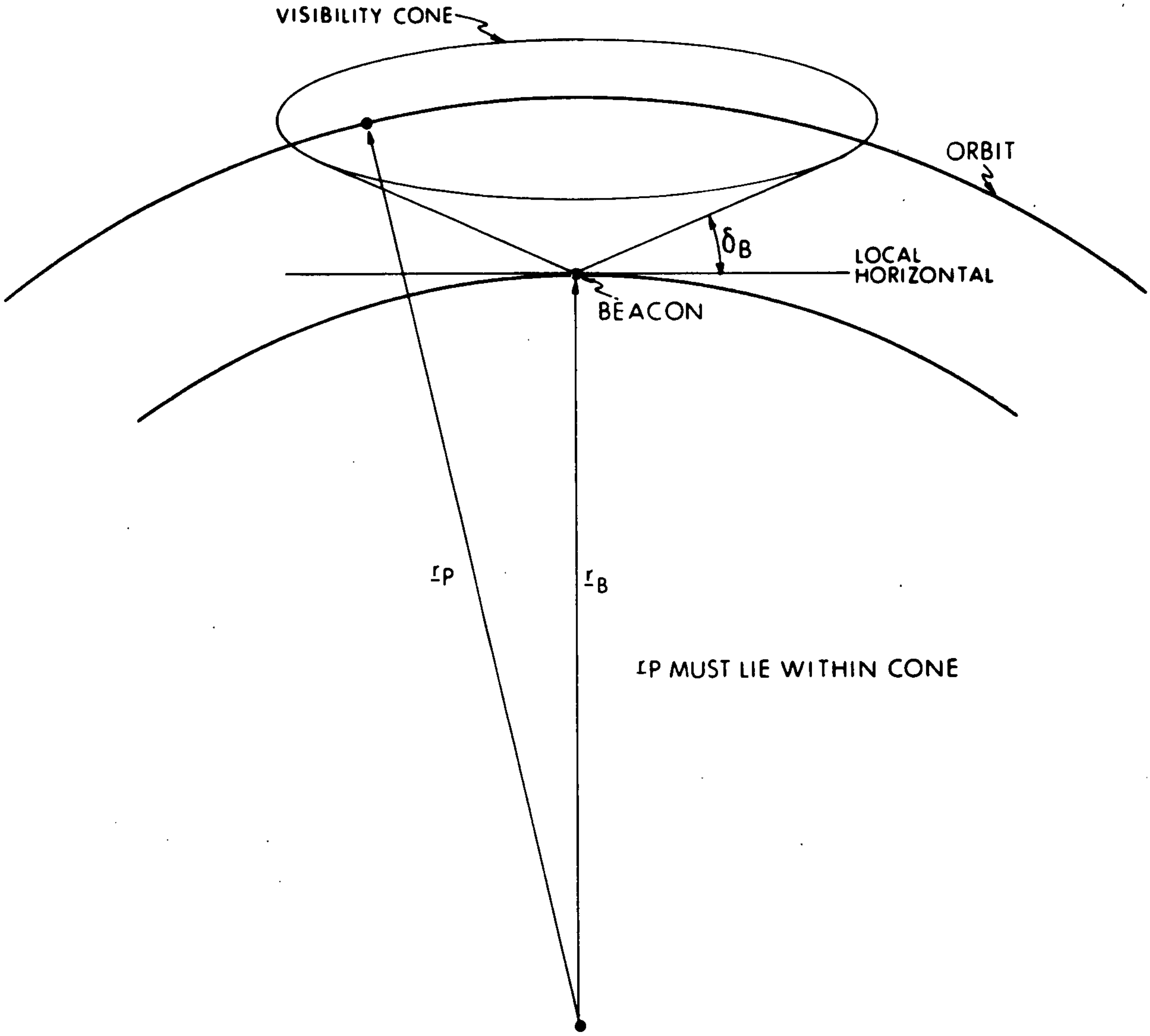


Figure 2. Beacon Visibility Constraint

example, if only U.S. based beacons are required, the search may be simplified considerably by just waiting some short time prior to stateside pass before determining the visible beacon(s) and turning on the interrogator.

2.2 Measurement Incorporation Routine

Computation of measurement geometry vector ($\underline{b}$), estimate of measured parameter (Q_{EST}) and measurement variance (VAR).

A. Ground Beacon Orbit Navigation

Compute position vector (reference coordinates) of the beacon being tracked from:

$$\underline{r}_B = M_{E-R} \quad RB_K \begin{pmatrix} \cos(LATB_K) \cos(LONGB_K) \\ \cos(LATB_K) \sin(LONGB_K) \\ \sin(LATB_K) \end{pmatrix}$$

Compute relative position vector from:

$$\underline{R}_{BP} = \underline{r}_B - \underline{r}_P$$

and

$$R_{BP} = \sqrt{R_{BP,0}^2 + R_{BP,1}^2 + R_{BP,2}^2}$$

$$U\underline{R}_{BP} = \underline{R}_{BP} / R_{BP}$$

A. 1 Range Measurement

Compute

$$\underline{b} = \begin{pmatrix} \underline{b}_{P,0} \\ \underline{b}_{P,3} \end{pmatrix}$$

from

$$\underline{b}_{P,0} = - \underline{UR}_{BP}$$

$$\underline{b}_{P,3} = \underline{0}$$

Compute Q_{EST} from:

$$Q_{EST} = R_{BP}$$

A. 2 Range Rate Measurement

Compute relative velocity vector from:

$$\underline{V}_{BP} = \underline{\omega}_E \times \underline{r}_B - \underline{v}_P$$

Compute $\underline{b}$ from

$$\underline{b}_{P,0} = \underline{UR}_{BP} \times (\underline{UR}_{BP} \times \underline{V}_{BP}) / R_{BP}$$

$$\underline{b}_{P,3} = - \underline{UR}_{BP}$$

Compute Q_{EST} from

$$Q_{EST} = \underline{V}_{BP} \cdot \underline{UR}_{BP}$$

Computation of VAR

The computation of VAR will depend on the error model ultimately formulated for ground beacon measurements. The current model assumes a constant value of VAR for either range or range rate measurements. This value will be prestored in the computer.

B. Horizon Sensing Orbit Navigation

Navigation analyses for SSV missions will determine the required directions for horizon measurements in order to achieve the desired performance without excessive attitude maneuvers. The horizon direction for a particular measurement may be described by an azimuth angle (ϕ) measured from the downrange direction in the local horizontal plane. (See Fig. 3).

Compute local vertical frame axes from:

$$U\underline{Z} = \text{UNIT}\,(\underline{r}_P)$$

$$U\underline{Y} = \text{UNIT}\,(\underline{r}_P \times \underline{v}_P)$$

$$U\underline{X} = U\underline{Y} \times U\underline{Z}$$

Computation of Vector to Horizon ($\underline{r}_{PH}$) (Ref. 4)

Using the current value of ϕ, from the pre-scheduled sequence of horizon measurement directions, compute:

$$\underline{\phi} = \cos\phi\, U\underline{X} + \sin\phi\, U\underline{Y} \text{ (Fig. 3)}$$

Define unit vector normal to horizon measurement plane from:

$$\underline{i}_2 = \text{UNIT}\,(\underline{\phi} \times \underline{r}_P)$$

7

and two orthogonal unit vectors in the horizon measurement plane from

$$\underline{i}_0 = \text{UNIT}\,(\underline{i}_Z \times \underline{i}_2)$$

$$\underline{i}_1 = \underline{i}_2 \times \underline{i}_0$$

($\underline{i}_0$ is in equatorial plane and $\underline{i}_Z$ is in direction of north pole given by third row of $M_{E-R}{}^T$)

Compute inclination angle between horizon measurement plane and equatorial plane from

$$I = \sin^{-1}(\underline{i}_1 \cdot \underline{i}_Z)$$

Compute $\underline{r}_P$ and $\underline{\phi}$ in horizon plane coordinate system ($\underline{i}_0$, $\underline{i}_1$, $\underline{i}_2$) from:

$$\underline{r} = \begin{pmatrix} \underline{i}_0{}^T \\ \underline{i}_1{}^T \\ \underline{i}_2{}^T \end{pmatrix} \underline{r}_P = M_{R-H}\,\underline{r}_P \triangleq \begin{pmatrix} X_H \\ Y_H \\ 0 \end{pmatrix}$$

$$\underline{\phi}_H = M_{R-H}\,\underline{\phi}$$

From the Fischer ellipsoid and a pre-stored horizon threshold altitude (γ_H), compute the semi-major and semi-minor axes of the horizon measurement plane ellipse from:

$$a_H = a_F + \gamma_H$$

$$b_H = b_F(I) + \gamma_H$$

where:

a_F is Fischer semi-major axis in equatorial plane

$b_F(I)$ radius of Fischer ellipsoid which corresponds to a latitude equal to I.

Compute the following quantities:

$$d = X_H^2 / a_H^2 + Y_H^2 / b_H^2$$

$$e = (a_H Y_H + \sqrt{d - 1}) / d b_H$$

$$f = (b_H X_H \sqrt{d - 1}) / d a_H$$

Compute horizon position vector from:

$$\underline{r}_H = \begin{pmatrix} X_{H/d} & + e \\ Y_{H/d} & - f \\ 0 & \end{pmatrix}$$

Compute vector to horizon from:

$$\underline{r}_{PH} = \underline{r}_H - \underline{r}_P$$

Compute elevation angle to horizon (ξ) (Fig. 4) from:

$$\xi = \cos^{-1} \left[\text{UNIT} (\underline{r}_{PH}) \cdot \underline{\phi}_H \right]$$

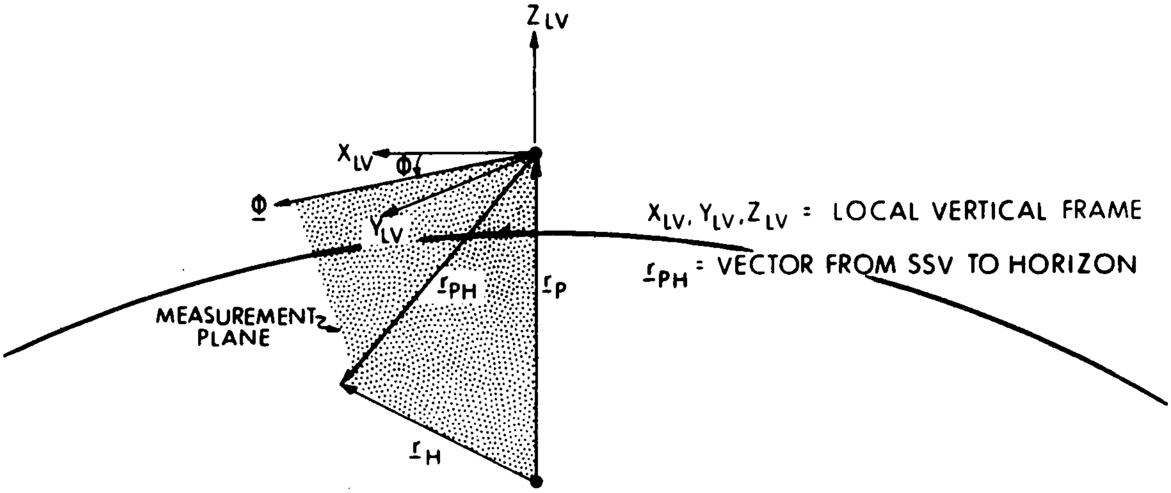


Figure 3. Definition of Horizon Measurement Plane

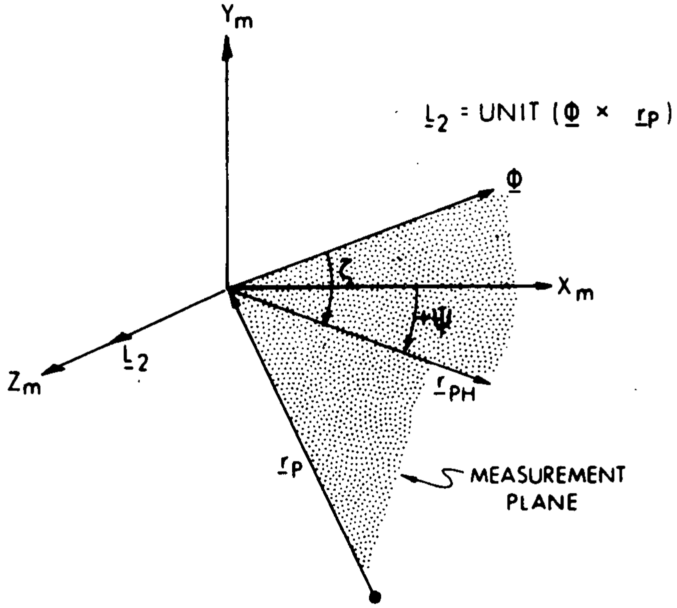


Figure 4. Horizon Sensor Coordinate Frame Geometry

If $\xi > \pi/2$ recompute $\underline{r}_H$, $\underline{r}_{PH}$ and ξ from:

$$\underline{r}_H \begin{pmatrix} X_{H/d} - e \\ Y_{H/d} + f \\ 0 \end{pmatrix}$$

$$\underline{r}_{PH} = \underline{r}_H - \underline{r}_P$$

$$\xi = \cos^{-1}\left[\text{UNIT}\,(\underline{r}_{PH}) \cdot \underline{\phi}_H\right]$$

Compute $\underline{r}_{PH}$ in reference coordinates from:

$$\underline{r}_{PH} = M_{R-H}^{\;T}\ \underline{r}_{PH}$$

$$u\underline{r}_{PH} = \text{UNIT}\,(\underline{r}_{PH})$$

From ϕ and ξ, vehicle attitude is adjusted so that sensor coordinate x_m (Fig. 4) is maintained in horizon measurement plane within sensor field of view from $\underline{r}_{PH}$.

Compute $\underline{x}_m$ in reference coordinates from:

$$\underline{x}_m = M_{SM-R}\ M_{NB-SM}\ M_{m-NB} \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

Compute measurement geometry vector $\left(\underline{b} = \begin{matrix} \underline{b}_{P,0} \\ \underline{b}_{P,3} \end{matrix} \right)$ from:

$$\underline{b}_{P,0} = \text{UNIT}\,(\underline{ur}_{PH} \times (\underline{r}_P \times \underline{ur}_{PH}))/\,r_{PH}$$

$$\underline{b}_{P,3} = \underline{0}$$

Compute Q_{EST} (angle ψ in Figure 4) from:

$$Q_{EST} = \cos^{-1}(\underline{x}_m \cdot \underline{ur}_{PH}) \left[\text{SIGN}((\underline{x}_m \times \underline{ur}_{PH}) \cdot (\underline{r}_P \times \underline{ur}_{PH})) \right]$$

Compute VAR from:

$$\text{VAR} = \text{VAR}_{\psi} + \text{VAR}_H /\, r_{PH}$$

State Vector and Filter Update at Measurement Incorporation Time

The filter weighting matrix (W) is available from one of the following sources:

At the first measurement incorporation:

1. Pre-loaded values based on mission simulations

Between measurement incorporations at a given t_m:

2. From the computation (below) after a measurement incorporation

At the first measurement incorporation of new t_m:

9.6.1.3 Orbit Navigation Using Navigation Sensors (continued)

3. From the Precision Extrapolation Routine

4. From pre-stored reinitialization values at prescribed reinitialization times.

Compute 6-dimensional $\underline{z}$ vector from:

$$\underline{z} = W^T \underline{b}$$

Compute 6-dimensional weighting vector, $\underline{\omega}$ from:

$$\underline{\omega} = \frac{1}{\underline{z} \cdot \underline{z} + VAR} W \underline{z}$$

Compute 6-dimensional navigation update of $\underline{x}_P$ from:

$$\delta\underline{x} = \underline{\omega} (Q_i - Q_{EST}) \begin{pmatrix} i = 1 & \text{horizon sensor} \\ i = 1, 2 & \text{ground beacon} \end{pmatrix}$$

Update $\underline{x}_P$ by:

$$\underline{x}_P = \underline{x}_P + \delta\underline{x}$$

Update W by:

$$W = W - \underline{\omega}\, \underline{z}^T / (1 + \left(\sqrt{\frac{VAR}{\underline{z} \cdot \underline{z} + VAR}} \right))$$

3. DETAILED FLOW DIAGRAMS

This section contains detailed flow diagrams for the Measurement Incorporation Routine of the Orbital Coast Navigation Module; and for use with ground beacon orbit navigation system, the prediction scheme for determining the next beacon encountered and the encounter time.

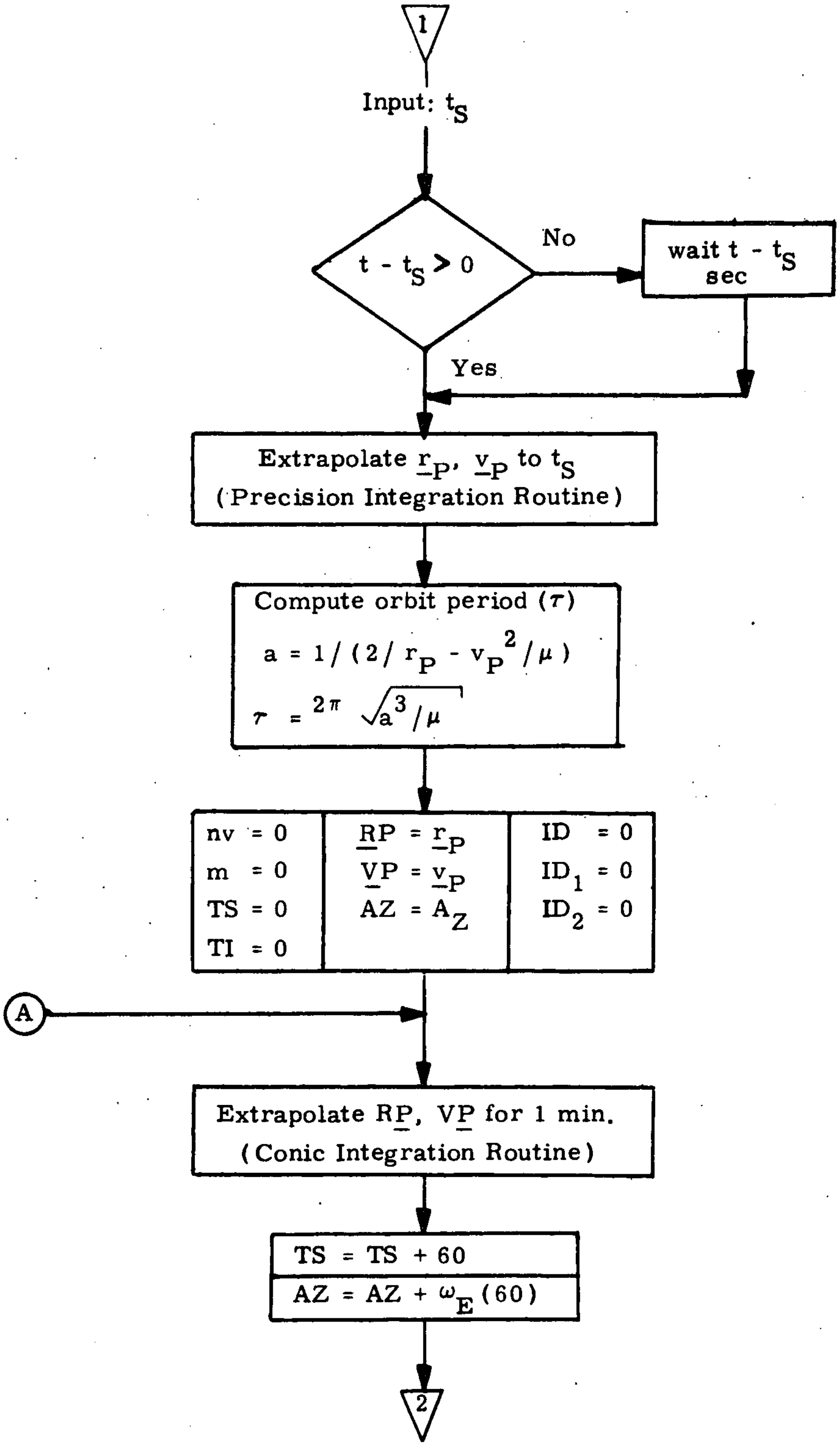

1
Input: t_S
$t - t_S > 0$
No
wait $t - t_S$ sec
Yes
Extrapolate $\underline{r}_P$, $\underline{v}_P$ to t_S
(Precision Integration Routine)
Compute orbit period (τ)
$a = 1/(2/r_P - v_P^2/\mu)$
$\tau = 2\pi\sqrt{a^3/\mu}$
nv = 0
m = 0
TS = 0
TI = 0
$\underline{RP} = \underline{r}_P$
$\underline{VP} = \underline{v}_P$
$AZ = A_Z$
ID = 0
$ID_1 = 0$
$ID_2 = 0$
A
Extrapolate $\underline{RP}$, $\underline{VP}$ for 1 min.
(Conic Integration Routine)
TS = TS + 60
$AZ = AZ + \omega_E (60)$
2

Figure 5a. Detailed Flow Diagram
Ground Beacon Prediction Routine

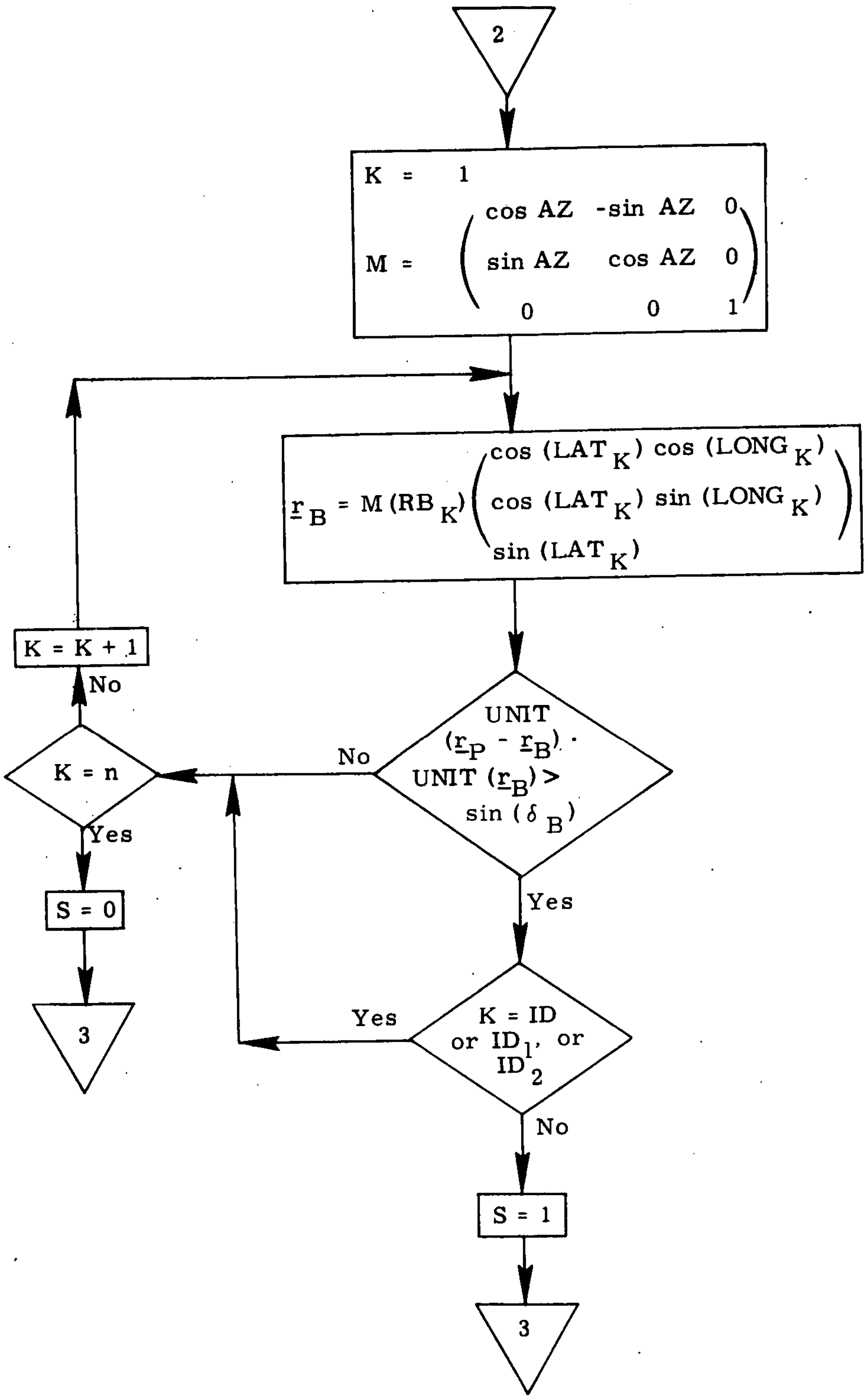

2
K = 1
M = $\begin{pmatrix} \cos AZ & -\sin AZ & 0 \\ \sin AZ & \cos AZ & 0 \\ 0 & 0 & 1 \end{pmatrix}$
$\underline{r}_B = M(RB_K) \begin{pmatrix} \cos(LAT_K)\cos(LONG_K) \\ \cos(LAT_K)\sin(LONG_K) \\ \sin(LAT_K) \end{pmatrix}$
UNIT $(\underline{r}_P - \underline{r}_B) \cdot$ UNIT $(\underline{r}_B) > \sin(\delta_B)$
No
Yes
K = n
No
K = K + 1
Yes
S = 0
3
K = ID or ID_1, or ID_2
Yes
No
S = 1
3

Figure 5b. Detailed Flow Diagram
Ground Beacon Prediction Routine
(Beacon Visibility Check)

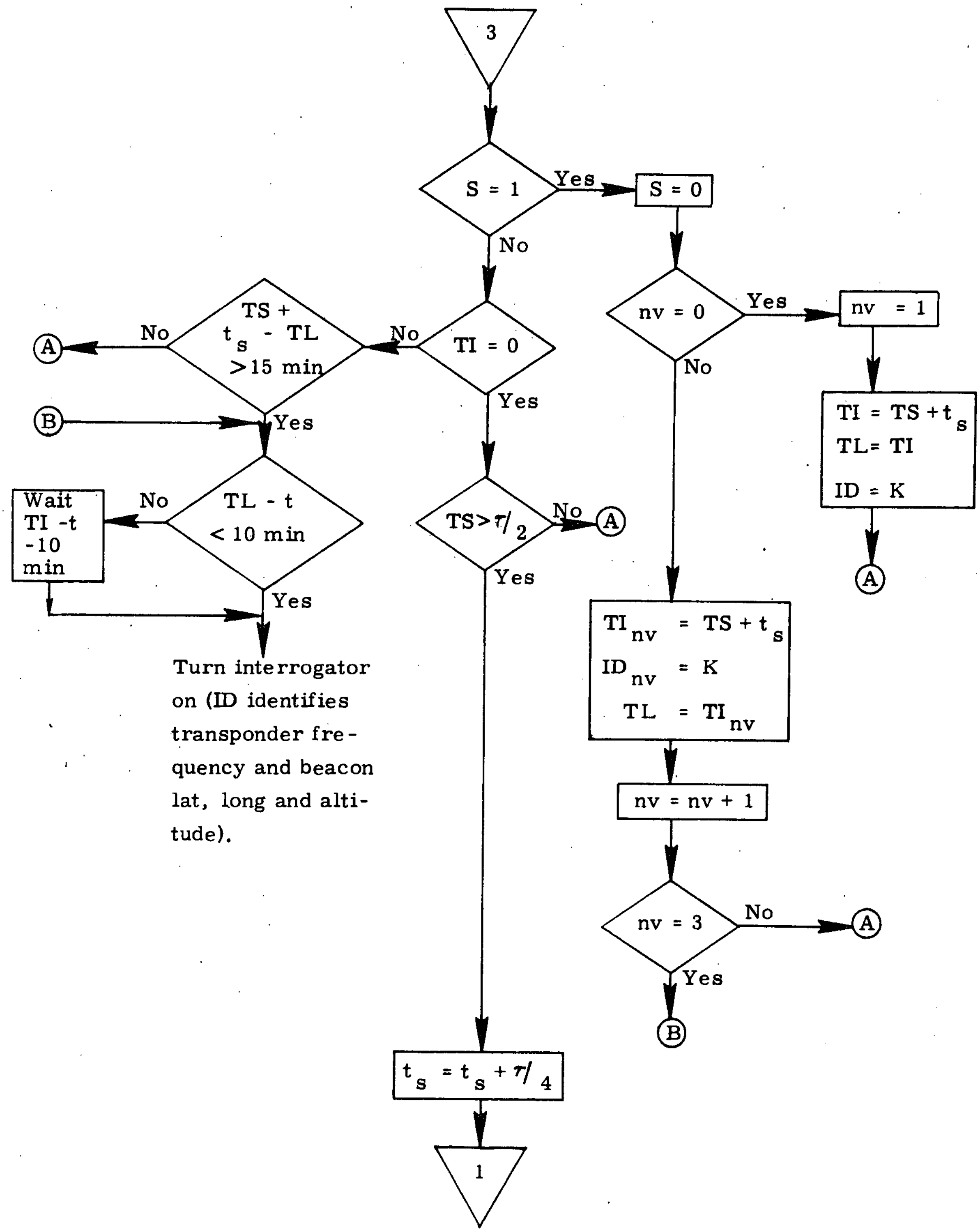
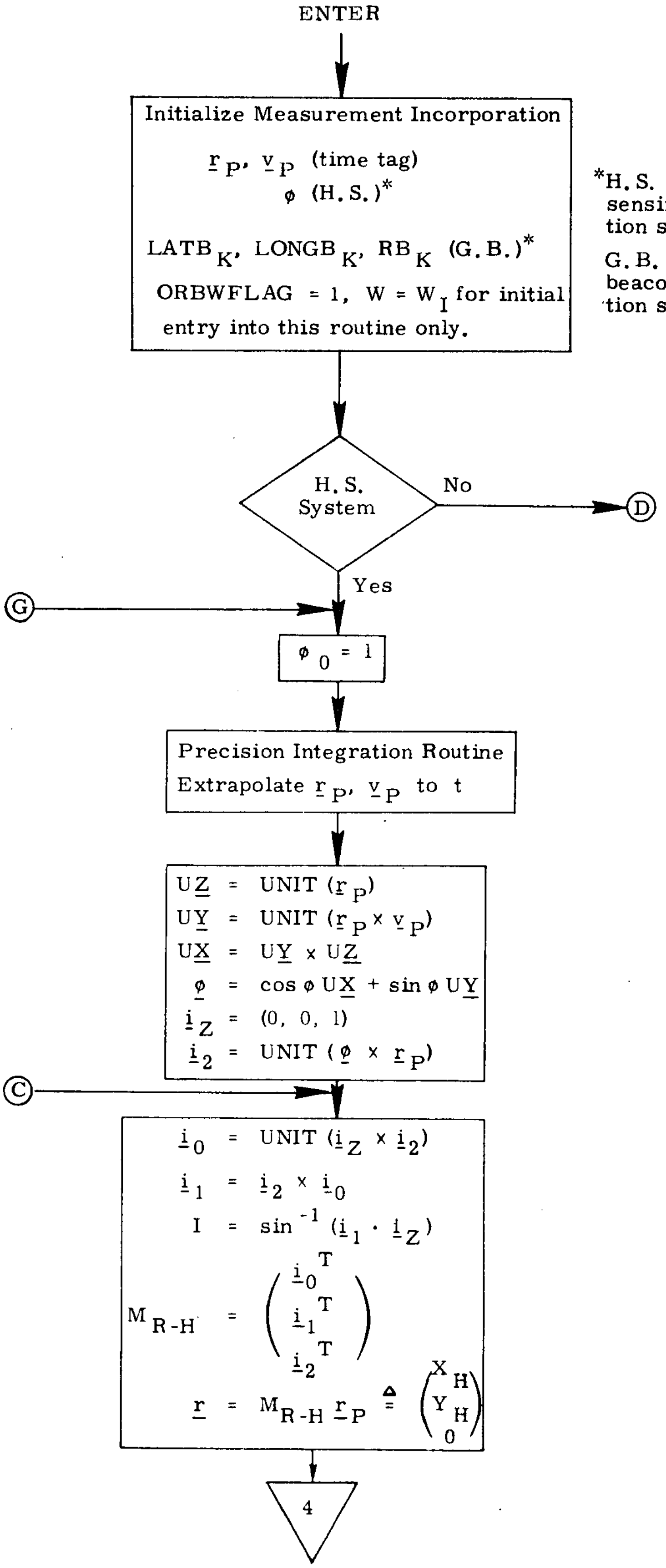


Figure 5c. Detailed Flow Diagram Ground Beacon Prediction Routine


ENTER
Initialize Measurement Incorporation
$\underline{r}_P$, $\underline{v}_P$ (time tag)
ϕ (H.S.)*
$LATB_K$, $LONGB_K$, RB_K (G.B.)*
ORBWFLAG = 1, W = W_I for initial entry into this routine only.
H.S. System
No
Yes
D
G
$\phi_0 = 1$
Precision Integration Routine
Extrapolate $\underline{r}_P$, $\underline{v}_P$ to t
$U\underline{Z} = UNIT(\underline{r}_P)$
$U\underline{Y} = UNIT(\underline{r}_P \times \underline{v}_P)$
$U\underline{X} = U\underline{Y} \times U\underline{Z}$
$\underline{\phi} = \cos\phi\, U\underline{X} + \sin\phi\, U\underline{Y}$
$\underline{i}_Z = (0,\ 0,\ 1)$
$\underline{i}_2 = UNIT(\underline{\phi} \times \underline{r}_P)$
C
$\underline{i}_0 = UNIT(\underline{i}_Z \times \underline{i}_2)$
$\underline{i}_1 = \underline{i}_2 \times \underline{i}_0$
$I = \sin^{-1}(\underline{i}_1 \cdot \underline{i}_Z)$
$M_{R-H} = \begin{pmatrix} \underline{i}_0^T \\ \underline{i}_1^T \\ \underline{i}_2^T \end{pmatrix}$
$\underline{r} = M_{R-H}\, \underline{r}_P \triangleq \begin{pmatrix} X_H \\ Y_H \\ 0 \end{pmatrix}$
4

*H.S. = horizon sensing navigation system

G.B. = ground beacon navigation system

Figure 6a. Detailed Flow Diagram Measurement Incorporation Routine

9.6.1.3 Orbit Navigation Using Navigation Sensors (continued)

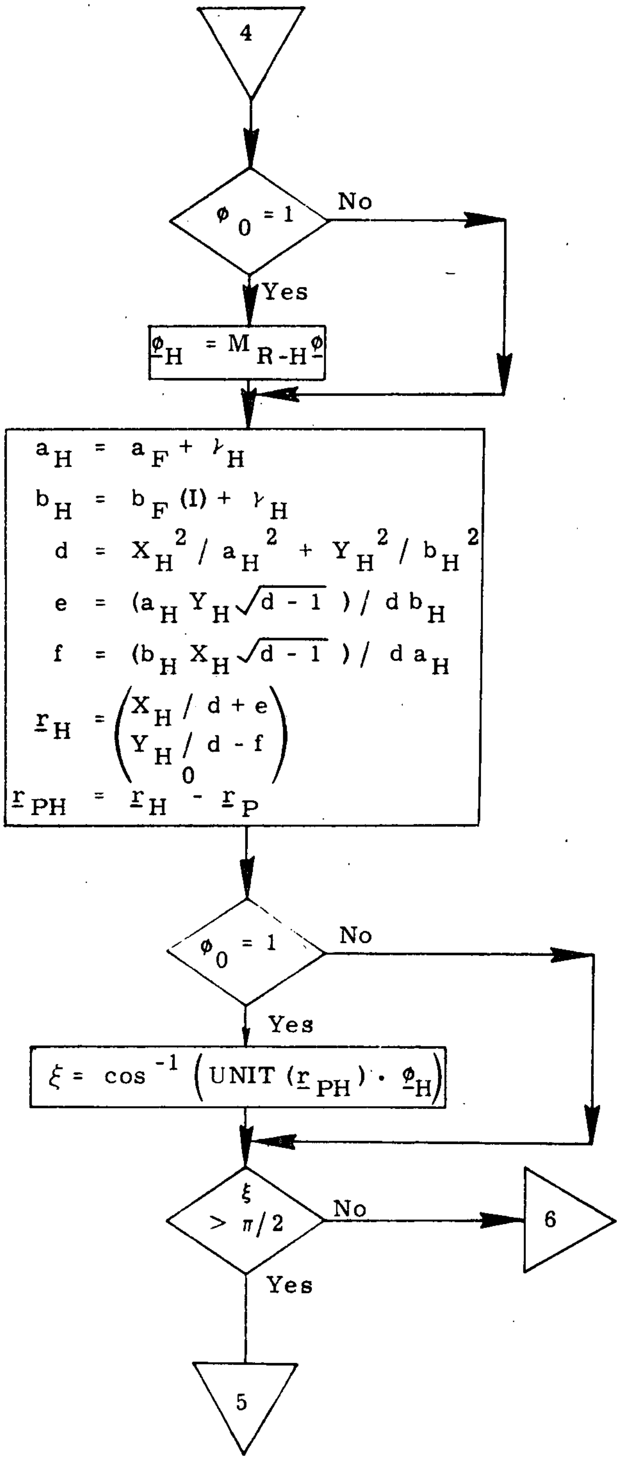

4
$\phi_0 = 1$
No
Yes
$\underline{\phi}_H = M_{R-H}\underline{\phi}$
$a_H = a_F + \nu_H$
$b_H = b_F (I) + \nu_H$
$d = X_H^2 / a_H^2 + Y_H^2 / b_H^2$
$e = (a_H Y_H \sqrt{d-1}) / d b_H$
$f = (b_H X_H \sqrt{d-1}) / d a_H$
$\underline{r}_H = \begin{pmatrix} X_H / d + e \\ Y_H / d - f \\ 0 \end{pmatrix}$
$\underline{r}_{PH} = \underline{r}_H - \underline{r}_P$
$\phi_0 = 1$
No
Yes
$\xi = \cos^{-1} \left(\text{UNIT} (\underline{r}_{PH}) \cdot \underline{\phi}_H \right)$
$\xi > \pi/2$
No
6
Yes
5

Figure 6b. Detailed Flow Diagram
Measurement Incorporation Routine

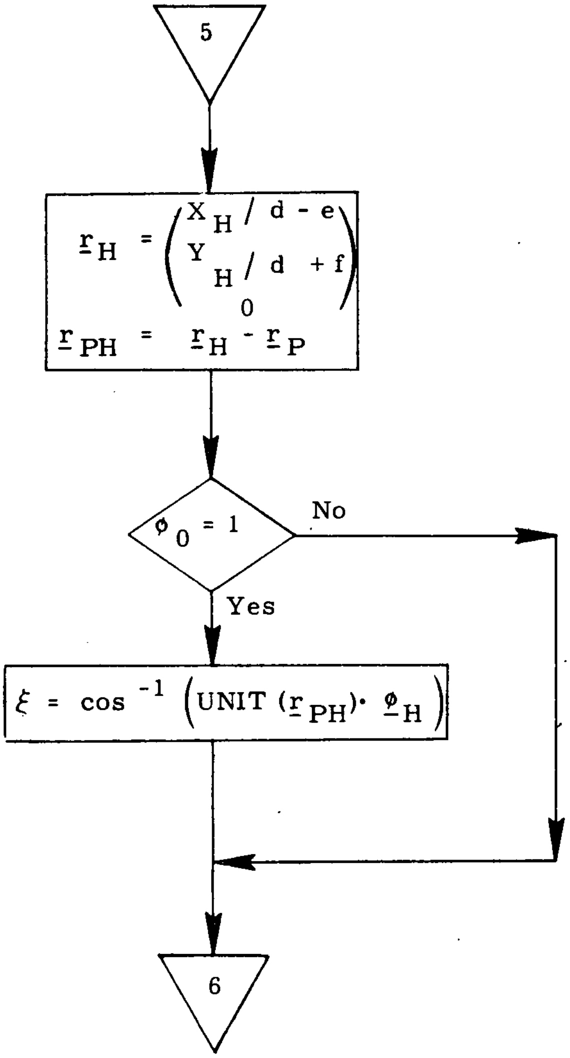

5
$\underline{r}_H = \begin{pmatrix} X_H / d - e \\ Y_H / d + f \\ 0 \end{pmatrix}$
$\underline{r}_{PH} = \underline{r}_H - \underline{r}_P$
$\phi_0 = 1$
No
Yes
$\xi = \cos^{-1}\left(\text{UNIT}(\underline{r}_{PH}) \cdot \underline{\phi}_H\right)$
6

Figure 6c. Detailed Flow Diagram
Measurement Incorporation Routine

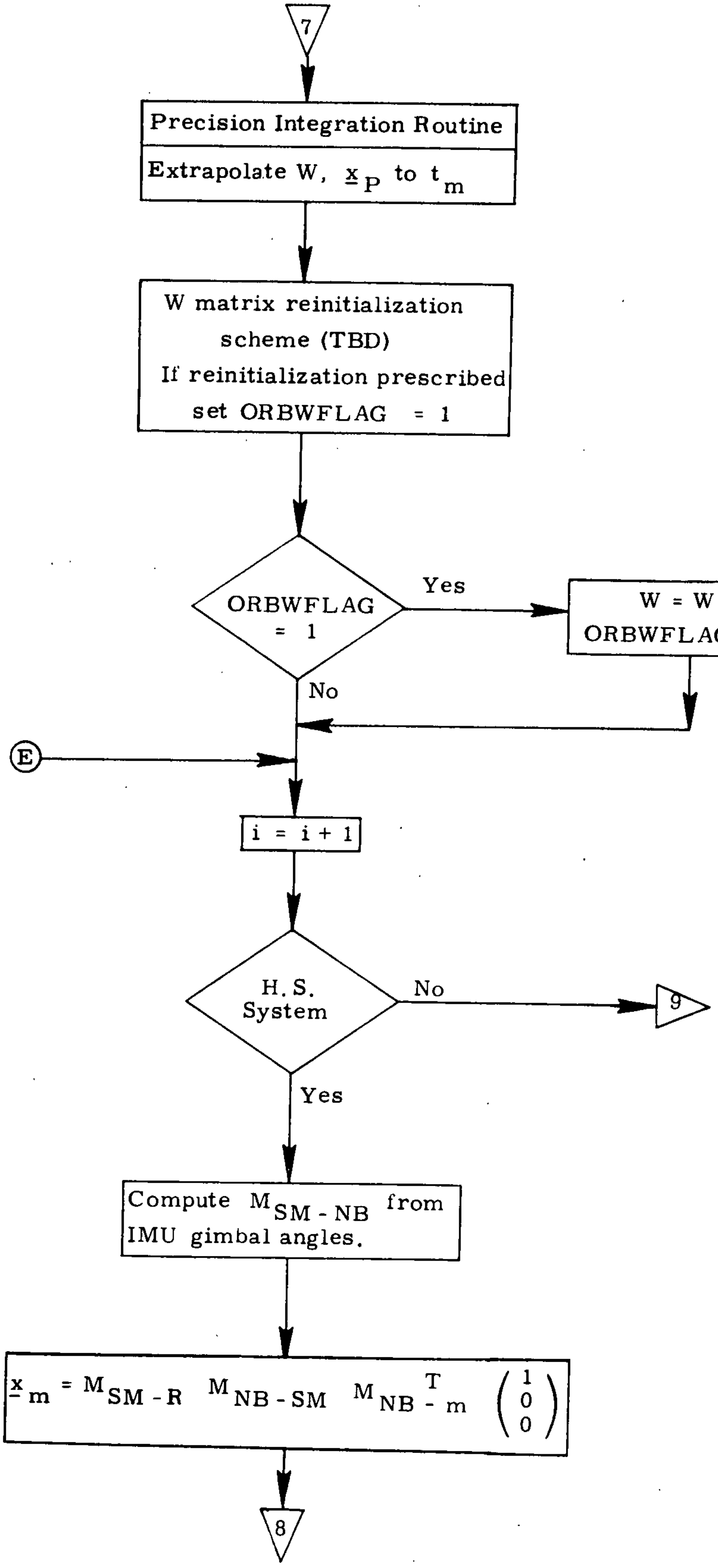

7
Precision Integration Routine
Extrapolate W, $\underline{x}_P$ to t_m
W matrix reinitialization scheme (TBD)
If reinitialization prescribed set ORBWFLAG = 1
ORBWFLAG = 1
Yes
$W = W_F$
ORBWFLAG = 0
No
E
i = i + 1
H. S. System
No
9
Yes
Compute M_{SM-NB} from IMU gimbal angles.
$\underline{x}_m = M_{SM-R} \; M_{NB-SM} \; M_{NB-m}^T \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$
8

Figure 6e. Detailed Flow Diagram
Measurement Incorporation Routine

9.6.1.3 Orbit Navigation Using Navigation Sensors (continued)

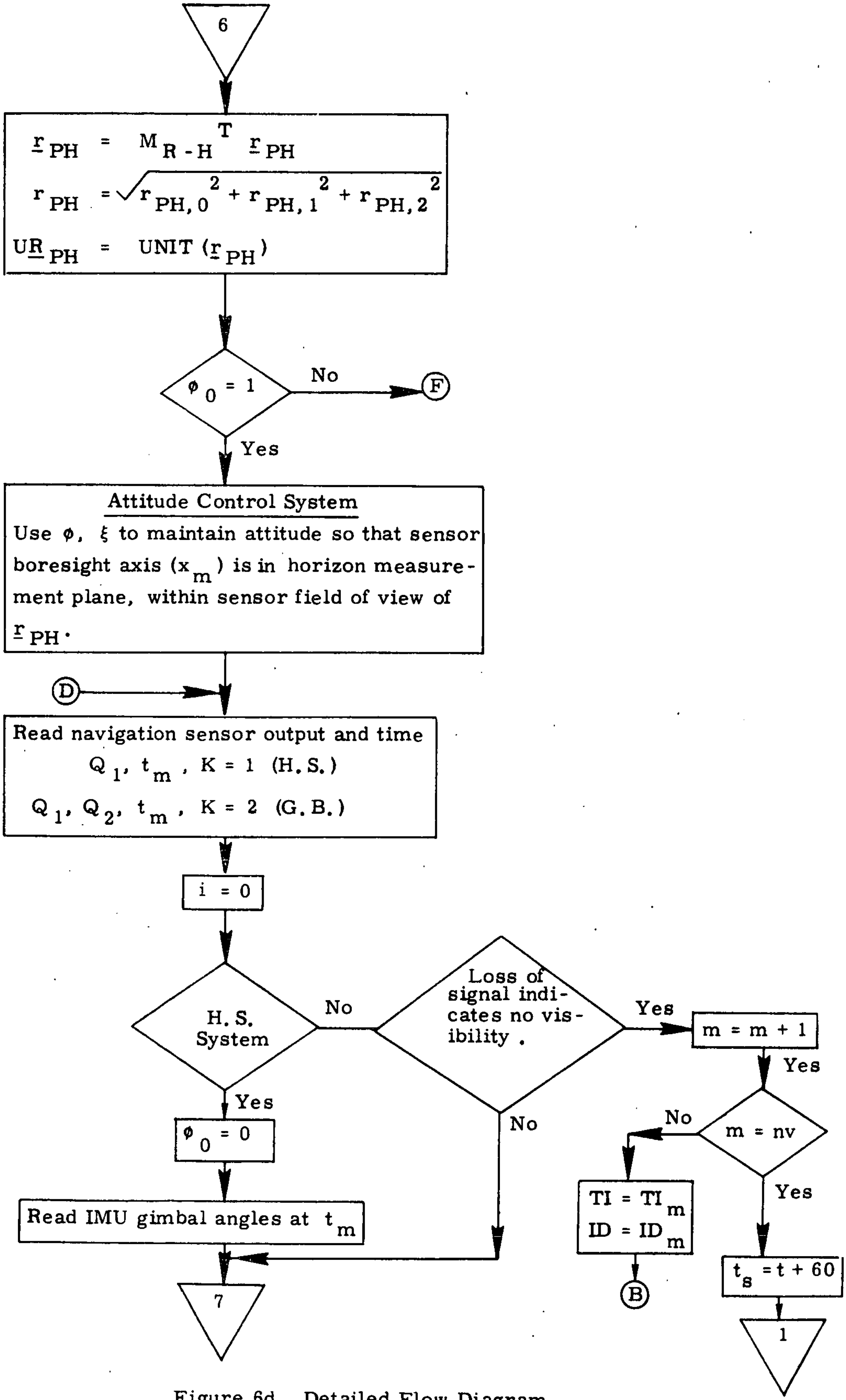

6
$\underline{r}_{PH} = M_{R-H}^{T} \underline{r}_{PH}$
$r_{PH} = \sqrt{r_{PH,0}^2 + r_{PH,1}^2 + r_{PH,2}^2}$
$U\underline{R}_{PH} = \text{UNIT} (\underline{r}_{PH})$
$\phi_0 = 1$
No
F
Yes
Attitude Control System
Use ϕ, ξ to maintain attitude so that sensor boresight axis (x_m) is in horizon measurement plane, within sensor field of view of $\underline{r}_{PH}$.
D
Read navigation sensor output and time
Q_1, t_m, K = 1 (H.S.)
Q_1, Q_2, t_m, K = 2 (G.B.)
i = 0
H.S. System
No
Yes
$\phi_0 = 0$
Read IMU gimbal angles at t_m
7
Loss of signal indicates no visibility.
Yes
No
m = m + 1
Yes
m = nv
No
Yes
$TI = TI_m$
$ID = ID_m$
B
$t_s = t + 60$
1

Figure 6d. Detailed Flow Diagram
Measurement Incorporation Routine

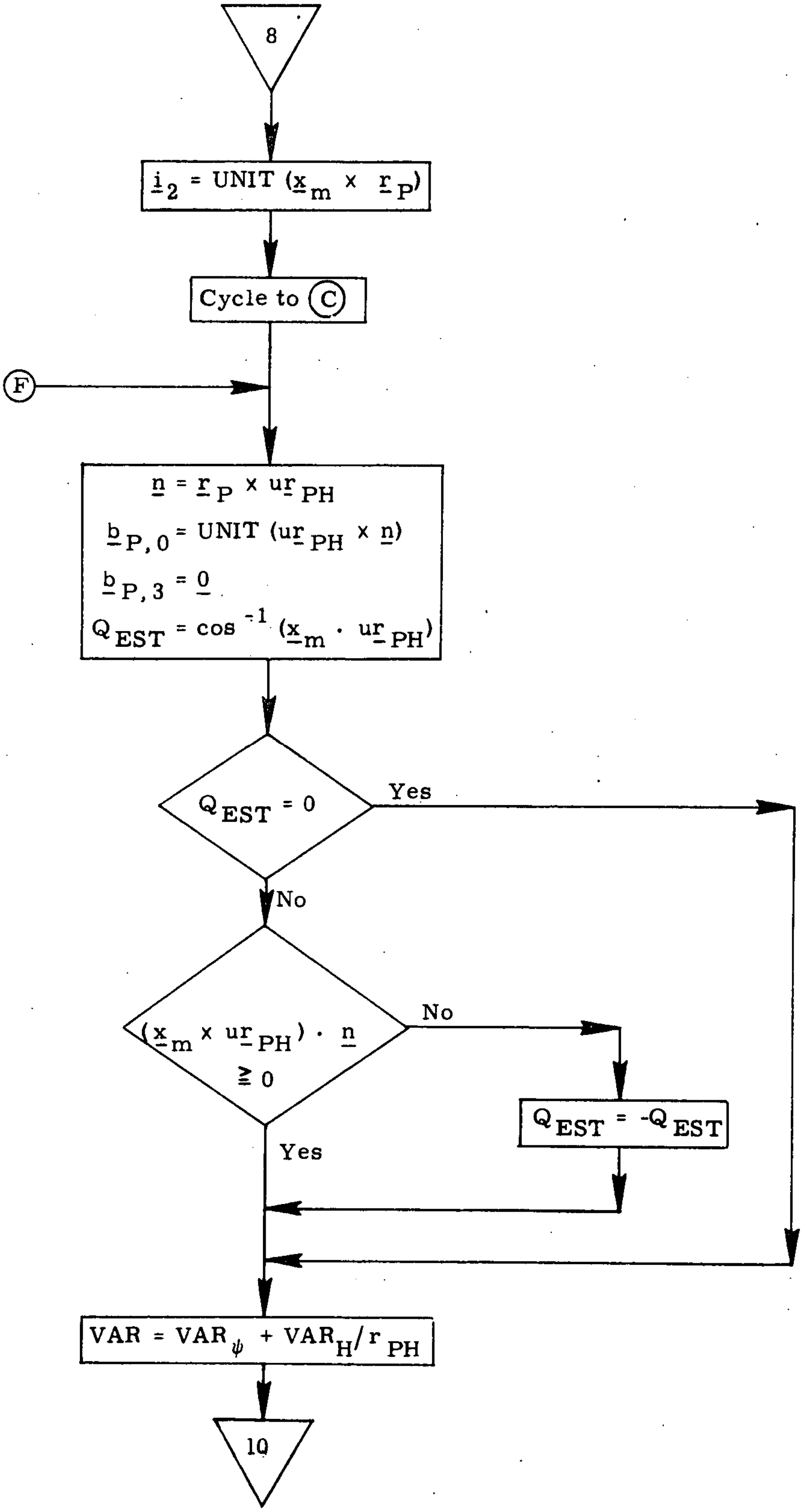

8
$\underline{i}_2$ = UNIT ($\underline{x}_m$ × $\underline{r}_P$)
Cycle to C
F
$\underline{n} = \underline{r}_P \times \underline{ur}_{PH}$
$\underline{b}_{P,0}$ = UNIT ($\underline{ur}_{PH} \times \underline{n}$)
$\underline{b}_{P,3} = \underline{0}$
$Q_{EST} = \cos^{-1} (\underline{x}_m \cdot \underline{ur}_{PH})$
$Q_{EST} = 0$
Yes
No
$(\underline{x}_m \times \underline{ur}_{PH}) \cdot \underline{n} \geqq 0$
No
$Q_{EST} = -Q_{EST}$
Yes
VAR = VAR$_\psi$ + VAR$_H$/r$_{PH}$
10

Figure 6f. Detailed Flow Diagram
Measurement Incorporation Routine

9.6.1.3 Orbit Navigation Using Navigation Sensors (continued)

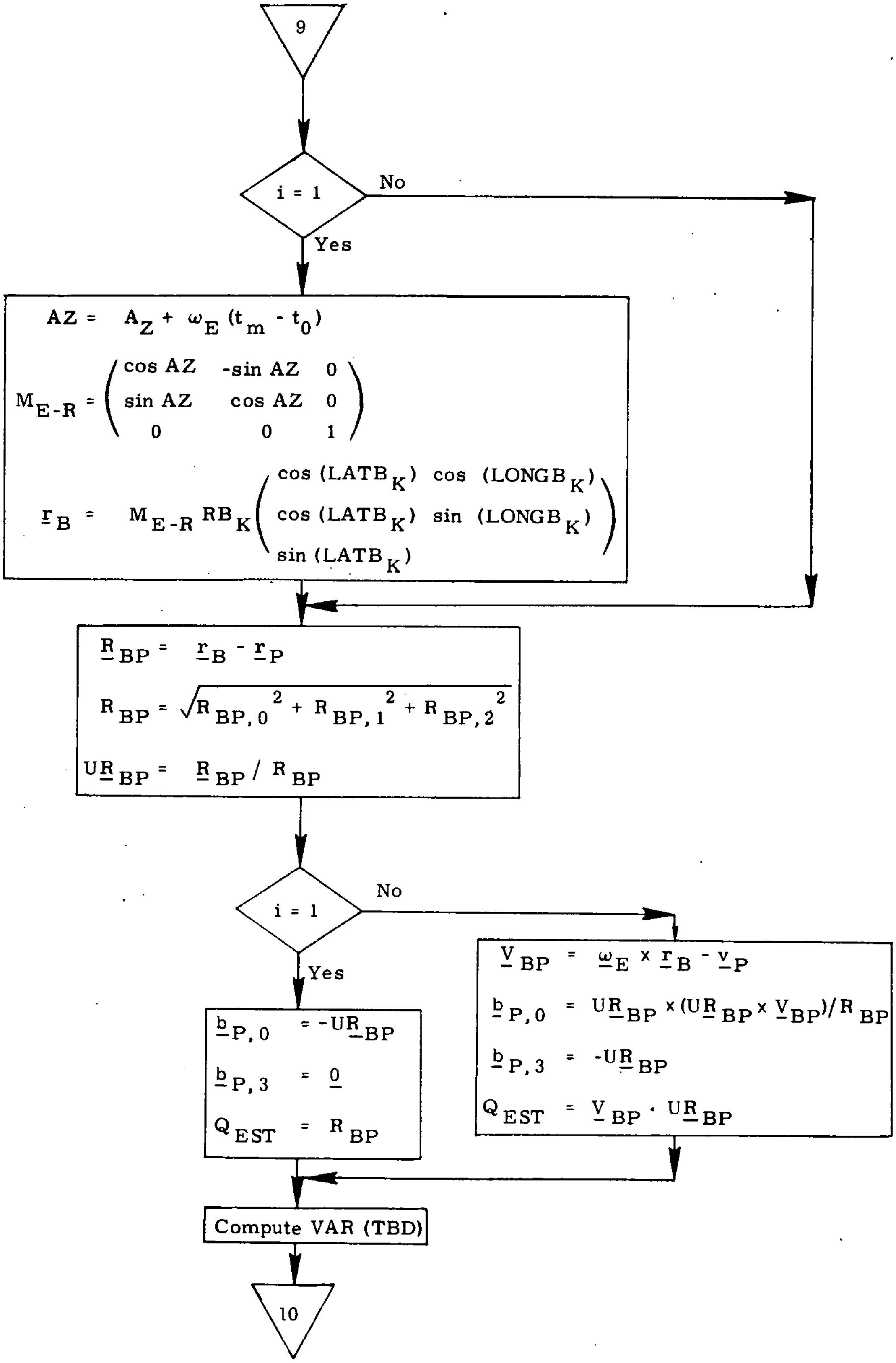

9
i = 1
No
Yes
$AZ = A_Z + \omega_E (t_m - t_0)$
$M_{E-R} = \begin{pmatrix} \cos AZ & -\sin AZ & 0 \\ \sin AZ & \cos AZ & 0 \\ 0 & 0 & 1 \end{pmatrix}$
$\underline{r}_B = M_{E-R}\, RB_K \begin{pmatrix} \cos(LATB_K) \cos(LONGB_K) \\ \cos(LATB_K) \sin(LONGB_K) \\ \sin(LATB_K) \end{pmatrix}$
$\underline{R}_{BP} = \underline{r}_B - \underline{r}_P$
$R_{BP} = \sqrt{R_{BP,0}^2 + R_{BP,1}^2 + R_{BP,2}^2}$
$U\underline{R}_{BP} = \underline{R}_{BP} / R_{BP}$
i = 1
No
Yes
$\underline{b}_{P,0} = -U\underline{R}_{BP}$
$\underline{b}_{P,3} = \underline{0}$
$Q_{EST} = R_{BP}$
$\underline{V}_{BP} = \underline{\omega}_E \times \underline{r}_B - \underline{v}_P$
$\underline{b}_{P,0} = U\underline{R}_{BP} \times (U\underline{R}_{BP} \times \underline{V}_{BP}) / R_{BP}$
$\underline{b}_{P,3} = -U\underline{R}_{BP}$
$Q_{EST} = \underline{V}_{BP} \cdot U\underline{R}_{BP}$
Compute VAR (TBD)
10

Figure 6g. Detailed Flow Diagram
Measurement Incorporation Routine

9.6.1.3 Orbit Navigation Using Navigation Sensors (continued)

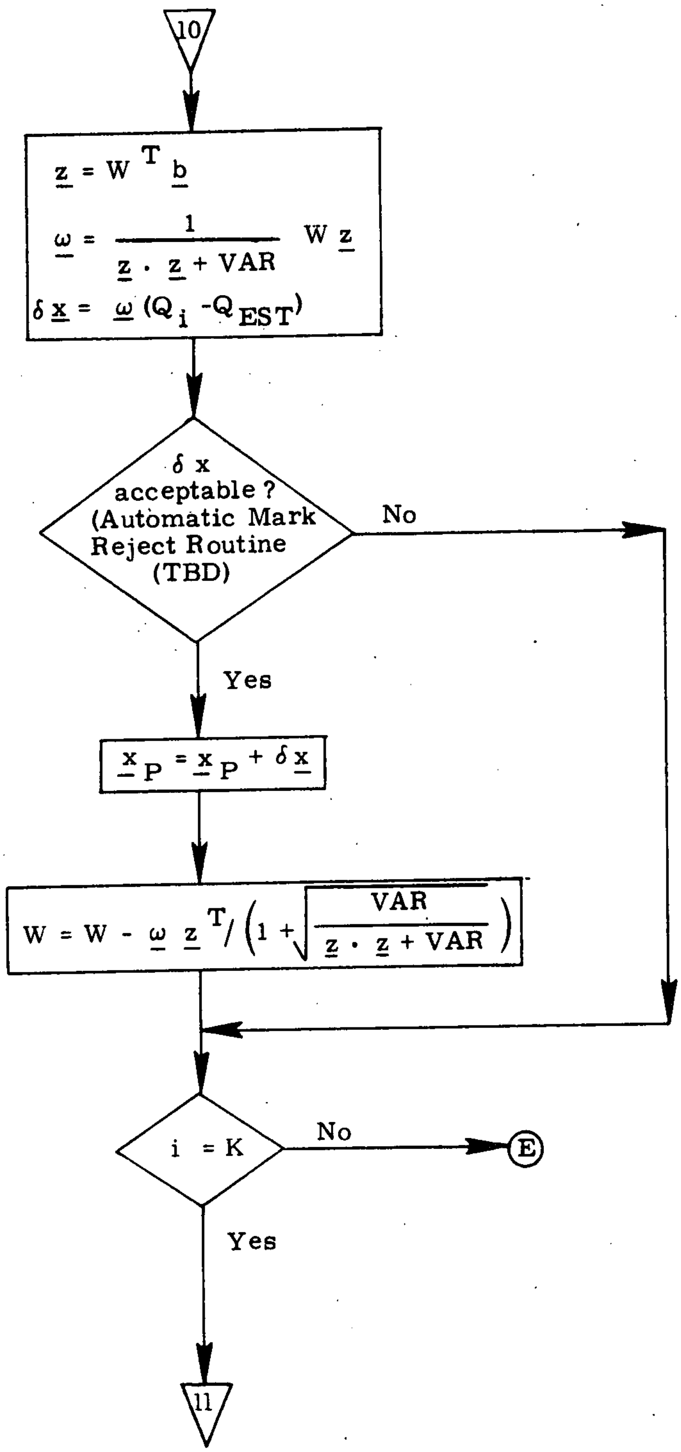

10
$\underline{z} = W^T \underline{b}$
$\underline{\omega} = \frac{1}{\underline{z} \cdot \underline{z} + VAR} W \underline{z}$
$\delta \underline{x} = \underline{\omega} (Q_i - Q_{EST})$
δ x acceptable ? (Automatic Mark Reject Routine (TBD)
No
Yes
$\underline{x}_P = \underline{x}_P + \delta \underline{x}$
$W = W - \underline{\omega}\, \underline{z}^T / \left(1 + \sqrt{\frac{VAR}{\underline{z} \cdot \underline{z} + VAR}}\right)$
i = K
No
E
Yes
11

Figure 6h. Detailed Flow Diagram
Measurement Incorporation Routine

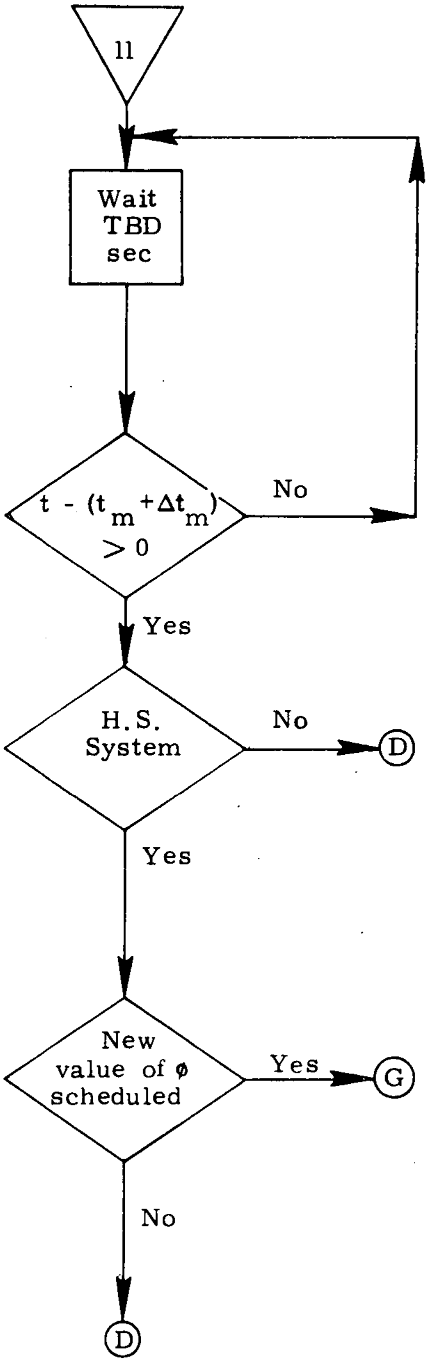


Figure 6i. Detailed Flow Diagram
Measurement Incorporation Routine

4. SUPPLEMENTARY INFORMATION

The equations presented in this report are the results to date of studies performed under a G&C shuttle task to develop G&N equations for automatic orbit navigation. For the system to be fully automated, an automatic mark reject routine remains to be formulated. In addition, a filter weighting matrix reinitialization schedule must be prescribed. For a ground beacon navigation system, preliminary analyses indicate the W matrix should be reinitialized when it has been "used" for more than an orbit. (Ref. 3). Also, if widely spaced beacons are used, this reinitialization should be performed approximately 3 navigation marks into a beacon pass. For closely spaced beacon pairs the reinitialization may be performed prior to the first mark on a beacon pass.

The prescribed horizon directions (ϕ) have not been finalized. Preliminary analyses (Ref. 5) indicate a satisfactory schedule might consist of a series of forward sightings and a series of backward sightings in the orbital plane, and a few sightings to each side of the orbital plane. The final schedule must take into account sunlight constraints assuming the horizon sensor utilizes ultra-violet radiation. A sunlit horizon prediction scheme will then also be required to be incorporated in the navigation equations.

The horizon sensor assumed for the equations presented in this document utilizes a single degree of freedom scan to determine the angle from its boresight axis to the horizon sighted. This requires the attitude control system to maintain the sensor "scan axis" to be normal to the estimated vehicle position vector with the sensor boresight axis at a prescribed azimuth angle from the forward direction and within the sensor field of view from the horizon. The final equations will of course be a function of the actual sensor operating characteristics and its location on the spacecraft.

9.6.1.3 Orbit Navigation Using Navigation Sensors (continued)

References

1. Muller, E.S., Phillips, R.E., "Relative State Updating", Space Shuttle GN&C Equation Document No. 6-71, February 1971, MIT C.S. Draper Lab.

2. Robertson, W.M., "Precision State and Filter Weighting Matrix Extrapolation", Space Shuttle GN&C Equation Document, No. 4-71, February 1971, MIT C.S. Draper Lab.

3. NASA MSC Memorandum #EG1-71-17, "Guidance and Control Division Space Shuttle Task Review", April 15, 1971.

4. Battin, R.H., Levine, G.M., "Application of Kalman Filtering Techniques to the Apollo Program", Report E-2401, April 1969, MIT.

5. Muller, E.S., Kachmar P., Philliou, P., Report on "Orbit Navigation Studies for SSV", 23A STS Memo #24-70, June 6, 1970, MIT C.S. Draper Lab.

9.6.2 Sensor Calibration and Alignment

9.6.2.1 IRU Alignment

SPACE SHUTTLE

GN&C SOFTWARE EQUATION SUBMITTAL

Software Equation Section: Sensor Calibration and Alignment

Submittal No. 33

Function: IRU Fine Alignment

Module No. OS-4 Function No. 4 (MSC-03690)

Submitted by: C. Manry Co. EG5 (MSC-04910)

Date: 8/24/71

NASA Contract: C. F. Lively Organization EG2

Approved by Panel III K. J. Cox Date 8/24/71

Summary Description: Equations are developed for use of two successive star sightings to perform fine alignment of the IRU. Included are star-table listing and a search routine for each of the three star tracker fields of view.

Shuttle Configuration: (Vehicle, Aero Data, Sensor, Et Cetera) Three Multi-Mode Optical Sensors are assumed, each with 17° x 17° field of view, each viewing a different sector of the upper hemisphere.

Comments:

(Design Status)

(Verification Status)

Panel Comments:

FINE ALINEMENT OF THE SPACE SHUTTLE INERTIAL REFERENCE UNIT BY THE MULTI-MODE OPTICAL SENSOR

Summary

After a brief description of the Multi-Mode Optical Sensor and its operational characteristics, a procedure is developed for fine alinement of the Space Shuttle's Inertial Reference Unit (IRU) by means of two successive star sightings. Then, the basic equations are derived for use in computer simulations of the sensor's operation in a realistic environment. Sample results from a computer simulation of these equations have been included in an appendix.

Introduction

Description of the Multi-Mode Optical Sensor - For a high-inertia vehicle like the Shuttle, where reliability requirements are extremely high, the apparent best choice for an optical alinement sensor is a wide-field, strapped-down, electronically-gimballed star tracker. The potential flexibility of this type of sensor for such additional applications as sunlit target tracking for rendezvous, and ultra-violet horizon tracking for orbital navigation, make this a very attractive choice for the Shuttle. The basic performance parameters for the sensor to be described have been proven in various applications aboard unmanned space vehicles and rocket-borne experiment payloads.

An engineering model of the Multi-Mode Optical Sensor is currently being procured for evaluation. The design requirements for this sensor will provide a capability to:

a. Acquire the brightest star (brighter than +3.0 visual magnitude) within the sensor's 17°-by-17° square field of view.

b. Acquire any star brighter than +3.0 visual magnitude within a square search field 2° on a side, centered about a point which can be computer-directed.

c. Track a star (or sunlit rendezvous target) within an accuracy of one minute of arc (one sigma) relative to the bore-sight axis.

9.6.2.1 IRU Alignment (continued)

d. Under computer control, execute a radiometric, earth-horizon profile scan to determine the altitude of the selected horizon point in vehicle-centered inertial coordinates for orbital navigation.

The detector in the sensor being procured is to be an ITT Image Dissector Type 4012 tube, with an S-20 photocathode. An objective aperture of about 2 inches at an f-number of approximately unity will provide the required field dimensions. The instantaneous field of view in such a sensor is set by the mechanical size of the aperture hole inside the image dissector. For the present application, it is anticipated that a one-quarter degree subtense will be selected. In the acquisition modes, this instantaneous field will be caused by magnetic deflection coils to sweep the search field of view in a sequentially-stepped, raster-type scan. At each point the average energy in the instantaneous field of view will be sampled for a dwell time of 230 microseconds. Reference 1 contains more details of the functional operation of the detector and of the sensor.

After acquisition of the star, the tracker switches to a track mode for higher accuracy. In this mode, the instantaneous field of view is rapidly swept over the target in a manner which will provide error signals to the deflection circuits to keep the target centered in the tracking field. The accuracy of this target centering is expected to be about 30 seconds of arc (one sigma), and is dependent mainly upon the target's signal-to-noise ratio.

At any time after tracking begins, the star's location relative to the sensor's field of view center can be read out by external command. In addition to the tracking accuracy mentioned above, there will be a non-linearity error component in the angle readout, which is due to scale factor non-linearity in the deflection circuitry. This error is expected to be between one-half and one minute of arc, but may be partially compensated by additional electronic circuitry or by the onboard computer.

Mechanically, the optical sensor is expected to weigh approximately 15 pounds, including electronics and optics, and will operate on about 20 watts electrical input power at 28 volts D.C.

Summary of Assumptions Pertaining to Fine Alinement

a. Computer Control of the Alinement Procedure - Complete control of the IRU alinement procedure is assumed to be contained within the Flight Control Computer (FCC). A monitor display will be provided to the crew at critical decision points. As presently envisioned, these decisions should include only those affecting or requiring attitude maneuvering fuel, or the actual process of torquing the IRU gimbals.

The decision to perform an IRU fine alinement procedure may be made by the FCC, although provision will be made for manual request of this action. Thus, the crew may obtain an IRU fine alinement in preparation for an unscheduled flight activity, for which the FCC has no information. But, in most cases, the FCC will request a new alinement because the time lapse since the previous alinement has become too long for acceptable attitude accuracy. If the FCC is informed in advance of mission events upcoming, then it can decide logically what level of attitude accuracy will be acceptable, and execute the alinement at an optimum time. This advance notice may well allow "stars of opportunity" to be used without expenditure of attitude maneuvering fuel. The procedures developed in this report were based upon this assumption primarily. Preliminary results have indicated that this is a reasonable approach.

b. Number of Sensors - For description purposes, it has been assumed that three identical Multi-Mode Optical Sensors will be located on the vehicle. Each will have a separate (non-overlapping) field coverage, and each sensor will have a well-known orientation with respect to the SSV navigation base. It has also been assumed that the FCC will have command control over these sensors sufficient to allow power switching, protective cover removal and return, selection of modes, and control over the deflection circuitry within the sensor.

c. Navigation Star Catalog - A catalog of star vectors will be available to the FCC. (See Appendix A). This catalog will contain all the stars that are brighter than +3.0 visual magnitude (approximately 150), and will include tables of planet positions for the planets having acceptable brightness. These will include Venus, Mars, and Jupiter. Presumably, the locations of the sun, earth and moon are also available with adequate precision because of their perturbation effects on the gravitational potential in the near-earth environment.

Coordinate System Definitions

a. Body Coordinate System - The SSV body coordinate system is illustrated in Figure 1. It consists of an orthogonal, three-axis system, with the $+X_B$ axis directed out the forward portion of the fuselage, and with the $+Y_B$ axis pointed out the right wing. The $+Z_B$ axis is directed out the bottom of the vehicle to complete a right-handed system. In this system, positive roll (defined as rotation about the X axis) will bring the +Y axis toward the +Z axis, positive pitch (rotation about the Y axis) will carry the +Z axis toward the +X axis, and positive yaw (rotation about the Z axis) will bring the +X axis closer to the +Y axis. The origin of this coordinate system is located at the nominal center-of-gravity of the vehicle.

Since the order of performing attitude maneuvers is critical to this description, some convention has to be adopted in order to develop meaningful equations. For the purposes of this report, it has been assumed that maneuvers will be carried out in the order of first roll, then pitch, and finally yaw.

b. Navigation Base Coordinate System - The navigation base coordinate system consists of an orthogonal set of axes which are defined to be parallel to the SSV body coordinate system described above, but with the origin offset from the nominal center-of-gravity by some undetermined distance and direction. The coordinate axes are identified as X_{NB}, Y_{NB}, and Z_{NB}.

c. Sensor Coordinate System - The locations of each sensor in the navigation base coordinate system will be expressed in the form of coordinate translations parallel, respectively, to the X_{NB}, Y_{NB}, and Z_{NB} coordinate axes. Any location within the field of view of a particular sensor will be expressed in terms of an azimuth angle α_i and an elevation angle β_i. The angle γ_i relates the rotation of the α_i, β_i system about the sensor field of view center to the navigation base coordinate system. The angle γ_i will be referred to as the tilt angle.

There is a three-axis rotational transformation between the navigation base coordinate system and the sensor field of view coordinate system. The transformation will be shown to be a function of α_i, β_i, γ_i and the three angles which relate the direction of the center of the field of view to the navigation base axes.

Description of the Procedure

Decision to Aline - The logical decision to perform an alinement may be based upon the following conditions:

a. The elapsed time since the previous alinement, when multiplied by the uncompensated gyro drift rate, indicates that the estimated attitude error will exceed the tolerance required for the next flight phase.

b. Selection of fine alinement may be requested by other FCC programs having to do with inflight calibration of inertial or optical sensors.

c. Manual request for alinement may be made by crew option in order to prepare for a special maneuver or other flight events.

Star Selection Process - A method of star selection has been developed specifically for use with a wide-field sensor in a fixed installation on a high-inertia vehicle. The primary intent is to minimize the fuel expenditure by making maximum use of the computational capability available in the FCC.

Upon receipt of an alinement request, the FCC first computes the current (time = t_o) direction cosines in inertial coordinates for the three individual sensor centerlines. Next, the FCC will carry out an update of the vehicle's inertial state vector estimate to the time t_o. Based upon this vehicle position and upon the ephemerides of the sun and the earth, the FCC then excludes from further consideration any sensor which is looking within 30 degrees of the sun, or which is completely blocked off by the earth. Only the usable sensors are included in the following calculations.

Next, the inertial direction of each of the usable sensors in turn is compared to the star vectors in the catalog. If the star does fall within the field of view of a usable sensor, it is subjected to the following tests:

a. Is it occulted by the earth? (Note: The sensor-to-earth test above is designed to allow the use of a sensor which has part of its view blocked by the earth. As a result it is also necessary to check the particular star to be used.)

b. Is the star too close to the moon? As with the sun, it is anticipated that the sensor will not be able to track a star too close to the moon. A tolerance of about 5 degrees is expected.

After all of the stars have been tested with each of the usable sensors, the FCC must now decide if an alinement is possible at time t_o, and if not, determine a strategy for accomplishing the alinement at a later time. If there are two or more stars available at t_o, the alinement can be immediately carried out if the particular pair of stars available pass a separation angle criteria. The angle between the two stars must be large enough to provide a satisfactory orientation reference for the alinement. A preliminary specification for this angle has been estimated at $35 < \lambda_{ss} < 145$ degrees.

If there are not two available stars which can pass this test, then the FCC attempts to plan a delayed sighting sequence which could be carried out if the computer is allowed to inhibit the attitude control thrusters. This planning function is accomplished by having the FCC integrate the body attitude rates in one minute steps for a maximum of ten minutes, with the new sensor pointing directions being used in the star selection procedure. The process is repeated until two stars have been found which meet the separation angle criteria, or until after the tenth iteration.

Upon completion of the star selection process, the crew will be informed by an appropriate display if the FCC has had to assume thruster inhibit action in order to perform the alinement. The crew may then permit this action, or if time permits, elect to postpone the alinement until a later time at which the sensor directions may be more favorable. If time is very limited, the crew may decide to perform an attitude maneuver to improve the sensor orientations with respect to the stars. There is no provision in this alinement routine to assist the crew in selecting a preferred attitude.

Auto-Optics Command - When this portion of the procedure is begun, there will exist at least one available catalog star within the field of view of one sensor. The inertial coordinates of the star are known in the star catalog, and the transformation matrix relating the inertial coordinate system to the navigation base coordinate system is known with fair accuracy, at least within one degree (three sigma). Then, with the relatively well-known transformation from the navigation base to sensor coordinates, it becomes possible to compute the approximate (two-axis) sensor coordinates to the desired star. The FCC then converts these two sensor coordinate angles into digital format for transmittal to the appropriate sensor.

At the sensor, the two digital quantities will be used directly to initialize the reacquisition mode center position. The sensor will then carry out a systematic raster scan of an angular region measuring two degrees on a side, centered about the auto-optics command position. Figure 2 illustrates the field of view layout in the reacquisition mode.

If no star is acquired, the sensor notifies the FCC and automatically continues to try for acquisition by searching the entire field of view for the brightest star. In this latter instance, the FCC should inform the crew that the IRU performance has possibly been degraded, as evidenced by a higher than normal gyro drift rate. Final proof of this, however, will require completion of the IRU alinement in order to rule out optical sensor malfunctions.

Star Acquisition - After the sensor has acquired a star, the sensor switches to a tracking mode. In the tracking mode, the two star angles may be read out from the sensor at any time by the FCC. There is at present no intention to use the signal level to determine the star magnitude of the acquired star.

"Mark" Data Processing - At any time after tracking has begun, the computer may issue a digital "Mark" command to the sensor. Upon receipt of this command, the sensor freezes the tracking circuitry position voltages, and converts them to digital format for transmittal to the computer.

At the same instant of the "Mark" command, the IRU gimbal angles are also made available to the computer in digital form. With the two optics angles, and the corresponding sensor-to-navigation base transformation matrix, and the three gimbal angles, the FCC can compute the measured inertial line of sight vector to star #1.

Second Star Sighting - If the second star selected is already within the field of view of a sensor, the computer carries out the second star sighting immediately. However, if a delay is anticipated, the computer goes to a stand-by posture until the second star is expected to be available. When the sighting becomes possible, the computer executes the auto-optics command, the "Mark" command, and the "Mark" data processing in exactly the same way as for the first star.

Preliminary Star Identification Check - At this point in the procedure there is one easy test that the computer can make upon the two star identifications. This is the star angle difference check that was used in the Apollo computer. The computer is used to calculate the angle between the inertial line of sight vectors to the two stars that have been measured. Then, a similar calculation is made with the unit vectors stored in the star catalog. The difference between these two angles is a fair measure of the overall sighting accuracy, and can be used to rule out almost all of the possible mis-identifications of stars.

It is impossible to determine from this check alone that the two stars have not been interchanged. Also, with the larger number of catalog stars, and with the reduced accuracy of the automatic star sensor (relative to the Apollo case), the star angle difference check is expected to incorrectly pass a higher percentage of star sightings. If the test is failed, the computer notifies the crew. Otherwise, it proceeds with the calculation of gimbal torquing angles.

Computation of Gimbal Torquing Angles - The computation of gimbal torquing angles can be accomplished using the two star vectors which have been determined. The torquing angles are derived as the three-axis rotation matrix required to bring the measured star vectors into alinement with the catalog star vectors.

Final Checks - The computed gimbal angle changes are displayed to the crew for action. If the angles are consistent with respect to the expected gyro drift rates and the elapsed time since the previous alinement (as determined by the computer), then the crew will probably accept the alinement.

Equation Development

Transformation of Coordinates

a. Inertial-To-Orbital Plane Axis System Transformation [I2O] - The inertial coordinate system used has the $+X_I$ axis directed at the vernal equinox, the $+Z_I$ axis pointed toward the north celestial pole, and the $+Y_I$ axis oriented to complete a right-handed system. The origin of this system is located at the center of mass of the earth. For convenience of operation, it is desirable to perform a coordinate transformation from the inertial system to an intermediate system referred to as the orbital plane system. In this new system, vehicle positions can be described as functions of the orbital inclination, longitude of the ascending node, and the orbital central angle.

The transformation involved consists of an initial rotation about the Z_I axis by an angle Ω to form the primed system. Next, follows a rotation about the X_I' axis by the angle δ to obtain the double-primed system. The third and last rotation is about the Z_I'' axis (pole of the orbit) by the angle ν, which is the orbital central angle. The angle Ω will be recognized as the longitude of the ascending node of the orbit, while the angle δ is the inclination of the orbit with respect to the inertial coordinate system. These angles are shown in Figure 3.

These three rotations make up the transformation matrix [I2O], which is defined below:

$$[I2O] = \begin{bmatrix} \cos\nu & \sin\nu & 0 \\ -\sin\nu & \cos\nu & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\delta & -\sin\delta \\ 0 & \sin\delta & \cos\delta \end{bmatrix} \begin{bmatrix} \cos\Omega & \sin\Omega & 0 \\ -\sin\Omega & \cos\Omega & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

b. Orbital Plane-To-Body Axis System Transformation [O2B] - In the vehicle's body axis system, the order of rotation has been defined as first roll, then pitch, then yaw. The (0°, 0°, 0°) reference for this system is assumed to lie along the vehicle velocity vector, with the wings level and pilot in a "heads up" position. Roll rotation is described as a rotation by the angle Λ about the X_o axis to form the primed system. Next, pitch is accomplished by a rotation through the angle Φ about the Y_o axis to obtain the double-primed system. Finally, yaw is effected by rotation through the angle χ about the Z_o'' to arrive at the desired body axis coordinate system. These rotations are defined by the following matrix expression:

$$[O2B] = \begin{bmatrix} \cos\chi & \sin\chi & 0 \\ -\sin\chi & \cos\chi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\Phi & 0 & -\sin\Phi \\ 0 & 1 & 0 \\ \sin\Phi & 0 & \cos\Phi \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\Lambda & \sin\Lambda \\ 0 & -\sin\Lambda & \cos\Lambda \end{bmatrix}$$

c. Body Axis To Sensor "i" Transformation NB2SI - Since the navigation base coordinate system is defined to be parallel with the body axis coordinate system, the former designation will be used in the remainder of this discussion. There are three sensor orientations for which transformations will be required. The transformation matrix will be developed for the "i-th" sensor, where the index "i" has values of 1, 2, and 3.

The first transformation is an angular rotation about the X_{NB} axis by an angle γ_i to form the X'_{NBI}, Y'_{NBI}, Z'_{NBI} coordinate system. Then follows an angular rotation through the angle δ_i about the Y'_{NBI} axis to obtain the X''_{NBI}, Y''_{NBI}, Z''_{NBI} coordinate system. Similarly, a third rotation is made about the Z''_{NBI} axis by the angle η_i to complete the transformation. Now the resulting "i-th" sensor coordinate system is related to the navigation base system by the equation:

$$[NB2SI] = \begin{bmatrix} \cos\eta_i & \sin\eta_i & 0 \\ -\sin\eta_i & \cos\eta_i & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\delta_i & 0 & -\sin\delta_i \\ 0 & 1 & 0 \\ \sin\delta_i & 0 & \cos\delta_i \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma_i & \sin\gamma_i \\ 0 & -\sin\gamma_i & \cos\gamma_i \end{bmatrix}$$

d. Sensor "i" To Elevation/Azimuth System SI2EA - In the "i-th" sensor coordinate system, the directions to stars and other targets will be measured in terms of azimuth and elevation angles relative to the optical axis and to the sensor vertical and horizontal axes (See Figure 4). Azimuth rotation will be defined as a rotation by an angle α_i about the Y_{SI} axis. Elevation will be defined similarly as a rotation by an angle β_i about the Z_{SI} axis. The X_{SI} axis will be parallel to the "i-th" sensor's optical axis. The sense of the angle α will be considered positive if the angle is measured from the X_{SI} - Y_{SI} plane toward the Z_{SI} axis. Since this is opposite to the convention for right-handed coordinate systems, the transformation has been defined to use the negative value of α. The sense of the angle β will be considered positive if the angle is measured from the X_S-Z_S plane toward the Y_S axis, which is in agreement with the right-handed convention.

The transformation from the sensor system to the elevation/azimuth system is given by:

$$(SI2EA) = \begin{bmatrix} \cos\beta_i & \sin\beta_i & 0 \\ -\sin\beta_i & \cos\beta_i & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\alpha_i & 0 & \sin\alpha_i \\ 0 & 1 & 0 \\ -\sin\alpha_i & 0 & \cos\alpha_i \end{bmatrix}$$

$$\therefore (SI2EA) = \begin{bmatrix} \cos\beta_i\cos\alpha_i & \sin\beta_i & \cos\beta_i\sin\alpha_i \\ -\sin\beta_i\cos\alpha_i & \cos\beta_i & -\sin\beta_i\sin\alpha_i \\ -\sin\alpha_i & 0 & \cos\alpha_i \end{bmatrix}$$

Logic For Decision To Perform IRU Fine Alinement - The FCC first computes the estimated inertial attitude error at the present time by the following equation:

$$\Delta\Omega_t = \dot{\Omega}(t - t_a)$$

where: $\dot{\Omega}$ = Uncompensated gyro drift rate magnitude in degrees/sec
t_a = Time of previous alinement, in seconds, since liftoff
t = Present time, in seconds, since liftoff
$\Delta\Omega_t$ = Estimated inertial attitude error at present time, in degrees (scalar quantity)

Depending upon the magnitude of the present attitude error, the FCC may initiate a fine alinement. The decision is made according to the logic below:

a. Prior to orbital navigation and/or orbital maneuvers

IF $\Delta\Omega_t \geq \Delta\Omega_{ON}$, initiate fine alinement

IF $\Delta\Omega_t + \dot{\Omega}(t_{FP} - t_a) \geq \Delta\Omega_{ON}$, initiate fine alinement

where: $\Delta\Omega_{ON}$ = alinement tolerance for orbital navigation and maneuvers
t_{FP} = time of next flight plan activity requiring IRU alinement

b. Re-entry Preparation

IF $\Delta\Omega_t \geq \Delta\Omega_{RE}$, initiate fine alinement

IF $\Delta\Omega_t + \dot{\Omega}(t_{RE} - t_a) \geq \Delta\Omega_{RE}$, initiate fine alinement

where: $\Delta\Omega_{RE}$ = alinement tolerance for re-entry
t_{RE} = time of atmospheric re-entry, in seconds

Logic For Star Selection Process - The center lines of the three sensors are first converted into inertial coordinates by the transformations below:

$$\vec{S}(I,t_o) = [I2O]^T[O2B]^T[NB2SI]^T \begin{bmatrix}1\\0\\0\end{bmatrix}$$

where: $\begin{bmatrix}1\\0\\0\end{bmatrix}$ = a unit vector along the X-axis of the "i-th" sensor
$\vec{S}(I,t_o)$ = inertial line of sight unit vector of the "i-th" sensor at time t_o. Body attitude angles at t_o are used.

Next, the computer carries out an update of the vehicle inertial state vector to the time, t_o. We call this vector $\vec{SV}(t_o)$, which contains three components of position, and three components of velocity, in inertial coordinates measured with respect to the center of the earth. Taking the inverse of the position components of the vehicle state vector as a vector toward the center of the earth, the computer checks the center lines of each sensor against this direction to see if the earth is blocking its field of view. Since the remainder of the field may be usable when part of the field is blocked, this angle test is not conclusive. But, it may avoid numerous unnecessary computation cycles later in the selection process.

First, the computer determines the value of the angle λ_E in degrees given by the relation:

$$\lambda_E = \sin^{-1}({}^{Re}/|\vec{SV}_{pos}(t_o)|)$$

where:R_e= radius of the earth

Next, an additional 2 degrees is added to λ_E to obtain the effective earth occultation angle λ_{ED}. This will positively prevent the use of a star which could be situated close enough to the atmosphere to cause a refractive error (See Figure 5). In testing the sensor centerlines, the semi-field of view angle 8.5 degrees is subtracted from λ_{ED}. Then, if the angle between the sensor and the earth center is less than the remainder, that sensor will be completely blocked and may be ignored for the rest of the calculations at time t_o.

A second test is made to see if any one of the sensors is too close to the sun to be used. If the angle between the sun and the centerline is less than 30 degrees, that sensor will also be ignored in the procedures that follow. After these initial exclusion tests, the computer systematically checks each star in the catalog to see if it falls within the field of a usable sensor. In equation form,

If $\cos^{-1}[\vec{S}(I,t_o) \cdot \vec{ST}(J)] < 8.5^\circ$, the "j-th" star is within the field of view of the "i-th" sensor.

Next, follows a logical set of tests to see if that star is actually usable.

First, the angle between the star and the center of the earth is compared to the effective earth occultation angle λ_{ED}, defined in the previous section. If λ_{ED} is the larger of the two, then that star is excluded. This is accomplished by the test:

If $\lambda_{ED} > \cos^{-1}[-\vec{SV}(t_o) \cdot \vec{ST}(J)]$, the "J" star will be excluded.

Next, the angle between the moon and the star is determined and compared to a constant angle of 5 degrees. If the star is within this angular radius of the moon, it also will be excluded.

For those stars which successfully pass these tests, the computer stores the following data:

a. Star catalog number, J

b. Sensor number, I

c. Time of the calculation, t

Sighting Strategy Logic Equations - After all the usable sensors have been tested with all the catalog stars at the time t_o, there may exist zero, one, or several available stars in the list constructed. If there are two or more stars available, the computer can immediately determine their acceptability for alinement by making the following test:

> If $\cos^{-1}(\overrightarrow{ST}(M) \cdot \overrightarrow{ST}(N)) > 35^{\circ}$, and $< 145^{\circ}$, the two stars with catalog numbers M and N are acceptable.

But if, as frequently happens, there are not two available stars which meet this separation angle requirement at time t_o, the computer attempts to predict when two stars could be found if a delay of up to ten minutes were allowed. It is assumed that there will be no control thruster firings during the delay so that the body attitude rates existing at time t_o may be expected to continue. Using the following equations, the computer predicts the vehicle body attitude angles in one minute steps:

$$\gamma(t) = \gamma(t_o) + \dot{\gamma}(t_o)[t - t_o]$$
$$\delta(t) = \delta(t_o) + \dot{\delta}(t_o)[t - t_o]$$
$$\eta(t) = \eta(t_o) + \dot{\eta}(t_o)[t - t_o]$$

After each of these computations, the star search procedure is repeated and the available star list is augmented until two or more stars have been found with acceptable separation angles, or until the ten minute delay limit has been exceeded. In the latter case, the crew is informed that an alinement is not possible for the existing conditions. If the computer does find an acceptable star pair, it requests the crew for permission to inhibit the attitude control thrusters for the required period. If thus allowed, the computer will plan the sighting schedule and carry out the alinement.

Auto-Optics Command Equations - When it has been established that a suitable star is within the field of view of a particular sensor, the computer carries out an angular transformation to determine the elevation and azimuth angles of the star in that sensor's field of view. The equations are:

$$\overrightarrow{ST}_{Auto}(\alpha, \beta) = [SI2EA](NB2SI)[O2B][I2O]\ \overrightarrow{ST}(J)$$

The values of α and β are then converted to properly scaled digital pulse trains and sent to the appropriate sensor. The sensor will establish a square search raster centered about the values of α and β, and measuring two degrees on a side. It starts at $(\alpha - 1^\circ, \beta + 1^\circ)$ and moves to $(\alpha + 1^\circ)$ in one-quarter degree steps. Then, elevation is reduced by one-quarter degree, azimuth goes back to $(\alpha - 1^\circ)$, and the horizontal trace is repeated. This continues until the raster is covered at $(\alpha + 1^\circ, \beta - 1^\circ)$. Each dwell point lasts for 230 microseconds, and each full-line retrace takes 100 microseconds, so that the square search is completed in about 17 milliseconds.

If no star is found, the sensor will automatically extend its raster to the full field of view and determine the location of the brightest star (differential must be one star magnitude) within that area. This full-field search is performed at a rate of 230 microseconds per point, and one-quarter degree steps, and can be completed in approximately one second.

"Mark" Data Processing Equations - At the time of the sighting mark, t_{mark}, the computer is provided with five angles; namely, the star's azimuth and elevation in the sensor field of view, and roll, pitch and yaw of the vehicle relative to the nominal inertial reference frame. The computer makes use of the following successive transformations to obtain the measured unit vector toward the star in inertial coordinates:

$$\vec{ST}_{meas}(J) = [I2O]^T[O2B]^T[NB2SI]^T[SI2EA]^T \vec{ST}_{meas}(\alpha, \beta)$$

This unit vector is stored for later application in checking the identity of the star, and in computing the amount of IRU misalinement present.

Star Identification Check - After completing the "mark" data processing for the first star, the computer returns the IRU alinement procedure to a stand-by status until the second star acquisition can be attempted. After the second star appears, the acquisition sighting and data processing are carried out in an identical fashion to that used for the first star.

As a check upon the proper identification of the two stars, the angle between the measured unit vectors to the two stars is compared to the angle computed using the star catalog unit vectors. This is carried out by the equations:

$$\Delta\lambda_{ss} = \left|\cos^{-1}\left\{\vec{ST}(K) \cdot \vec{ST}(L)\right\}\right| - \left|\cos^{-1}\left\{\vec{ST}_{meas}(K) \cdot \vec{ST}_{meas}(L)\right\}\right|$$

where K= the star catalog number for the first star
L= the star catalog number for the second star
$\Delta\lambda_{ss}$= star angle difference, in degrees

Rejection of a set of sighting data may be made on the basis of this check. If $|\Delta\lambda_{ss}| > 0.072$ degree, then the data should be rejected. However, if the data is not rejected, it will still not be considered acceptable until a later check has been made. Rejection of data at this point may be a result of an erroneous star acquisition, or may be an indicator of sensor failure. It is planned to implement logical test sequences of the individual sensors with available stars to determine their performance whenever a data test failure of the above type occurs. These equations have not yet been developed.

Calculation of Gimbal Torquing Angles - The procedure for determining the angles through which the gyros must be torqued for alinement is based upon similar procedures used in the Apollo Guidance Computer (Reference 2). First, the computer constructs two dummy coordinate systems, one for the star catalog unit vectors (unprimed system), and the other (primed system) set up from the measured star unit vectors.

The primed coordinate axes are given by the unit vectors computed from:

$$\bar{U}'_x = \vec{ST}_{meas}(K)$$

$$\bar{U}'_y = \frac{\vec{ST}_{meas}(K) \times \vec{ST}_{meas}(L)}{|\vec{ST}_{meas}(K) \times \vec{ST}_{meas}(L)|}$$

$$\bar{U}'_z = \bar{U}'_x \times \bar{U}'_y$$

where: K and L are the star catalog numbers for the first and second stars, respectively.

The unprimed coordinate axes are given similarly by the equations:

$$\bar{U}_x = \vec{ST}(K)$$

$$\bar{U}_y = \frac{\vec{ST}(K) \times \vec{ST}(L)}{|\vec{ST}(K) \times \vec{ST}(L)|}$$

$$\bar{U}_z = \bar{U}_x \times \bar{U}_y$$

The correct gyro torquing angles $\Delta\gamma$, $\Delta\delta$, and $\Delta\eta$ can be found from solution of the general equation:

$$\begin{bmatrix} \bar{U}_x \\ \bar{U}_y \\ \bar{U}_z \end{bmatrix} = \begin{bmatrix} T_{11} & T_{12} & T_{13} \\ T_{21} & T_{22} & T_{23} \\ T_{31} & T_{32} & T_{33} \end{bmatrix} \begin{bmatrix} \bar{U}'_x \\ \bar{U}'_y \\ \bar{U}'_z \end{bmatrix}$$

where the elements of the transformation matrix are functions of the angles desired. To obtain a unique solution,an order of operation is defined as follows: the first rotation is about the U_x axis by the angle $\Delta\gamma$ to form the $\bar{U}''_x$, $\bar{U}''_y$, $\bar{U}''_z$ system. Then follows a rotation about the new $\bar{U}''_y$ axis by the angle $\Delta\delta$ to form the $\bar{U}''_x$, $\bar{U}''_y$, $\bar{U}''_z$ system. Last, follows a rotation about the new $\bar{U}''_z$ axis by the angle $\Delta\eta$ to form the

C.3

$\bar{U}_x^{IV}$, $\bar{U}_y^{IV}$, $\bar{U}_z^{IV}$ system.

The elements of the T matrix are given by the following equations:

$$T_{11} = \cos(\Delta\eta) \cos(\Delta\delta)$$
$$T_{12} = \cos(\Delta\eta) \sin(\Delta\delta) \sin(\Delta\gamma) + \sin(\Delta\eta) \cos(\Delta\gamma)$$
$$T_{13} = -\cos(\Delta\eta) \sin(\Delta\delta) \cos(\Delta\gamma) + \sin(\Delta\gamma) \sin(\Delta\eta)$$
$$T_{21} = -\sin(\Delta\eta) \cos(\Delta\delta)$$
$$T_{22} = -\sin(\Delta\eta) \sin(\Delta\delta) \sin(\Delta\gamma) + \cos(\Delta\eta) \cos(\Delta\gamma)$$
$$T_{23} = \sin(\Delta\eta) \sin(\Delta\delta) \cos(\Delta\gamma) + \sin(\Delta\gamma) \cos(\Delta\eta)$$
$$T_{31} = \sin(\Delta\delta)$$
$$T_{32} = -\cos(\Delta\delta) \sin(\Delta\gamma)$$
$$T_{33} = \cos(\Delta\delta) \cos(\Delta\gamma)$$

An explicit solution of this set of equations is required to compute the gyro torquing angles $\Delta\gamma, \Delta\delta$, and $\Delta\eta$.

Final Check Logic Equations - Upon derivation of the gyro torquing angles, the computer checks them against previous gyro drift rates to determine their reasonableness. If the absolute value of all three of the torquing angles agrees within plus or minus 0.05 degree with the absolute value of the previous drift rate data point multiplied by the elapsed time since the previous alinement, then the star identification and sighting data accuracy is confirmed. If the agreement is different by as much as plus or minus 0.10 degree, then the sighting data may be erroneous, or the gyro drift rate may be changing, or the stars may have been incorrectly identified. These possibilities should be checked by further sightings and calibrations. Also, the crew should be notified of a possible malfunction. If the error exceeds ± 0.1 degree, then the star identification is probably in error.

The equations for this logic are as follows:

Let: $$\Delta\Omega_{check} = \left\| \left| \begin{Bmatrix} \Delta\gamma \\ \Delta\delta \\ \Delta\eta \end{Bmatrix} \right| - \left| \begin{Bmatrix} \dot{\Omega}_\gamma \\ \dot{\Omega}_\delta \\ \dot{\Omega}_\eta \end{Bmatrix} (t - t_a) \right| \right\|$$

If the value of $\Delta\Omega_{check} < 0.05^\circ$, recommend acceptance.

If the value of $\Delta\Omega_{check}$ is $\geq 0.05^\circ$, but $< 0.10^\circ$, possible acceptance, but calibrations of gyro and tracker should be performed soon.

If the value of $\Delta\Omega_{\text{check}}$ is ≥ 0.10°, recommend rejection and perform calibrations of sensor and gyros before repeating.

Conclusions and Recommendations

The equations which have been described in this report are preliminary in form and will be varied as required to suit the changing requirements of the Shuttle as they develop. However, these equations are considered acceptable representations of the IRU alinement procedure using the Multi-Mode Optical Sensor.

Further expansion of the logic sections of this procedure should be made to incorporate sensor failure detection schemes. These will probably involve some form of in-flight calibrations of each sensor on a low-priority basis.

The equations developed in this report have been combined into a FORTRAN language digital computer program. Appendix B contains a set of sample results obtained with this program. These should be useful as test cases when these equations have been integrated into a Space Shuttle mission simulator program.

REFERENCES

1. MSC Internal Note, MSC-EG-71-1, "A Functional Description of a Proposed Optical Subsystem for the Space Shuttle Vehicle", by C. Manry, I. Saulietis, and D. Tadlock.

2. Apollo Document R-567, "Guidance System Operations Plan for Manned LM Earth Orbital and Lunar Missions using Program Luminary - Section 5. Guidance Equations (Rev. 1)", MIT/Instrumentation Laboratory, Cambridge, Massachusetts, November 1968.

9.6.2.1 IRU Alignment (continued)

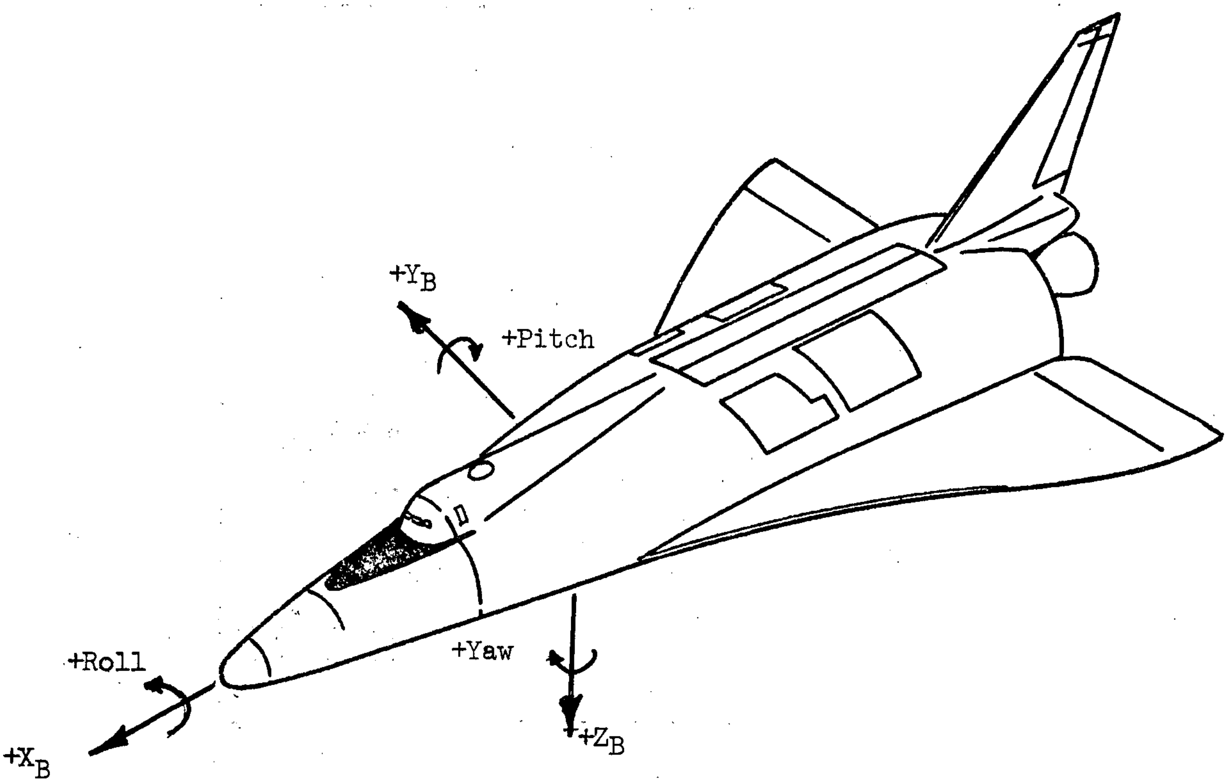


Figure 1. SSV Body Coordinate System

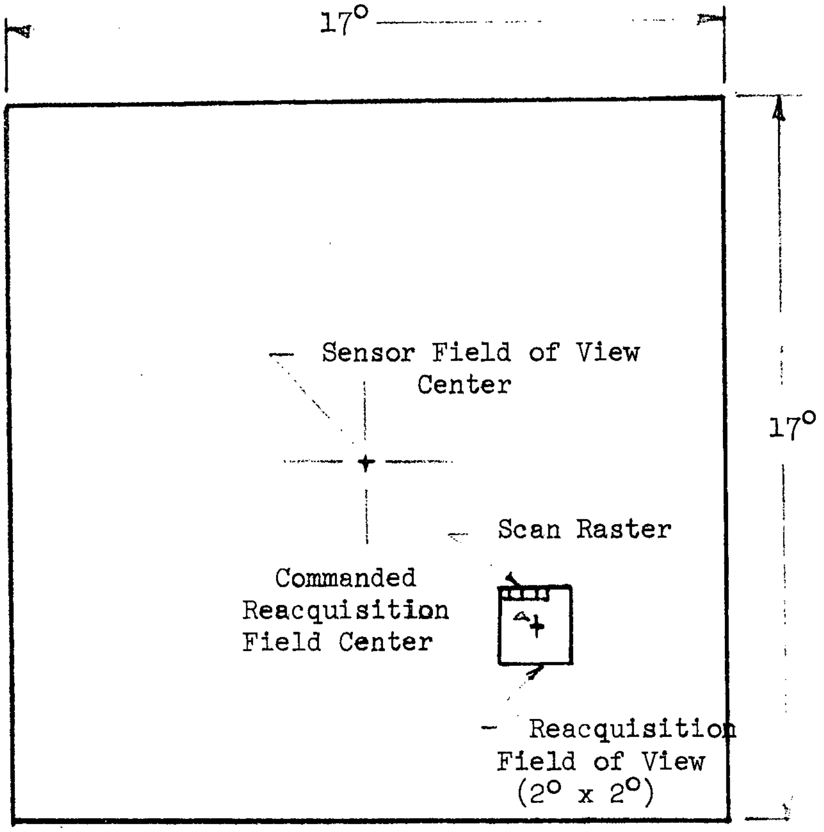

17°
17°
Sensor Field of View Center
Scan Raster
Commanded Reacquisition Field Center
Reacquisition Field of View (2° x 2°)

Figure 2. Multi-Mode Optical Sensor Field of view format in reacquisition mode.

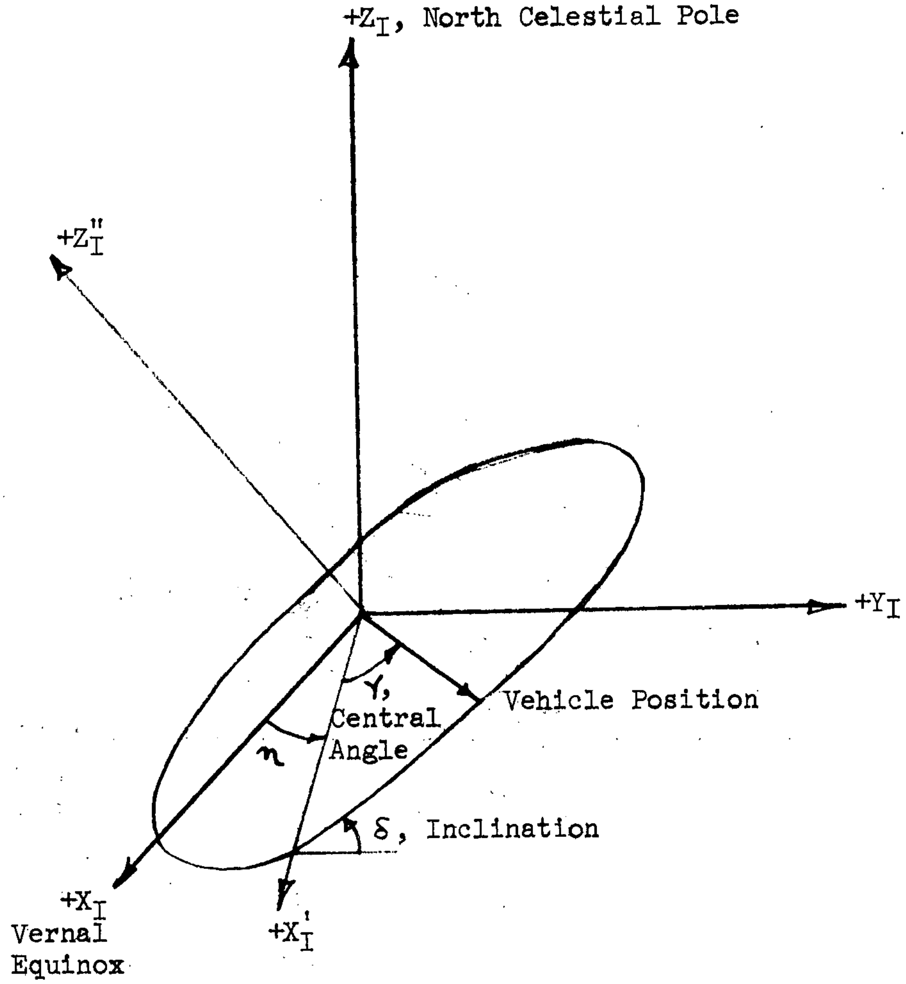


Figure 3. Inertial-to-orbital plane coordinate transformation angles.

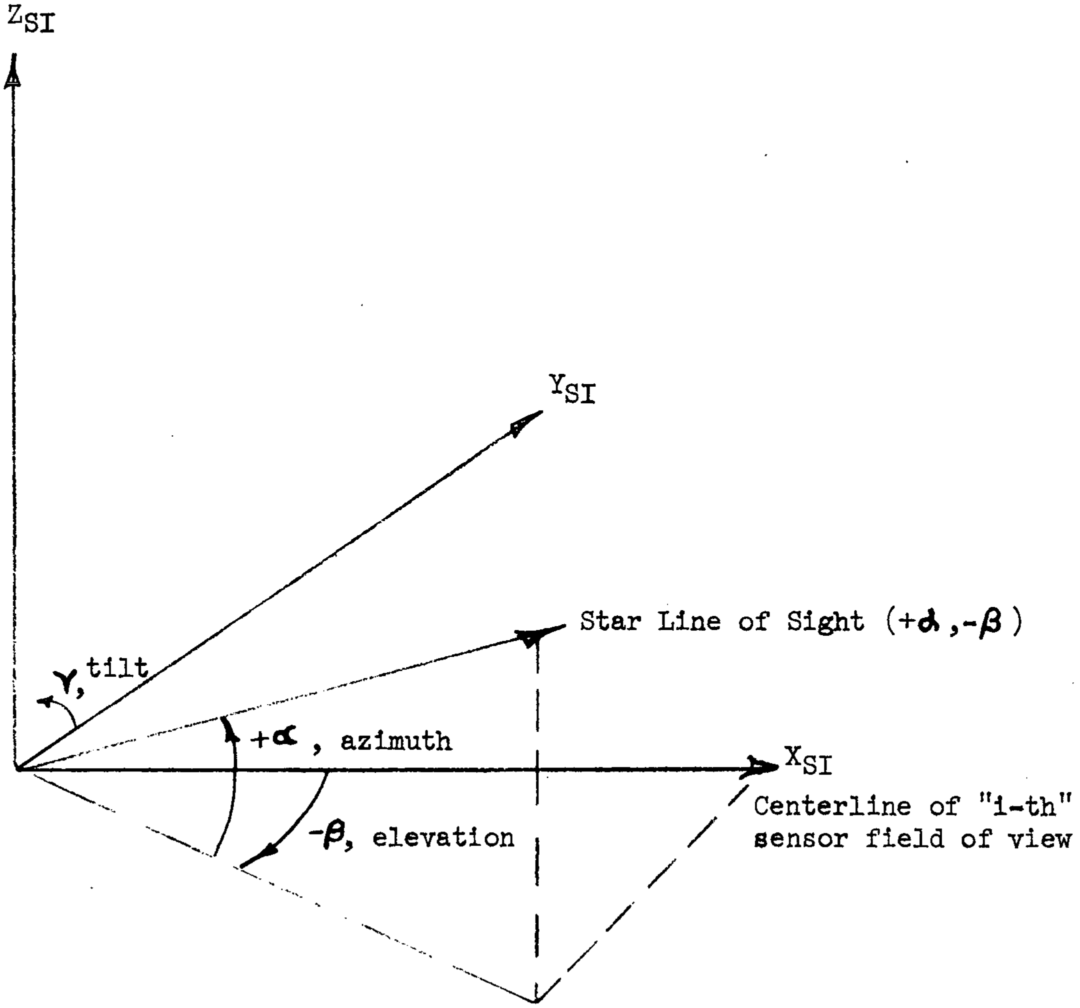

Z_{SI}
Y_{SI}
Star Line of Sight (+α, -β)
γ, tilt
+α, azimuth
X_{SI}
Centerline of "i-th" sensor field of view
-β, elevation

Figure 4. Coordinate transformation from sensor "i" system to elevation/azimuth system [SI2EA].

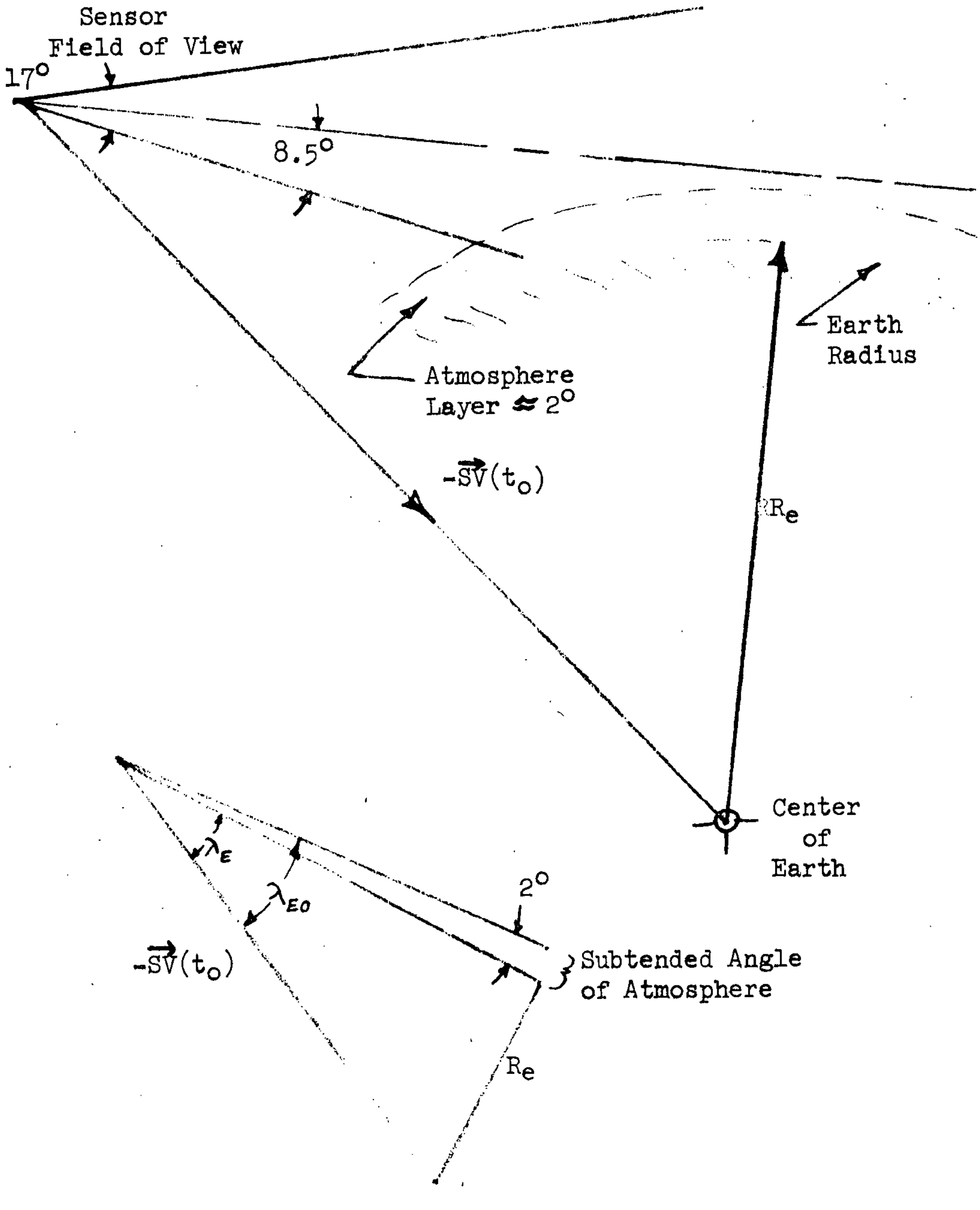


Figure 5. Earth Occultation Test Geometry

APPENDIX A - STAR CATALOG

Catalog Number	Star Name	Spectral Class	Direction Cosines		
			X	Y	Z
1	Alpheratz	+2.15, A0	.87513	.02511	.48324
2	Caph	+2.42, F5	.51542	.01642	.85678
3	Gamma Pegasus	+2.87, B2	.96477	.04816	.25864
4	Beta Horologium	+2.90, G0	.21608	.02265	-.97611
5	Ankaa	+2.44, A3	.73309	.07885	-.67554
6	Schedar	+2.47, K0	.54637	.09268	.83240
7	Diphda	+2.24, K0	.93431	.17246	-.31198
8	Gamma Cass	+2.30, B0	.47811	.11607	.87059
9	Mirach	+2.37, M0	.77941	.23742	.57979
10	Ksora	+2.80, A5	.46641	.17789	.86650
11	Achernar	+0.60, B5	.49161	.21994	-.84259
12	Sheratan	+2.72, A5	.82494	.44187	.35246
13	Polaris	+2.12, F8	.01359	.00780	.99988
14	Almach	+2.28, K0	.63910	.37549	.67124
15.	Hamal	+2.23, K2	.78480	.47714	.39551
16	Mira	+2.00V, M5E	.82394	.56403	.05470
17	Menkar	+2.82, M2	.70405	.70680	.06895
18	Algol	+2.30, B8	.52119	.54866	.65371
19	Mirfak	+1.90, F5	.41149	.49835	.76310
20	Alcyone	+2.96, B5P	.50622	.76048	.40671
21	Menkhib	+2.91, B1	.45070	.72075	.52668
22	Eper	+2.96, B1	.39650	.65662	.64159
23	Aldebaran	+1.06, GK5	.35187	.89224	.28302
24	Hassaleh	+2.90, K2	.23540	.80384	.54628
25	Cursa	+2.92, A3	.23197	.96860	-.08942
26	Rigel	+0.34, CB8	.20215	.96881	-.14333
27	Capella	+0.21, GK0	.13828	.68121	.71891
28	Bellatrix	+1.70, B2	.15869	.98117	.11007
29	Elnath	+1.78, B8	.13708	.86739	.47838

APPENDIX A (Cont'd)

Catalog Number	Star Name	Spectral Class	Direction Cosines X	Y	Z
30	Nihal	+2.96, GO	.13518	.92509	-.35488
31	Mintaka	+2.48, BO	.12959	.99155	-.00565
32	Arneb	+2.69, FO	.11936	.94437	-.30646
33	Hatysa	+2.87, OE5	.11380	.98811	-.10334
34	Alnilam	+1.75, BO	.11128	.99356	-.02134
35	Zeta Taurus	+3.00, B3P	.09932	.92751	.36037
36	Phakt	+2.75, B5P	.07801	.82446	-.56052
37	Alnitak	+2.05, BO	.09148	.99522	-.03420
38	Saiph	+2.20, BO	.05977	.98395	-.16816
39	Betelgeuse	+0.00V, MO	.02909	.99124	.12883
40	Menkalinan	+2.07, AOP	.00939	.70771	.70644
41	Theta Auriga	+2.71, AOP	.00927	.79635	.60477
42	Tejat Prior	+3.00V, MO	-.05143	.92232	.38298
43	Mirzam	+1.99, B1	-.08771	.94734	-.30797
44	Canopus	-0.86, FO	-.06121	.60322	-.79522
45	Alhena	+1.93, AO	-.14877	.94756	.28285
46	Sirius	-1.37, AO	-.18118	.94070	-.28680
47	Tau Puppis	+2.83, KO	-.13494	.62060	-.77243
48	Adhara	+1.63, B1	-.21638	.84810	-.48364
49	Wezen	+1.98, F8P	-.25816	.85821	-.44364
50	Pi Puppis	+2.74, K5	-.25959	.75490	-.60228
51	Aludra	+2.43, B5P	-.30813	.81642	-.48838
52	Castor	+1.58, AO	-.33274	.78038	.52942
53	Procyon	+0.48, F5	-.41081	.90700	.09265
54	Pollux	+1.21, KO	-.38381	.79414	.47120
55	Naos	+2.27, OD	-.39036	.66038	-.64150
56	Rho Puppis	+2.88, F5	-.47675	.77758	-.40996
57	Gamma Velorum	+1.90, OO	-.36097	.57513	-.73412
58	Avior	+1.74, KO	-.29527	.41472	-.86071
59	Delta Velorum	+2.01, AO	-.37973	.43784	-.81492
60	Suhail	+2.22, K5	-.52953	.49941	-.68570
61	Miaplacidus	+1.80, AO	-.26021	.23260	-.93712

APPENDIX A (Cont'd)

Catalog Number	Star Name	Spectral Class	Direction Cosines X	Y	Z
62	Tureis	+2.25, FO	-.38748	.33638	-.85832
63	Kappa Velorum	+2.63, B3	-.44256	.36798	-.81776
64	Alphard	+2.16, K2	-.77364	.61612	-.14791
65	Vel	+3.00, K5	-.43372	.33233	-.83752
66	Regulus	+1.34, B8	-.86020	.46458	.21028
67	Algieba	+2.61, KO	-.84813	.40433	.34233
68	Mu Velorum	+2.84, G5	-.61855	.20919	-.75739
69	Merak	+2.44, AO	-.53198	.14319	.83456
70	Dubhe	+1.95, KO	-.45519	.11860	.88246
71	Zosma	+2.58, A3	-.91512	.19347	.35373
72	Denebola	+2.23, A2	-.96548	.05366	.25490
73	Phecda	+2.54, AO	-.58899	.02058	.80788
74	Delta Centauri	+2.88, B3P	-.63545	-.01813	-.77193
75	Gienah	+2.78, B8	-.95273	-.05829	-.29817
76	Acrux	+1.00, B1	-.45283	-.04887	-.89025
77	Gamma Acrux	+1.61, M3	-.54142	-.06939	-.83789
78	Kraz	+2.84, G5	-.90986	-.13008	-.39400
79	Alpha Muscis	+2.94, B3	-.35510	-.05478	-.93322
80	Gamma Centauri	+2.38, AO	-.64931	-.11326	-.75204
81	Arich	+2.91, FO	-.98466	-.17311	-.02195
82	Beta Crucis	+1.50, B1	-.49753	-.10046	-.86161
83	Alioth	+1.68, AOP	-.54248	-.12648	.83049
84	Chara	+2.90, AOP	-.76060	-.18399	.62261
85	Vindemiatrix	+2.95, KO	-.94722	-.25573	.19333
86	Icen	+2.91, A2	-.75674	-.27029	-.59522
87	Mizar	+2.40, A2P	-.53536	-.20155	.82023
88	Spica	+1.21, B2	-.91749	-.34919	-.19045
89	Hya	+3.00 V, M7E	-.85314	-.34381	-.39236
90	Epsilon Centauri	+2.56, B1	-.54432	-.24708	-.80166
91	Alkaid	+1.91, B3	-.58115	-.29027	.76026
92	Mufrid	+2.80, GO	-.83498	-.44873	.31850
93	Beta Centauri	+0.86, B1	-.42887	-.25095	-.86781

9.6.2.1 IRU Alignment (continued)

APPENDIX A (Cont'd)

Catalog Number	Star Name	Spectral Class	Direction Cosines X	Y	Z
94	Menkent	+2.26, KO	-.69059	-.41748	-.59059
95	Arcturus	+0.24, GKO	-.78657	-.52093	.33157
96	Haris	+3.00, FO	-.61983	-.47849	.62198
97	Eta Centauri	+2.65, B3P	-.58301	-.46074	-.66920
98	Alpha Centauri	+0.06, GO	-.37878	-.31005	-.87200
99	Alpha Lupus	+2.89, B2	-.52087	-.43549	-.73419
100	Izar	+2.70, KO	-.67250	-.58181	.45742
101	Zuben A	+2.90, A3	-.71208	-.64646	-.27392
102	Kochab	+2.24, K5	-.19890	-.18351	.96269
103	Beta Lupus	+2.81, B2P	-.52563	-.50862	-.68193
104	Zuben B	+2.74, B8	-.65037	-.74239	-.16083
105	Gamma Lupus	+2.95, B3	-.45174	-.60386	-.65672
106	Alphecca	+2.31, AO	-.53327	-.71547	.45137
107	Cor Serpentis	+2.75, KO	-.56079	-.82010	.11380
108	Pi Scorpii	+3.00, B2	-.46040	-.77179	-.43860
109	Dschubba	+2.54, BO	-.46792	-.79644	-.38306
110	Acrab	+2.90, B1	-.45856	-.82217	-.33728
111	Sigma Scorpii	+2.87, B1	-.38478	-.81637	-.43068
112	Eta Draconis	+2.89, G5	-.19442	-.43419	.87959
113	Antares	+1.22, GKO	-.35279	-.82368	-.44395
114	Kornephoros	+2.81, KO	-.36071	-.85720	.36755
115	Tau Scorpii	+2.91, BO	-.32424	-.81998	-.47170
116	Zeta Ophiuchi	+2.70, BO	-.35546	-.91676	-.18220
117	Zeta Herculis	+3.00, GO	-.29123	-.79976	.52495
118	Atria	+1.88, K2	-.11546	-.33984	-.93337
119	Epsilon Scorpii	+2.36, KO	-.25586	-.78619	-.56254
120	Sabik	+2.63, A2	-.21507	-.93844	-.27032
121	Beta Ara	±2.80, K2	-.09255	-.55881	-.82412
122	Upsilon Scorpii	+2.80, B3	-.10944	-.78823	-.60557
123	Alpha Ara	+2.97, B3P	-.08658	-.63894	-.76437
124	Alwaid	+2.99, GO	-.08072	-.60581	.79151
125	Shaula	+1.71, B2	-.09988	-.79151	-.60293

APPENDIX A (Cont'd)

Catalog Number	Star Name	Spectral Class	Direction Cosines X	Y	Z
126	Rasalhague	+2.14, A5	-.11341	-.96937	.21787
127	Theta Scorpii	+2.04, F0	-.08028	-.72720	-.68171
128	Kappa Scorpii	+2.51, B2	-.06750	-.77406	-.62951
129	Kelb-Alrai	+2.94, K0	-.07933	-.99364	.07986
130	Eltanin	+2.42, K5	-.01143	-.62252	.78252
131	Kaus Medius	+2.84, K0	.07090	-.86447	-.49766
132	Kaus-Australis	+1.95, A0	.07854	-.82134	-.56501
133	Kaus-Bor.	+2.94, K0	.10148	-.89729	-.42961
134	Vega	+0.14, A0	.12117	-.77041	.62592
135	Nunki	+2.14, B3	.20576	-.87222	-.44373
136	Ascella	+2.71, A2	.22570	-.83671	-.49897
137	Delta Cygni	+2.97, A0	.30940	-.63521	.70766
138	Reda	+2.80, K2	.43329	-.88254	.18269
139	Altair	+0.89, A5	.45280	-.87846	.15255
140	Eta Aquila	+3.00V, GOP	.46436	-.88550	.01596
141	Sador	+2.32, F8P	.44111	-.62432	.64471
142	Peacock	+2.12, B3	.31928	-.44396	-.83723
143	Deneb Cygni	+1.33, A2	.45385	-.53974	.70901
144	Gienah	+2.64, K0	.54710	-.62498	.55685
145	Alderamin	+2.60, A5	.35152	-.30091	.88650
146	Enif	+2.54, K0	.81344	-.55663	.16872
147	Deneb Algiedi	+2.98, A5	.79836	-.53289	-.28045
148	Naquir	+2.16, B5	.59793	-.32453	-.73292
149	Alpha Tucanae	+2.91, K2	.44359	-.21607	-.86980
150	Beta Grus	+2.24, M3	.64064	-.23132	-.73217
151	Formalhaut	+1.29, A3	.83379	-.24019	-.49710
152	Scheat	+2.61, M0	.85575	-.22102	.46781
153	Markab	+2.57, A0	.93615	-.23768	.25910

APPENDIX B
TEST RESULTS

Introduction

The computer program, which has been developed from the equations derived in the report, is intended to serve two purposes. First, it is to be submitted formally as a baseline equations document for use in on-board computer simulation studies. Secondly, it will be used to provide solutions to technical problems encountered in the development of the MMOS itself. This appendix is primarily to provide a summary of typical results for specified test cases which can be used to check the performance of the submitted program after it has been incorporated into a much larger on-board computer simulation program. However, a few results have been included which may be of interest principally to those involved in the hardware development task.

Task Method

Locations for three sensors were hypothesized as shown in Table B-1. The orientations may be visualized simply as three equally-spaced directions in the X-Y plane (yaw plane) of the body coordinate system. Sensor number 1 looks along the +X axis, sensor number 2 looks behind the right wing, and sensor number 3 looks behind the left wing of the vehicle.

Table B-1. Sensor Locations

Sensor Number	γ	δ	η
1	0°	0°	0°
2	0°	0°	120°
3	0°	0°	240°

Six possible orbit conditions were assumed to provide fairly complete utilization of the star catalog. The coordinate angles for these conditions are shown in Table B-2.

Table B-2. Orbit Conditions

Name	Longitude of the Ascending Node Ω	Orbit Inclination δ
Ecliptic	0°	33°
Galactic	281°	62°
Equatorial	0°	0°
Perpendicular to Ecliptic	180°	67°
Perpendicular to Galactic	101°	28°
Perpendicular to Equatorial	0°	90°

In the orbital plane, the orbital angular velocity $\dot{\gamma}$ was set at 0.067 degrees per second to correspond to an altitude H of 270 nautical miles. The sun and the moon were assigned arbitrary locations at opposite sides of the celestial sphere at the intersections between the ecliptic plane and the galactic plane. In this way, maximum interference with catalog stars was obtained. Table B-3 contains the sun and moon location data that was used.

Table B-3. Sun and Moon Unit Vectors

	X	Y	Z
Sun	0.000	0.906	0.423
Moon	0.000	-0.906	-0.423

At the beginning of each simulated orbit, the vehicle was oriented in a wings level, "heads up" attitude, with the +X axis directed along the forward velocity vector. The angular rotations required to obtain this attitude (referenced to the orbital plane system) are given in Table B-4. An attitude-hold mode about the nominal attitude was also established. The angular deadband limits were set at plus and minus 5 degrees, with the angular velocities as given in Table B-4.

9.6.2.1 IRU Alignment (continued)

Table B-4. Vehicle Attitude and Angular Velocities

Axis	Attitude (Degrees)	Angular Velocity (Degrees/Second)
Roll	-90	0.012
Pitch	-90	0.008
Yaw	0	0.170

During simulated orbit, the program carries out the state vector update, and the star search and selection processes at one-half minute time intervals. At each time point, the vehicle's position vector in the inertial coordinate system, and the unit vectors for the center line-of-signt of each sensor are printed out. Whenever the star selection is completed successfully, the catalog numbers of the selected stars, and the numbers of the usable sensors also are given. Also given are the times at which each star will be available, and the expected star separation angle.

In order to represent the more important aspects of this output data, the format shown in Figures B-1 through B-6 has been selected. The orbital time in minutes is displayed along the horizontal axis. In the upper graph, the possibility of performing an immediate IRU alinement is indicated by a raised section of the curve. If the alinement cannot be accomplished at that moment, the curve is lowered until a later trial gives a positive result.

When the upper curve has its lower value, the solution may consist of inhibiting the attitude hold mode and waiting until two stars can be acquired. A maximum time of ten minutes has been arbitrarily allowed for this inhibit process. The lower curve indicates the magnitude of the delay in minutes (after the current time) before the second star could be acquired. In the results illustrated in these six cases, there were no instances in which the delay exceeded the ten minutes allowed.

The figures have been arranged in a sequence of increasing percentage of immediate alinements. The orbit about the north galactic pole was the worst case. After about seven minutes, the forward-looking sensor experienced sun interference and was not usable again until about 22 minutes elapsed time. During that interval, the other two sensors could have provided an alinement capability if the thrusters had been inhibited for as long as three minutes. In the period from about 25 minutes until about 35 minutes, the forward-looking sensor was passing through the sparsely-populated region around the north celestial pole, although the other two sensors were capable of providing the alinement if an inhibit time of one or two minutes could be accepted. In the ten-minute interval beginning at 56 minutes elapsed time, the forward-looking sensor is having some interference from the moon. Then, after a brief period of alinement capability, the same sensor passes into a low-density region near the south celestial pole. However, as the spacecraft again approaches the initial orbit position, the capability for immediate alinement is also regained.

The point of maximum delay time was reached at 62 minutes elapsed time. A delay of six minutes was predicted; however, an immediate alinement was found to be possible four minutes later, using an entirely different pair of stars and sensors. This type of occurrence is highly typical for the three-sensor mode of operation in all of the conditions tested so far. Further tests will be necessary to determine an optimum balance between the maximum allowable delay time and the percentage of trials in which thruster inhibit is required. For this set of conditions, a maximum delay time of ten minutes resulted in a worst-case percentage of about 67 percent for the thruster inhibit condition. This is also a function of the field of view size as will be illustrated next.

Figure B -7 presents the results of reducing the field of view radius from 8.5 degrees to 5.0 degrees. The orbit perpendicular to the ecliptic plane as shown in Figure B -2 was selected for this test. The vertical axis on the left side of the figure shows the 180 time points during the orbit at which the star selection process was initiated. The horizontal axis is approximately proportional to the field of view area indicated by the square of the angular radius. These results indicate that the percentage of immediate alinements can be fairly accurately estimated for any reasonable sensor field of view, if the performance at one field size is known for the desired orbital condition. If this indication proves valid in further tests, a considerable number of computer runs can be eliminated. Another interesting fact to be obtained from this graph is that the average delay time in completing the alinement increases more rapidly than the inverse square relationship anticipated from the field of view area reduction. The trade-off between the number of active sensors and the field of view size required will be very dependent upon the maximum delay time allowed for the thruster inhibit.

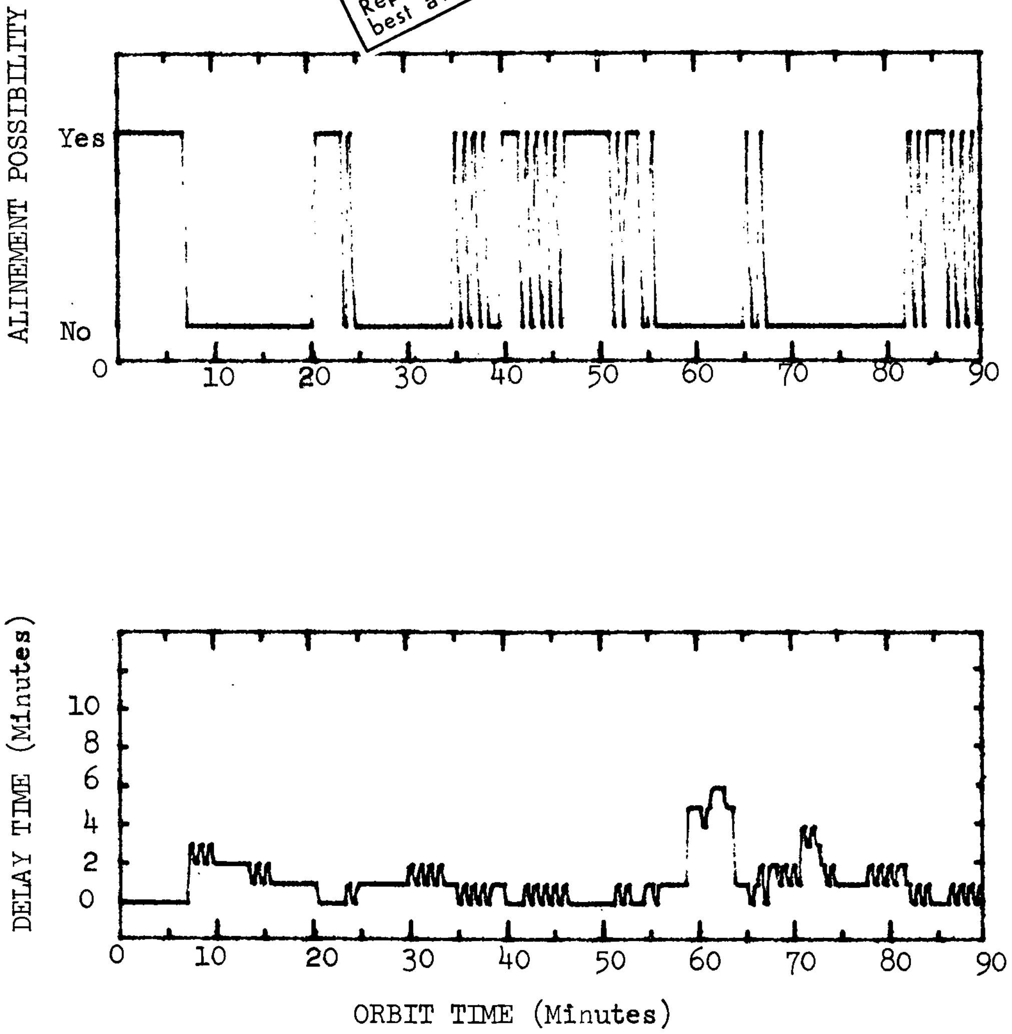


Figure B-1. Alinement possibility and delay time results for an orbit about the north galactic pole. Immediate alinement was possible in 33.2 percent of the trials.

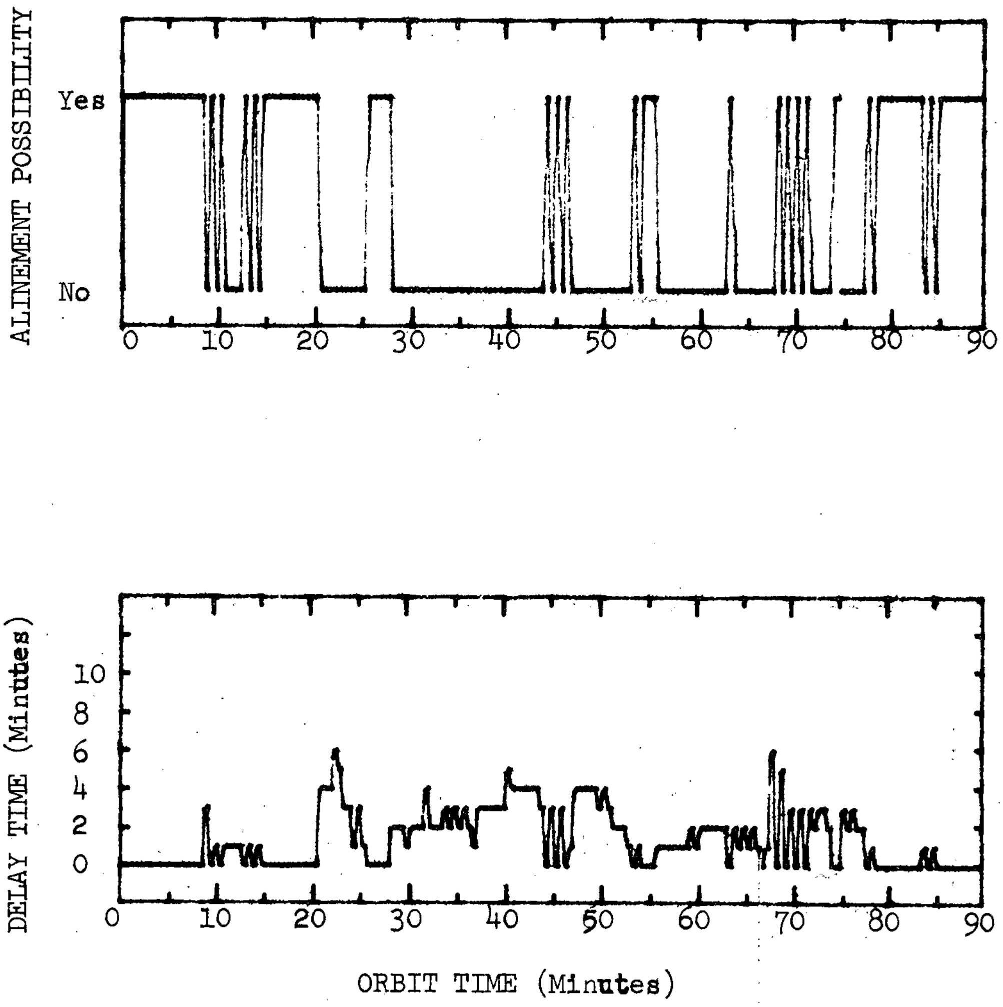


Figure B-2. Alinement possibility and delay time results for an orbit perpendicular to the ecliptic plane. Immediate alinement was possible in 42.2 percent of the trials.

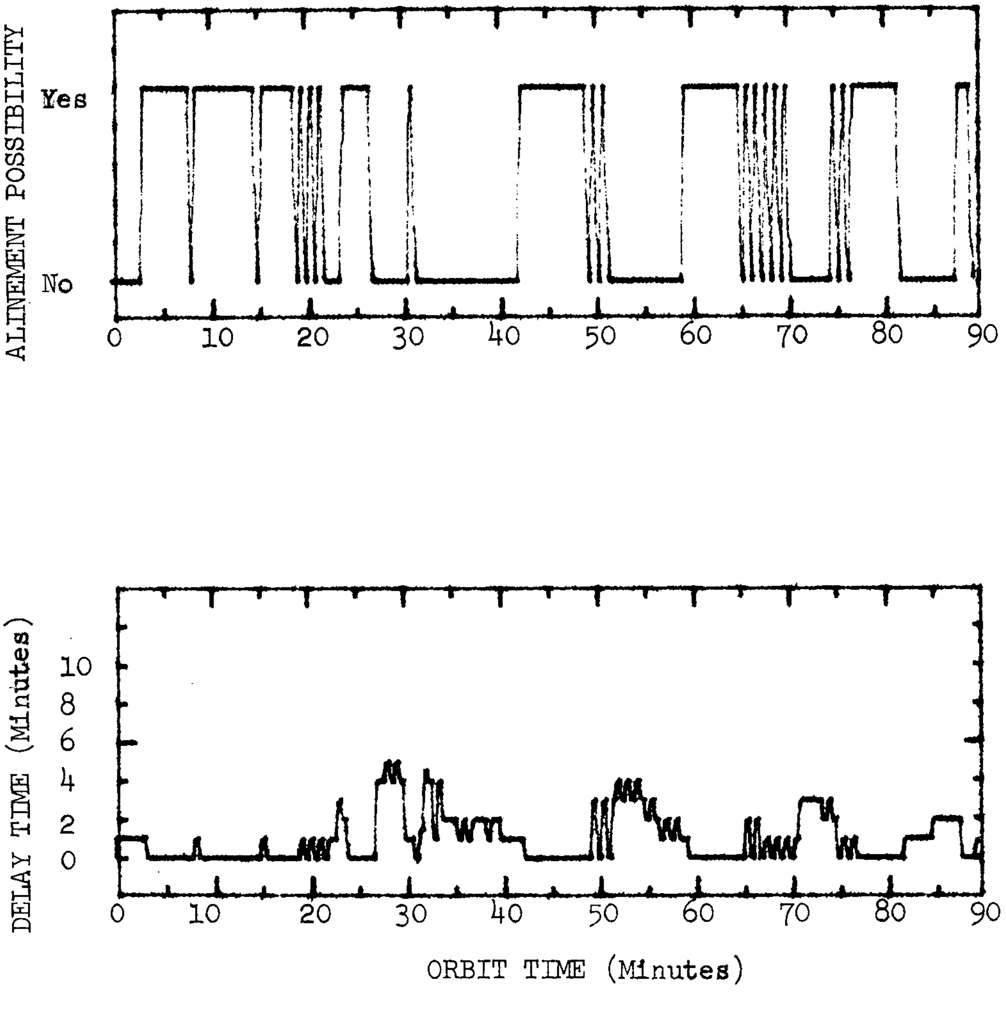


Figure B-3. Alinement possibility and delay time results for an orbit perpendicular to the galactic plane. Immediate alinement was possible in 49.4 percent of the trials.

9.6.2.1 IRU Alignment (continued)

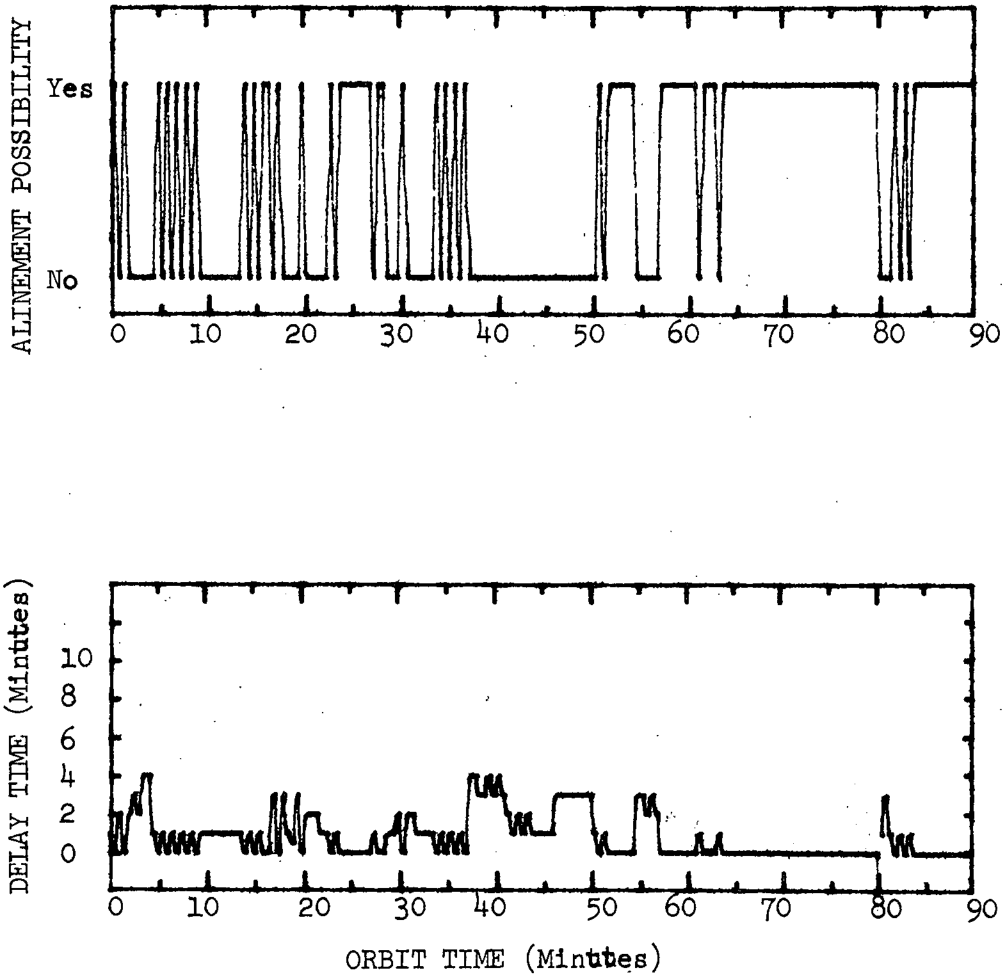


Figure B-4. Alinement possibility and delay time results for an orbit in the equatorial plane. Immediate alinement was possible in 52.7 percent of the trials.

9.6.2.1 IRU Alignment (continued)

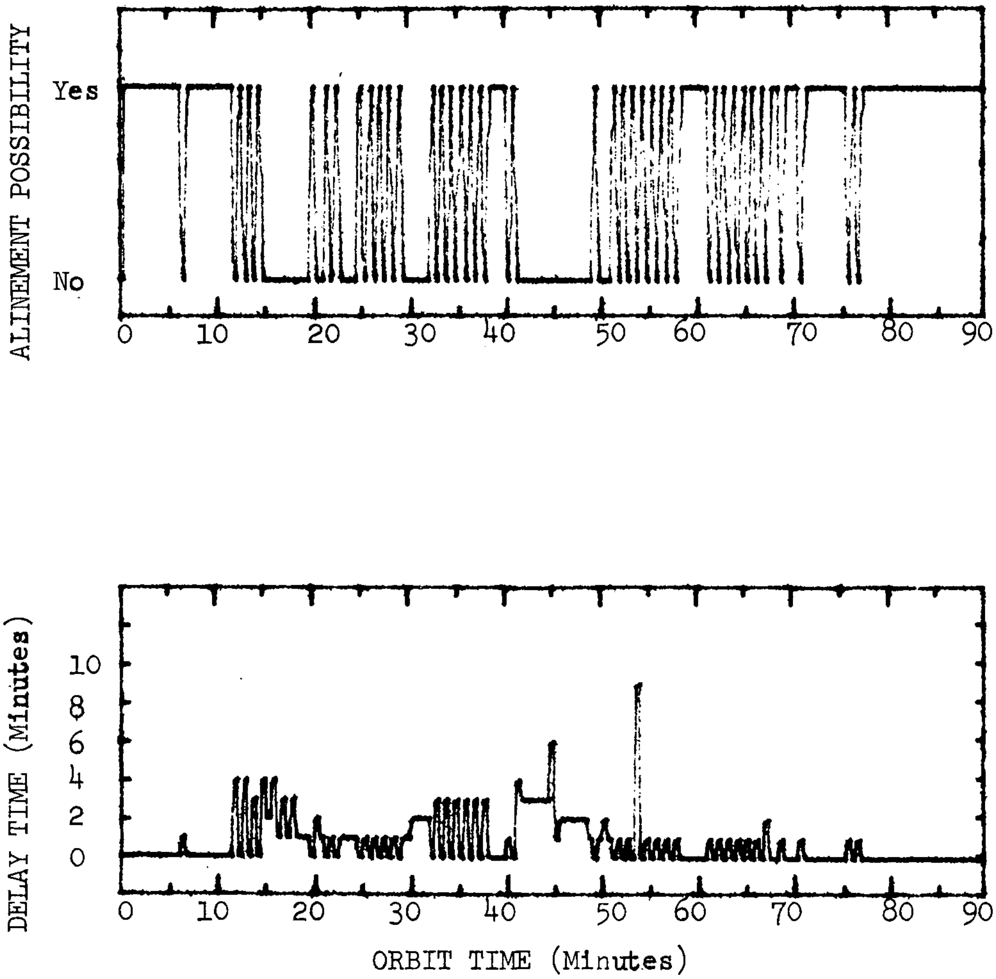


Figure B-5. Alinement possibility and delay time results for an orbit perpendicular to the equatorial plane. Immediate alinement was possible in 59.5 percent of the trials.

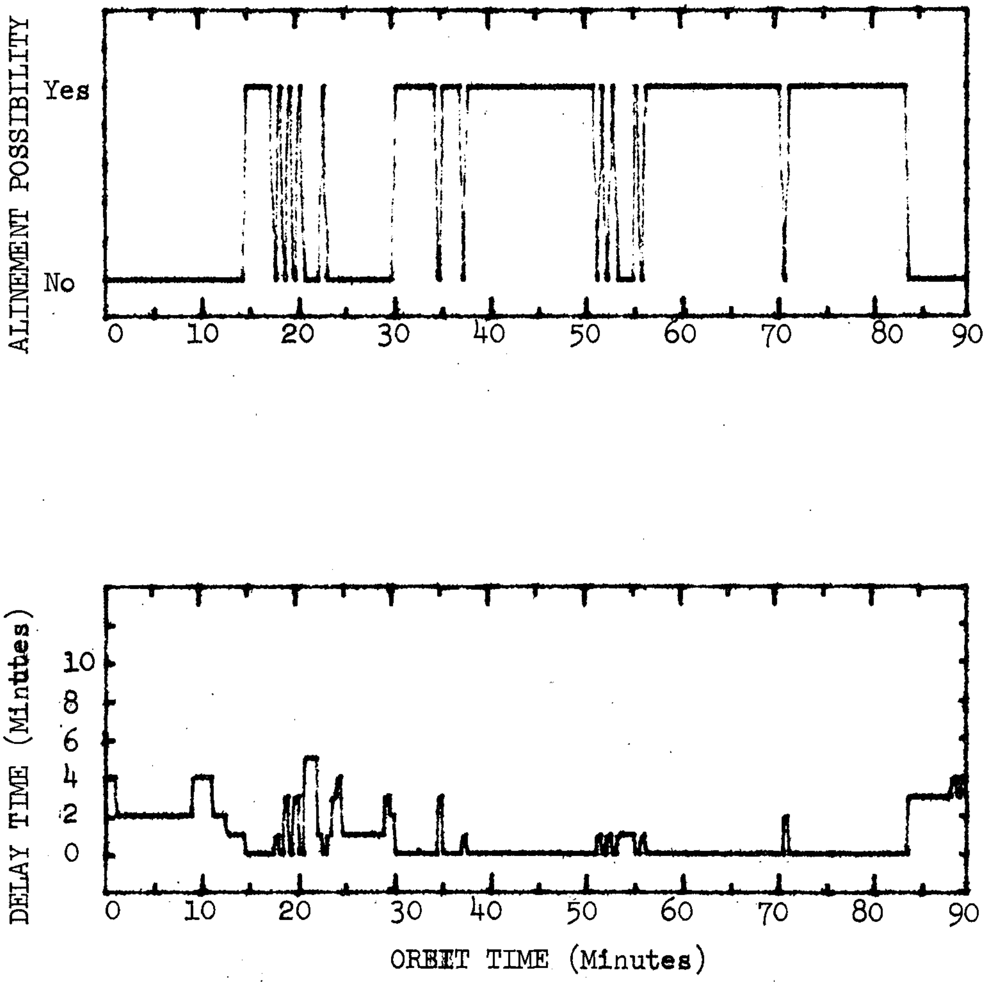


Figure B-6. Alinement possibility and delay time results for an orbit in the ecliptic plane. Immediate alinement was possible in 59.5 percent of the trials.

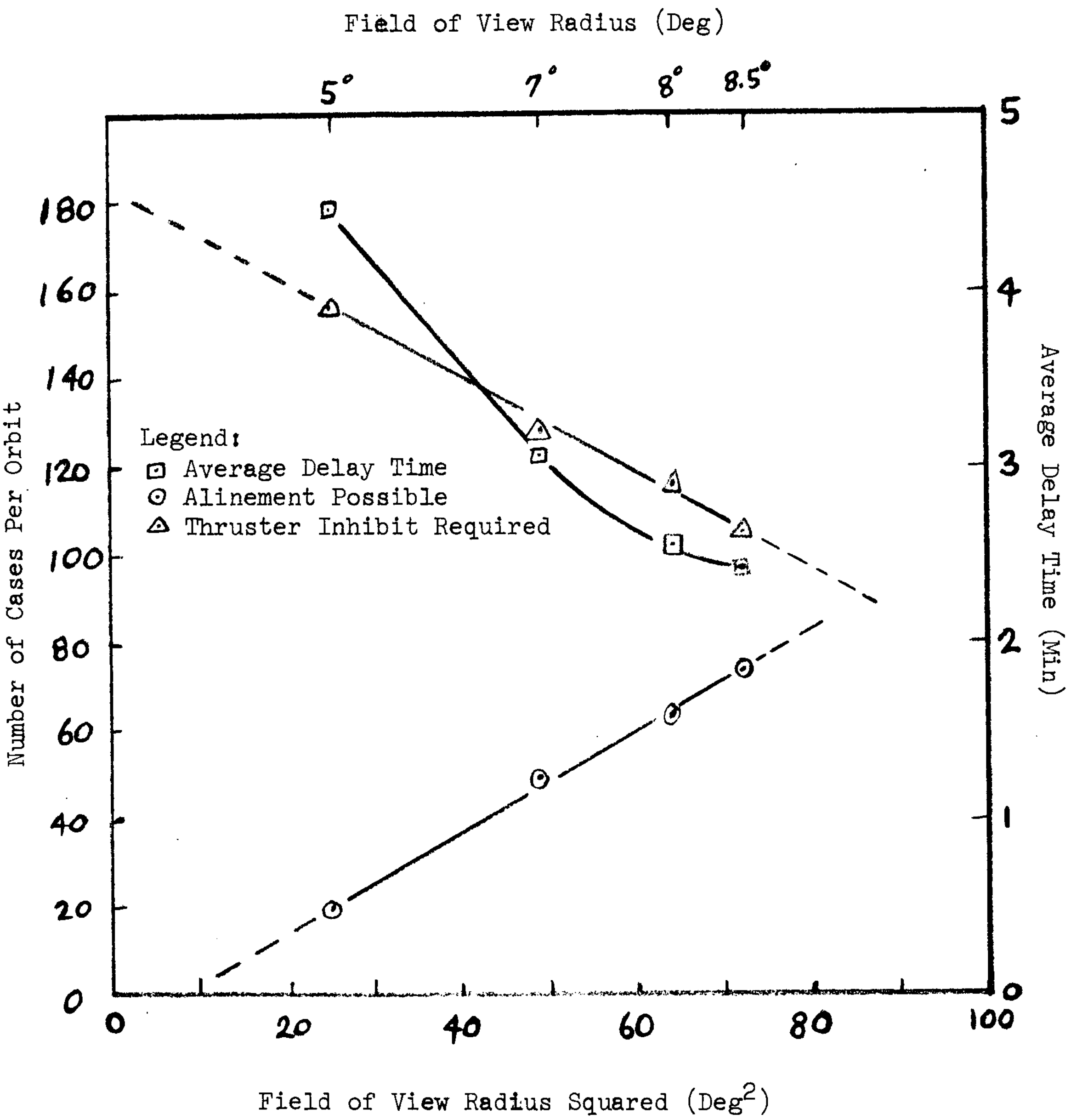


Figure B-7. Field of view size effects upon the capability to aline with 3 sensors separated by 120 degrees in the yaw plane. The orbit is perpendicular to the ecliptic.

9.7 ORBITAL POWERED FLIGHT

The following GN&C software functions are envisioned for the orbital phase:

Guidance Functions

1. Perform orbit modifications targeting, or accept ground/base targeting solutions as initialization.
2. Compute and command initial thrusting attitude.
3. Command orbiter maneuvering engine on.
4. Compute velocity to be gained vector (before and during burn).
5. Provide orbit maneuver steering commands to autopilot.
6. Compute time-to-cut-off and issue engine off commands.
7. Provide commands to null residual velocities.

Navigation Functions

1. Specific Force Integration - Advance the inertial state utilizing accelerometer measurement of thrust and aerodynamic forces.
2. Update inertial state from other navigation sensor data if available. An example is radar altimeter data.
3. Provide coordinate transformations for state vectors as required.
4. Compare state with that calculated by other vehicle during launch for use in decision making and possible updating.

Control Functions

1. Perform vehicle stabilization and control during TVC by engine gimbal commands.
2. Provide vehicle roll stabilization during single-engine burns using RCS.
3. Perform attitude-hold RCS ΔV maneuvers.
4. Perform steered-attitude RCS ΔV maneuvers for docking if required.

5. Do cg/trim estimation during TVC burns.

6. Make high-frequency steering estimates between guidance samples for docking if required.

7. Perform adaptive-loop gain calculation if required.

9.7.1 Required Velocity Determination

9.7.7.1 REQUIRED VELOCITY DETERMINATION, CONIC

SPACE SHUTTLE

GN&C SOFTWARE EQUATION SUBMITTAL

Software Equation Section: Required Velocity Determination (Conic)

Submittal No. 24A

Function: Solution of Multi-Revolution Lambert's and De-Orbit Problems

Module No.	OG5	Function No.	1,2 (MSC 03690)
	OG2		4
	OG3		1,2,3

Submitted byL W. M. Robertson Co. MIT No. 10 (Rev. 1)

Date: 21 October 1971

NASA Contract: J. Suddath Organization: EG2

Approved by Panel III: K.J. Cox Date: 10/21/71

Summary Description: Computes velocity vector required at an initial position to transfer through an inverse square central force field from the initial position (1) to a specified target position in a specified time interval or (2) to a specified (lower) target radius with a specified flight-path angle in a specified transfer time interval. Revision includes logic (1) to preclude difficulties for transfer angles near 180° and (2) to improve Secant Iterator.

Shuttle Configuration: This software is essentially independent of shuttle configuration.

Comments:

(Design Status) __________

(Verification Status) __________

Panel Comments:

Revision: A. Prior Submittal July 1971.

9.7.1.1 Conic (continued)

1. INTRODUCTION

The Conic Required Velocity Determination Routine provides the capability to solve the following two astrodynamic problems:

"The Multiple-Revolution Lambert Required Velocity Determination Problem": compute the velocity vector required at an initial position to transfer through an inverse square central force field from the initial position to a specified target position in a specified transfer time interval by making a specified number of complete revolutions (plus some fraction of another one). Also optionally compute the velocity vector at the target position and various parameters of the conic transfer orbit.

"The De-orbit Required Velocity Determination Problem": compute the velocity vector required at an initial position to transfer through an inverse square central force field from the initial position to a specified target radius (which is less than the initial radius) with a specified flight-path angle at that radius in a specified transfer time interval. Also optionally compute the velocity vector at the target position and various parameters of the conic transfer orbit.

NOMENCLATURE

a Semi-major axis of conic

c_1 First conic parameter $[\, c_1 = \sqrt{r_0 \, p_N} \; \Gamma_0 = (\underline{r}_0 \cdot \underline{v}_0)/\sqrt{\mu_E} \,]$

c_2 Second conic parameter $[\; c_2 = 1 - \alpha_N = r_0 \, v_0^2 / \mu_E - 1 \,]$

C or $C(\xi)$ Power series in ξ defined in the text

E Eccentric anomaly

f True anomaly

H Hyperbolic analog of eccentric anomaly

i Iteration counter

$\underline{i}_{c-}$ The negative unit chord vector connecting $\underline{r}_0$ and $\underline{r}_1$. $[\, \underline{i}_{c-} = -\text{unit} \; (\underline{r}_1 - \underline{r}_0)]$.

i_{max} Maximum allowable number of iterations

$\underline{i}_N$ Unit vector in direction of angular momentum vector of the transfer and normal to the transfer plane. In the Lambert Foutine the vector $\underline{i}_N$ always determines the direction of the transfer, and will also determine the plane of the transfer when either the switch $s_{proj} = 1$, or the switch $s_{proj} = 0$ but the initial position vector $\underline{r}_0$ is inside one of the cones. In the De-orbit Routine, the vector $\underline{i}_N$ always determines the plane and direction of the transfer.

$\underline{i}_{r_0}$ Unit vector in direction of $\underline{r}_0$

$\underline{i}_{r_1}$ Unit vector in direction of $\underline{r}_1$

9.7.1.1 Conic (continued)

k Intermediate variable equal to either k_{bg} or k_{sm}

k_{bg} Constant establishing by what fraction of its permissible range ($\Gamma_{max} - \Gamma_{min}$) the independent variable Γ_0 will be biased in the first iteration when no guess Γ_{guess} is available, in order to establish a second point for the secant iteration

k_{sm} Constant establishing by what fraction of its permissible range ($\Gamma_{max} - \Gamma_{min}$) the independent variable Γ_0 will be biased in the first iteration when a guess Γ_{guess} is available in order to establish a second point for the secant iteration.

m The slope of the line joining two successive points on the transfer time interval vs. independent variable curve.

m' Previous value of m

m_{err} Difference between desired value of the slope m (namely zero) and the value calculated on most recent iteration.

n Loop counter in the Marscher Equation Inversion

n_{rev} Integer number of complete 360° revolutions to be made in the desired transfer. [Hence the transfer will be between n_{rev} and $n_{rev} + 1$ revolutions].

$\underline{N}$ Intermediate vector variable normal to transfer plane

p Semi-latus rectum of conic

p_1 Intermediate variable in the Lambert problem equal to $1 - \cos\theta$

p_2 Intermediate variable in the Lambert problem equal to $\cos\theta - (r_0 / r_1)$

p_N Normalized semi-latus rectum of conic transfer orbit ($p_N = p / r_0$).

q Intermediate variable equal to $\lambda / \sin^2 \gamma_1$

9.7.1.1. Conic (continued)

$\underline{r}_0$ Initial or current inertial position vector (corresponds to time t_0).

$\underline{r}_1$ Terminal or target inertial position vector (corresponds to time t_1).

r_1 Radius at terminal or target position (corresponds to time t_1).

s Switch used in Secant Iterator to determine whether secant method or offsetting (biasing) will be performed.

s_{cone} Switch indicating whether the outcome of the cone test involving the tolerance criterion ϵ_{cone} was that initial position $\underline{r}_0$ lies outside both of the cones around the positive and negative target position vector $\underline{r}_1$ ($s_{cone} = 0$), or inside one of these cones ($s_{cone} = 1$). [See Section 4.7.]

s_{guess} Switch indicating whether the routine is to compute its own guess of the independent variable Γ_0 to start the iterative procedure ($s_{guess} = 0$), or is to use a guess Γ_{guess} supplied by the user ($s_{guess} = 1$)

s_{proj} Switch indicating whether the initial and target position vectors, $\underline{r}_0$ and $\underline{r}_1$, are to be projected into the plane defined by the unit normal $\underline{i}_N$ before the main Lambert computations are performed. If $s_{proj} = 0$, no projection will be made unless the initial position $\underline{r}_0$ is found to lie within one of the cones defined by ϵ_{cone}, in which case s_{cone} will be set equal to 1. If $s_{proj} = 1$, the projections will be carried out immediately, and no cone test will be made.

9.7.1.1 Conic (continued)

s_{soln} Switch indicating which of the two physically possible solutions is desired in the multi-revolution case. [Not used in the less-than-360° transfer case] . In particular, $s_{soln} = -1$ indicates the solution with the smaller initial flight path angle γ_0 measured from local vertical, and $s_{soln} = +1$ indicates the one with the larger γ_0.

s_{180} Switch indicating whether the central transfer angle is between 0° and 180° ($s_{180} = +1$), or between 180° and 360° ($s_{180} = -1$). The determination of which one of the above two possibilities is desired is made automatically by the routine on the basis of the direction of the unit normal vector $\underline{i}_N$.

[In the multiple-revolution case, the number of complete 360° revolutions is neglected; i.e., s_{180} is the sign of the sine of the transfer angle.]

S or $S(\xi)$ Power series in ξ defined in the text.

t_{err} Difference between specified time interval and that calculated by Universal Kepler Equation [$t_{err} = \Delta t - \Delta t_c$]

$\underline{v}_0$ Inertial velocity required at the initial position $\underline{r}_0$ to transfer to the terminal point in exactly the specified time interval Δt.

$\underline{v}_1$ Inertial velocity at the terminal position $\underline{r}_1$.

V_n ($n=1, 2..$) Intermediate scalar variables used in Marscher Equation Inversion

W_n ($n=1, 2..$) Intermediate scalar variables used in Marscher Equation Inversion

9.7.1.1 (Continued)

x Universal eccentric anomaly difference corresponding to the transfer from $\underline{r}_0$ to $\underline{r}_1$.

x_N Normalized universal eccentric anomaly difference ($x_N = x / \sqrt{r_0}$)

α_N Reciprocal of normalized semi-major axis of conic transfer orbit ($\alpha_N = r_0 / a$).

γ_0 Flight path angle at initial position $\underline{r}_0$ measured from local vertical, i. e., angle from $\underline{r}_0$ to $\underline{v}_0$.

γ_1 Flight-path angle at terminal or target position measured from local vertical (corresponds to time t_1).

Γ_0 Cotangent of flight-path angle γ_0 at the initial position $\underline{r}_0$ measured from local vertical; i. e., cotangent of the angle between $\underline{r}_0$ and $\underline{v}_0$. [Independent variable in iterative scheme] .

Γ_0' Previous value of Γ_0

$\Gamma_0^{(i)}$ The "i-th" value of Γ_0

Γ_1 Cotangent of flight path angle γ_1 at the terminal or target position $\underline{r}_1$ measured from local vertical

Γ_{guess} Guess of independent variable Γ_0 corresponding to solution (disregarded when $s_{guess} = 0$).

Γ_{parab} Value of Γ_0 corresponding to the physically realizable parabolic transfer

Γ_{max} Upper bound on Γ_0

Γ_{ME} Value of Γ_0 corresponding to the minimum energy transfer

9.7.1.1 Conic (continued)

Γ_{min} Lower bound on Γ_0

Δt Specified transfer time interval $(t_1 - t_0)$ between $\underline{r}_0$ and $\underline{r}_1$

Δt_c Value of the transfer time interval calculated in the Universal Kepler Equation from the current value of Γ_0 and the conic parameters

$\Delta t_c{}'$ Previous value of Δt_c

$\Delta t_c{}^{(i)}$ The "i-th" value of the transfer time interval calculated in the Universal Kepler Equation as a function of the "i-th" value $\Gamma_0{}^{(i)}$ of Γ_0 and the conic parameters

$\Delta\Gamma_0$ Increment in Γ_0

$\Delta\Lambda$ Increment in Λ

ϵ_{cone} Tolerance criterion establishing small cones around both the positive and negative target position directions inside of which the Lambert routine will define the plane of the transfer by the unit normal $\underline{i}_N$ rather than the cross product of the initial and target position vectors, $\underline{r}_0$ and $\underline{r}_1$. [ϵ_{cone} = sin (the half cone angle)].

ϵ_t Primary convergence criterion: relative error in transfer time interval

$\epsilon_t{}'$ Secondary convergence criterion: minimum permissible difference of two successive calculated transfer time intervals.

ϵ_T Convergence criterion in iteration to adjust Γ_{min} and Γ_{max} in multiple revolution case: absolute precision to which transfer time interval minimum is to be determined

ϵ_Γ Tertiary convergence criterion: minimum permissible size of increment $\Delta\Gamma_0$ of the independent variable

ϵ_{Λ} Tolerance criterion in iteration to adjust Γ_{min} and Γ_{max} in multiple revolution case: absolute difference of two successive values of independent variable to prevent division by zero

θ Transfer angle (true anomaly increment)

λ Ratio of initial position radius to terminal position radius

Λ Average of the two most recent values of Γ_0. [Λ is used as the independent variable in the Multi-revolution Bounds Adjustment Coding Sequence Iteration] .

Λ' Previous value of Λ

μ Gravitational parameter of the earth (product of earth's mass and universal gravitation constant)

ξ The dimensionless variable $\alpha x^2 = x^2 / a = \alpha_N x^2 / r_0$. . [Equivalent to square of standard eccentric or hyperbolic anomaly difference] .

2. FUNCTIONAL FLOW DIAGRAM

The Conic Required Velocity Determination Routine basically consists of two major parts—one for solving the multi-revolution Lambert's problem and one for solving the De-orbit problem—which are quite similar. In fact, certain subsections of the parts are identical as well as being identical to certain subsections of the Conic State Extrapolation Routine (Ref. 7) and these may of course be arranged as subroutines on a computer.

The Conic Lambert and De-orbit Required Velocity Determination Routines each involve a single loop iterative procedure, and hence are organized in three sections: initialization, iteration, and final computations, as shown in Figure 1. The independent variable in the iteration in both routines is the cotangent of the flight-path angle at the initial position measured from local vertical, or equivalently the cotangent of the angle between the initial position vector (extended) and the as yet unknown required velocity vector. The dependent variable is the transfer time interval; it is a function solely of the independent variable and certain other quantities which depend explicitly on the input and which are thus constant in any one problem. In the iterative procedure, the independent variable (denoted by Γ_0) is adjusted between upper and lower bounds by a secant technique until the transfer time interval computed from it agrees with the specified transfer time interval (to within a certain tolerance). Then the velocity vector at the initial position (i.e., the required velocity), as well as the velocity vector at the terminal position, is calculated from the last adjusted value of the independent variable.

In the less-than-one-complete revolution case in both routines, the upper and lower bounds on the independent variable are explicitly computed since the dependent and independent variables are monotonically related. However, in the multi-revolution case in the Lambert routine, there are two distinct physically-meaningful transfers which solve the problem, and an iterative procedure (entirely separate from, and not containing nor contained in the previously described iteration scheme) must be used to solve for the value of the independent variable which separates the two regions in each of which exactly one solution lies so that upper and lower bounds may be established corresponding to the unique solution desired. The multi-revolution case for the de-orbit problem is not considered in this document.

9.7.1.1 Conic (continued

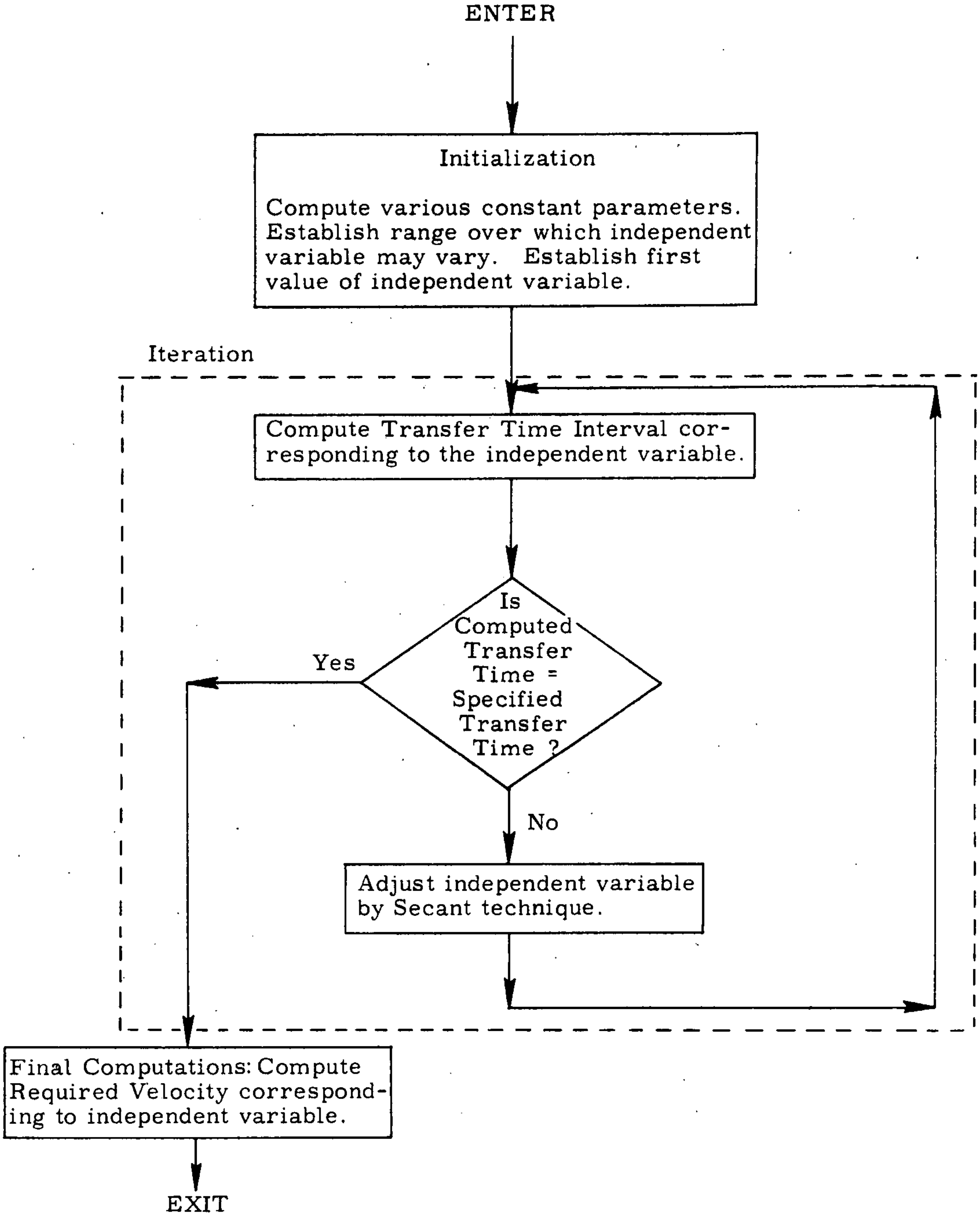

ENTER
Initialization
Compute various constant parameters. Establish range over which independent variable may vary. Establish first value of independent variable.
Iteration
Compute Transfer Time Interval corresponding to the independent variable.
Is Computed Transfer Time = Specified Transfer Time ?
Yes
No
Adjust independent variable by Secant technique.
Final Computations: Compute Required Velocity corresponding to independent variable.
EXIT

Figure 1. Conic Lambert and De-orbit Required Velocity Determination Routines Functional Flow Diagram

3. INPUT AND OUTPUT VARIABLES

The Conic Required Velocity Determination Routines have only one universal constant: the gravitational parameter of the earth. However, associated with each of the routines are a number of more or less fixed constants which are primarily convergence tolerance criteria. Since their values will be firmly established when the routines are coded into the Orbiter computer, they are not shown as input parameters below. The principal real-time input variables are the initial inertial position vector, the desired transfer time interval, and either the desired terminal position vector for the Lambert problem or the desired terminal radius and flight-path angle for the de-orbit problem. The principal real-time output of both routines is the inertial required velocity vector.

3.1 Lambert Required Velocity Determination

Input Variables

$\underline{r}_0$ Initial or current inertial position vector (corresponds to time t_0).

$\underline{r}_1$ Terminal or target inertial position vector (corresponds to time t_1).

Δt Transfer time interval $(t_1 - t_0)$ between $\underline{r}_0$ and $\underline{r}_1$.

n_R Integer number of complete 360° revolutions to be made in the desired transfer. [Hence the transfer will be between n_R and $n_R + 1$ revolutions] .

s_{soln} Switch indicating which of the two physically possible solutions is desired in the multi-revolution case. [Not used in the less-than-360° transfer case] . In particular, $s_{soln} = -1$ indicates the solution with the smaller initial flight path angle γ_0 measured from vertical, and $s_{soln} = +1$ indicates the one with the larger γ_0.

s_{guess} Switch indicating whether the routine is to compute its own guess of the independent variable Γ_0 to start the iterative procedure ($s_{guess} = 0$), or is to use a guess Γ_{guess} supplied by the user ($s_{guess} = 1$).

Γ_{guess} Guess of independent variable Γ_0 corresponding to solution (disregarded when $s_{guess} = 0$).

ϵ_{cone} Tolerance criterion establishing small cones around the positive and negative target position directions inside of which the routine will define the plane of the transfer by the unit normal vector $\underline{i}_N$, rather than the cross product of $\underline{r}_0$ and $\underline{r}_1$. The tolerance ϵ_{cone} is the sine of the half-cone angle.

s_{proj} Switch indicating whether the initial and target position vectors, $\underline{r}_0$ and $\underline{r}_1$, are to be projected into the plane defined by the unit normal $\underline{i}_N$ before the main Lambert computations are performed. If $s_{proj} = 0$, no projection will be made unless the initial position vector $\underline{r}_0$ is found to lie within one of the cones defined by ϵ_{cone}. If $s_{proj} = 1$, the projections are always made.

$\underline{i}_N$ Unit vector normal to transfer plane and in direction of angular momentum vector of transfer. The vector $\underline{i}_N$ always determines the direction of the transfer, and will also determine the plane of the transfer when either the switch $s_{proj} = 1$, or the switch $s_{proj} = 0$ but the initial position vector $\underline{r}_0$ is inside one of the cones.

9.7.1.1 Conic (continued)

Output Variables

$\underline{v}_0$ Inertial velocity required at the initial position $\underline{r}_0$ to transfer to the terminal position $\underline{r}_1$ in exactly the specified time interval Δt.

$\underline{v}_1$ Inertial velocity at the terminal position $\underline{r}_1$.

Γ_0 Value of independent variable to which the last iteration converged.

s_{cone} Switch indicating the outcome of the cone test. If $s_{cone} = 0$, the initial position $\underline{r}_0$ lies outside both of the cones around the positive and negative target position vectors. If $s_{cone} = 1$, it lies inside one of these cones.

$\underline{r}_1$ Terminal or target position vector actually used in computations. It is different from input $\underline{r}_1$ only if projection was performed. (See s_{proj} and s_{cone} switches.)

$\left.\begin{matrix}\sin\theta \\ \cos\theta\end{matrix}\right\}$ Sine and cosine of the transfer angle.

α_N Reciprocal of normalized semi-major axis of conic transfer orbit $(\alpha_N = r_0 / a)$.

p_N Normalized semi-latus rectum of conic transfer orbit $(p_N = p / r_0)$.

x Universal eccentric anomaly difference corresponding to the transfer from $\underline{r}_0$ to $\underline{r}_1$.

9.7.1.1 Conic (continued)

ξ The variable $\alpha x^2 = x^2 / a = \alpha_N x^2 / r_0$.

$S(\xi)$, $C(\xi)$ Values of the S and C functions corresponding to the transfer from $\underline{r}_0$ to $\underline{r}_1$. (Used by the Automatic Initialization Routine of the Relative State Updating Function to compute sensitivity matrices).

3. 2 De-orbit Required Velocity Determination

Input Variables

$\underline{r}_0$ Initial or current inertial position vector (corresponds to time t_0).

γ_1 Flight-path angle at terminal or target position measured from local vertical (corresponds to time t_1).

r_1 Radius at terminal or target position (corresponds to time t_1).

[NOTE: r_1 must be less than $|\underline{r}_0|$.]

Δt Transfer time interval $(t_1 - t_0)$ between initial and terminal positions.

s_{guess} Flag indicating whether the routine is to compute its own guess of the independent variable Γ_0 to start the iterative procedure ($s_{guess} = 0$), or is to use a guess Γ_{guess} supplied by the user ($s_{guess} = 1$).

Γ_{guess} Guess of independent variable Γ_0 corresponding to solution (disregarded when $s_{guess} = 0$).

$\underline{i}_N$ Unit vector which defines the plane and the direction of the transfer; it is normal to the transfer plane and in the direction of the angular momentum vector of the transfer.

Output Variables

$\underline{v}_0$ Inertial velocity required at the initial position $\underline{r}_0$ to transfer to the terminal flight-path angle and radius in exactly the specified transfer time.

$\underline{v}_1$ Inertial velocity at the terminal position.

Γ_0 Value of the independent variable to which last iteration converged.

$\underline{r}_1$ Inertial position vector at the terminal position.

$\left.\begin{matrix}\sin\ \theta \\ \cos\ \theta\end{matrix}\right\}$ Sine and cosine of transfer angle.

α_N Reciprocal of normalized semi-major axis of conic transfer orbit. ($\alpha_N = r_0 / a$)

p_N Normalized semi-latus rectum of conic transfer orbit ($p_N = p/r_0$)

x Universal eccentric anomaly difference corresponding to transfer from $\underline{r}_0$ to $\underline{r}_1$.

9.7.1.1 Conic (continued)

ξ The variable $\alpha x^2 = x^2/a = \alpha_N x^2/r_0$

$S(\xi)$, $C(\xi)$ Values of the S and C functions corresponding to the transfer from $\underline{r}_0$ to $\underline{r}_1$. (May be used by other routines to evaluate sensitivity matrices.)

4. DESCRIPTION OF EQUATIONS

The following description applies to both the Lambert and De-orbit problems unless otherwise noted.

4.1 Preliminary Comments

A combination of the Marscher Equation Inversion and the Stumpff-Herrick-Battin Universal Kepler Equation is used in this formulation of the solutions to the Lambert and De-orbit problems. This is the particularly convenient when used in conjunction with the Kepler and Theta problem solutions described previously (Ref. 7) as those problems utilize the same two equations, which may hence be coded for a computer as common subroutines.

The independent variable in the iterative solution in both the Lambert and De-orbit routines is the cotangent of the flight-path angle at the initial position measured from local vertical, and this variable is denoted by Γ_0. Thus Γ_0 = cot γ_0 with γ_0 being the angle from the initial position vector $\underline{r}_0$ (extended) to the as yet unknown required velocity $\underline{v}_0$. A guess of the independent variable Γ_0 is transformed by the Marscher Equation Inversion into a corresponding value of the universal eccentric anomaly difference x, from which the corresponding transfer time interval Δt_c between the initial and terminal positions is evaluated by the Universal Kepler Equation. The subscript "c" on the dependent variable Δt_c indicates the calculated transfer time interval determined from some value of Γ_0, as opposed to the desired transfer time interval Δt specified in the input. The universal eccentric anomaly difference x may be defined by the relations:

$$x = \begin{cases} \sqrt{a}\ (E - E_0) \text{ for ellipse} \\ \sqrt{p}\ (\tan f/2 - \tan f_0/2) \text{ for parabola} \\ \sqrt{-a}\ (H - H_0) \text{ for hyperbola} \end{cases}$$

where a is the semi-major axis, E and H are the eccentric anomaly and its hyperbolic analog, p the semi-latus rectum and f the true anomaly.

4.2 Computation of the Conic Parameters

In order to evaluate the transfer time interval Δt_c for a given value of the independent variable Γ_0, it is first necessary to obtain the intermediate parameters α_N and p_N.

In the Lambert problem case, the normalized semi-latus rectum $p_N = p / r_0$ and the normalized semi-major axis reciprocal $\alpha_N = r_0 \alpha = r_0 / a$ are determined from the central transfer angle θ, the initial and terminal radii r_0 and r_1 (these three parameters being constant throughout any one problem), and from the value Γ_0 of the independent variable by the equations:

$$p_N = \frac{1 - \cos \theta}{\Gamma_0 \sin \theta - (\cos \theta - r_0 / r_1)}$$

$$\alpha_N = 2 - p_N (1 + \Gamma_0^2)$$

In the De-orbit problem case, the central transfer angle is not known directly from the input (it depends on the current value of Γ_0). The parameters p_N and α_N are instead computed from the desired terminal flight path angle γ_1, and the initial and terminal radii r_0 and r_1 (these three parameters being constant throughout any one problem) and from the value Γ_0 of the independent variable by the equations:

$$p_N = \frac{2 (\lambda - 1)}{q \lambda - (1 + \Gamma_0^2)}$$

$$\alpha_N = 2 - p_N (1 + \Gamma_0^2)$$

where

$$\lambda = r_0 / r_1 \text{ and } q = \lambda / \sin^2 \gamma_1 .$$

4.3 Computation of the Transfer Time Interval and Required Velocity Vector

When the conic parameters p_N and α_N are known for a particular value Γ_0 of the independent variable in either the Lambert or De-orbit problem, the universal eccentric anomaly difference x is obtained from the Marscher Equation Inversion:

Let

$$W_1 = \sqrt{p_N} \left(\frac{\sin \theta}{1 - \cos \theta} - \Gamma_0 \right)$$

If

$$|W_1| > 1, \text{ let } V_1 = 1.$$

Let

$$W_{n+1} = + \sqrt{W_n^2 + \alpha_N} + |W_n| \qquad (|W_1| \leq 1)$$

or

or

$$V_{n+1} = + \sqrt{V_n^2 + \alpha_N (|1/W_1|)^2} + V_n \qquad (|W_1| > 1).$$

Let

$$\omega_n = W_n \qquad (|W_1| \leq 1)$$

or

$$1/\omega_n = (|1/W_1|)/V_n \qquad (|W_1| > 1).$$

Let

$$\Psi = \frac{2^n}{\omega_n} \sum_{j=0}^{\infty} \frac{(-1)^j}{2j+1} \left(\frac{\alpha_N}{\omega_n^2} \right)^j$$

where n is an integer ≥ 4. Then

$$x/\sqrt{r_0} = \begin{cases} \Psi & (W_1 > 0) \\ 2\pi/\sqrt{\alpha_N} - \Psi & (W_1 < 0) \end{cases}$$

The above equations have been specifically formulated to avoid certain numerical difficulties.

Finally, the transfer time interval Δt_c is evaluated by the Universal Kepler Equation:

$$(t_1 - t_0)_c = \Delta t_c = [c_1 x^2 C(\xi) + x(c_2 x^2 S(\xi) + r_0)]/\sqrt{\mu_E}$$

where

$$c_1 = \sqrt{r_0 p_N} \, \Gamma_0$$

$$c_2 = 1 - \alpha_N$$

$$\xi = \alpha_N x^2 / r_0$$

and

$$C(\xi) = \frac{1}{2!} - \frac{\xi}{4!} + \frac{\xi^2}{6!} - \dots$$

$$S(\xi) = \frac{1}{3!} - \frac{\xi}{5!} + \frac{\xi^2}{7!} - \dots$$

Since the transfer time interval Δt is specified, it is desired to find the Γ_0 corresponding to it through the above equations, and then to evaluate the required velocity vector using the expression

$$\underline{v}_0 = \sqrt{\frac{p_N \mu}{r_0}} \left[\Gamma_0 \, \underline{i}_{r_0} + (\underline{i}_N \times \underline{i}_{r_0}) \right],$$

where $\underline{i}_{r_0}$ and $\underline{i}_N$ are unit vectors in the directions of $\underline{r}_0$ and the angular momentum vector respectively.

4.4 Iteration Method and Independent Variable Bounds

Unfortunately, the combination of the Marscher Equation Inverstion and the Universal Kepler Equation expresses the transfer time interval Δt_c as a transcendental function of Γ_0 rather than conversely, and no power series inversion of the relationship is known which has good convergence properties for all orbits, so it is necessary to solve the relationship iteratively for the independent variable Γ_0.

For this purpose, the secant method (linear inverse interpolation / extrapolation) is used. It merely finds the increment $\Delta\Gamma_0$ in the independent variable Γ_0 which is required in order to adjust the dependent variable Δt_c to the desired value Δt based on a linear interpolation / extrapolation of the last two points calculated on the Δt_c vs Γ_0 curve. The method uses the formula

$$\Delta\Gamma_0 = \left(\Gamma_0^{(n+1)} - \Gamma_0^{(n)} \right) = - \frac{\Delta t_c^{(n)} - \Delta t}{\Delta t_c^{(n)} - \Delta t_c^{(n-1)}} \left(\Gamma_0^{(n)} - \Gamma_0^{(n-1)} \right),$$

where $\Delta t_c^{(n)}$ denotes the evaluation of the Marscher Equation Inversion followed by the Universal Kepler Equation using the nth value $\Gamma_0^{(n)}$ of the independent variable.

In order to prevent the scheme from starting in or iterating into regions in which it is known on theoretical grounds that no physically valid solution can occur, it is necessary to establish a priori upper and lower bounds on the independent variable Γ_0. The bounds are also useful in preventing the taking of an increment back into regions in which past iterations have shown that the solution does not lie; this is accomplished by continually resetting the bounds during the course of the iteration as more and more values of Γ_0 are found to be too large or too small. In addition, it has also been found expedient to damp by 10% away from the current bound any increment in the independent variable which would (if applied) take the value of the independent variable past this bound. Furthermore, in the multiple-revolution Lambert case, the bounds are indispensable in constraining the problem to the desired one of the two physically possible solutions.

To start the iteration scheme, two successive initial guesses, $\Gamma_0^{(0)}$ and $\Gamma_0^{(1)}$, of the independent variable are required. In the lack of other information, $\Gamma_0^{(0)}$ may be taken as the midpoint of the interval between the bounds over which Γ_0 may vary, and $\Gamma_0^{(1)}$ as a point biased away from $\Gamma_0^{(0)}$ by the relatively large fraction k_{bg} (perhaps 1/4) of that interval. This procedure will automatically be performed by the routines described in the detailed flow diagram section when the switch $s_{guess} = 0$. However, if some relatively good guess of the independent variable is available, such as a linear extrapolation of the values of the independent variable to which the last two calls of the Lambert or De-orbit routine had converged during powered flight guidance, then this guess should obviously be used as $\Gamma_0^{(0)}$, and $\Gamma_0^{(1)}$ should be a point biased away from $\Gamma_0^{(0)}$ by only a relatively small fraction k_{sm} (perhaps 1/10000) of the interval between the bounds. This procedure will automatically be carried out by the diagrammed routines when the switch $s_{guess} = 1$.

The iteration continues until the calculated transfer time interval has been driven to within the relative error ϵ_t of the desired transfer time interval,or until two successive calculated transfer time intervals differ by less than ϵ_t', or until the maximum number i_{max} of iterations has been reached, or until the increment $\Delta\Gamma_0$ in the independent variable is less than a certain minimum value ϵ_Γ.

4. 5 Computation of the Bounds in the Lambert Problem

In the less-than-one-complete-revolution case in the Lambert problem, the lower and upper bounds on the independent variable are computed from:

$$\Gamma_{min} = \begin{cases} (\cos\theta - r_0/r_1)/\sin\theta & (0<\theta<180^\circ) \\ -\infty & (180^\circ<\theta<360^\circ) \end{cases}$$

$$\Gamma_{max} = \left\{ \frac{\sin\theta}{1-\cos\theta} + \sqrt{\frac{2(r_0/r_1)}{1-\cos\theta}} \right. \qquad (0<\theta<360^\circ)$$

These equations result from the constraints of having finite semi-latus rectums (Γ_{min}) and not transferring "through infinity" (Γ_{max}); the transfer time interval Δt_c is zero at Γ_{min} and infinite at Γ_{max}.

9.7.1.1 Conic (continued)

In the multiple-revolution Lambert case, the bounds must be adjusted to reflect the fact that such transfers can only occur on elliptic orbits. The upper bound Γ_{max} already corresponds to the "longer" of the two possible parabolic transfers, while the value of the independent variable corresponding to the "shorter" (and physically realizable) one is:

$$\Gamma_{parab} = \frac{\sin\theta}{1-\cos\theta} - \sqrt{\frac{2(r_0/r_1)}{1-\cos\theta}}$$

Hence the independent variable may not vary outside the interval (max (Γ_{min}, Γ_{parab}), Γ_{max}) for the multiple revolution case. Moreover, in this case, there are two physically possible transfers having the same transfer time interval but different values of the independent variable. A graph of the typical functional relationship is shown in Figure 2.

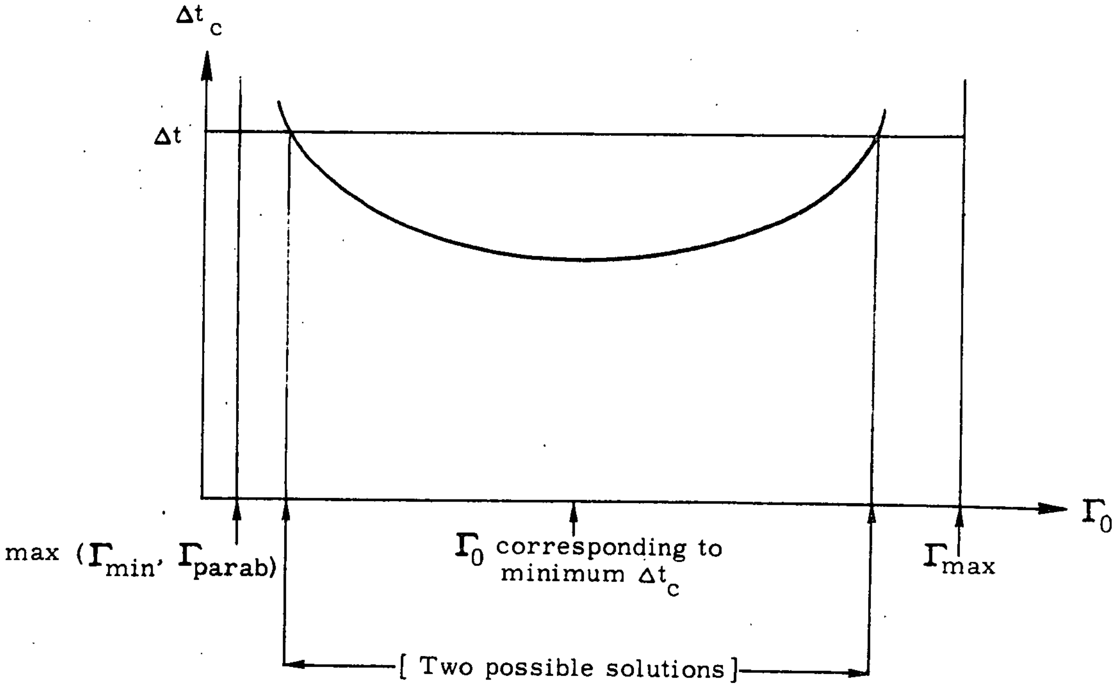


Figure 2. Typical Relationship Between Dependent and Independent Variables in Multiple Revolution Lambert Problem

To distinguish between the two possible solutions, it is necessary to find the value of Γ_0 corresponding to the minimum of Δt_c. This is accomplished by a secant iteration on the secant, i. e., by a secant iteration on the numerical approximation to the first derivative of the curve. Obviously, it is now desired to drive the value of the first derivative to zero. A good starting point for the iteration is furnished by the value of the independent variable corresponding to the minimum energy transfer:

$$\Gamma_{ME} = s_{180} \left| \underline{i}_{r_0} \times \underline{i}_{c-} \right| / (1 + \underline{i}_{r_0} \cdot \underline{i}_{c-})$$

where

$\underline{i}_{r_0}$ is the unit initial position vector,

$\underline{i}_{c-}$ is the negative unit chord vector connecting the initial and terminal points (the chord vector goes from the initial to the terminal point),

and

s_{180} is the sign of the sine of the transfer angle.

Once the value of Γ_0 corresponding to a minimum of Δt_c has been found, it serves to separate the interval into two subintervals in each of which exactly one solution lies, and hence the minimum point is taken as a further upper or lower bound depending on which solution is desired in order to constrain the variation of the independent variable to one of the two subintervals. In one of the subintervals the biasing constant k must be reversed in sign since the dependent variable there is a monotonically decreasing function of the independent variable, rather than a monotonically increasing one.

4.6 Computation of the Bounds in the De-orbit Problem

Only the less-than-one-complete-revolution case has been considered for the De-orbit problem in this document. The lower and upper bounds on the independent variable for this case are computed from:

$$\Gamma_{min} = -\sqrt{(1 + \cot^2 \gamma_1)(r_0 / r_1)^2 - 1}$$

$$\Gamma_{max} = +\sqrt{(1 + \cot^2 \gamma_1)(r_0 / r_1) - 1}$$

9.7.1.1 Conic (continued)

The transfer time interval Δt_c is zero at Γ_{min} and infinite at Γ_{max}. These equations are equivalent to

$$\Gamma_{min} = - \sqrt{q \lambda - 1}$$

$$\Gamma_{max} = + \sqrt{q - 1}$$

where

$$\lambda = r_0 / r_1 \text{ and } q = \lambda / \sin^2 \gamma_1 \quad \text{as before.}$$

It is to be noted that these bounds are based on the assumption that the terminal radius is less than the initial radius ($\lambda = r_0 / r_1 > 1$). When the converse is true, a different and rather more complex set of bounds is valid instead.

4.7 Treatment of the 180° Transfer Singularity in the Lambert Problem

For transfers of exactly 180° (or 540°, 900°, etc.), Lambert's problem has a partial physical singularity in that the plane of the transfer becomes indeterminate although the other orbit parameters,such as flight path angle and required velocity magnitude,are well-determined by the specification of the desired transfer time interval. A transfer in any plane with the correct other parameters will solve the problem mathematically and physically. In actual computer solutions to Lambert's problem, however, the singularity will arise not only at exactly the 180° transfer but also everywhere within a small neighborhood of the 180° direction, due to the fact that the computer, whether fixed or floating point, has a finite word length and cannot carry out the cross-product of the initial and target vectors with infinite precision. The small neighborhood, inside of which the transfer plane must be defined by other means, may be conceived of as a cone with apex at the origin and with axis along the negative target position direction. In fact, since the singularity also occurs in the vicinity of 0° (360°, 720°, etc.) transfers, two cones may be established point to point along both the negative and positive directions. The sine of the half-cone angle of these cones has been denoted by ϵ_{cone} in this document.

9.7.1.1 Conic (continued)

For transfers in the vicinity of integral multiples of 180° but for which the initial position vector $\underline{r}_0$ is outside the singularity cones described above, the plane in which the solution of Lambert's problem lies is highly sensitive both mathematically and physically to the input initial and target position directions, due to the near-colinearity of these vectors. The computer will be able to calculate the plane of the solution in these high sensitivity regions since $\underline{r}_0$ is not inside the singularity cones. However, in these regions slight movements of either of the input position vectors can cause the plane spanned by them, and hence the required velocity plane, to drastically change its orientation — for example, from an around-the-equator to an over-the-pole direction. This could have great consequences during the operation of various functions such as powered-flight guidance. Thus, in order to avoid system operational difficulties, it is important to take into account the high sensitivity regions, inside of which the transfer plane should be defined by other means such as an input unit normal, even though the Lambert problem solution is non-singular there.

The Lambert Routine diagrammed in the next section has been designed with provision for both the singularity cones and the sensitivity regions. As the angular shape of the cones is well determined, the routine will (1) test to decide whether the initial position vector $\underline{r}_0$ is outside or inside the cones based on the angular size given by ϵ_{cone}, (2) set the cone switch s_{cone} to 0 or 1 respectively to indicate the result to the user, and (3) utilize an input unit normal vector $\underline{i}_N$ to define the transfer plane in the inside-the-cone case. On the other hand, as the angular shape and size of the sensitivity regions may vary according to each system function which utilizes a Lambert problem solution, the Lambert routine requires the setting of an input switch s_{proj} to indicate whether the initial position vector $\underline{r}_0$ is to be considered as outside or inside a sensitivity region.* In the inside-the-region case, the routine will project the input initial and target

*The rationale for the choice and the detailed flow diagrams precisely defining it are given in the document describing the particular function. For the targeting and powered-flight guidance functions, they are given in the Precision Required Velocity Determination document (Ref. 2).

position vectors $\underline{r}_0$ and $\underline{r}_1$ into the plane defined by the input unit normal vector $\underline{i}_N$ before any main computations are performed, and will utilize the normal to determine the transfer plane. In the outside-the-sensitivity-region case, the routine will perform the singularity cone test previously described as a double-check on the setting of the projection switch s_{proj}, and if the initial position $\underline{r}_0$ is not inside the cone the unit normal to the transfer plane will be calculated internally from the cross product of the initial and target vectors. Should the cone test find the initial position inside the cone, the cone switch s_{cone} will be set to 1 as a warning, and the routine will override the zero setting of the projection switch and project the initial and target vectors into the plane defined by the input unit normal $\underline{i}_N$ as in the inside-the-sensitivity-region case above.

It should be pointed out that the unit normal $\underline{i}_N$ must always be input and must always be in the direction of the desired angular momentum vector of the transfer. In the outside-the-sensitivity-region case exactly one bit of information is extracted from $\underline{i}_N$, namely the polarity of the desired angular momentum, or whether the transfer is clockwise or counterclockwise when viewed from the tip of the cross-product vector of the initial and target positions.* In the inside-the-sensitivity-region or inside-the-cone-cases, the input unit normal provides orientation as well as polarity.

It should also be noted that a first approximation to a sensitivity region is a cone of larger central angle centered around the singularity cone. As the parameter ϵ_{cone} is an input variable, the Lambert routine diagrammed in the next section may be used to determine whether the initial position vector $\underline{r}_0$ lies inside this larger cone-shaped approximation to a sensitivity region by suitably adjusting ϵ_{cone}.This technique is utilized by the Precision Required Velocity Determination Routine.

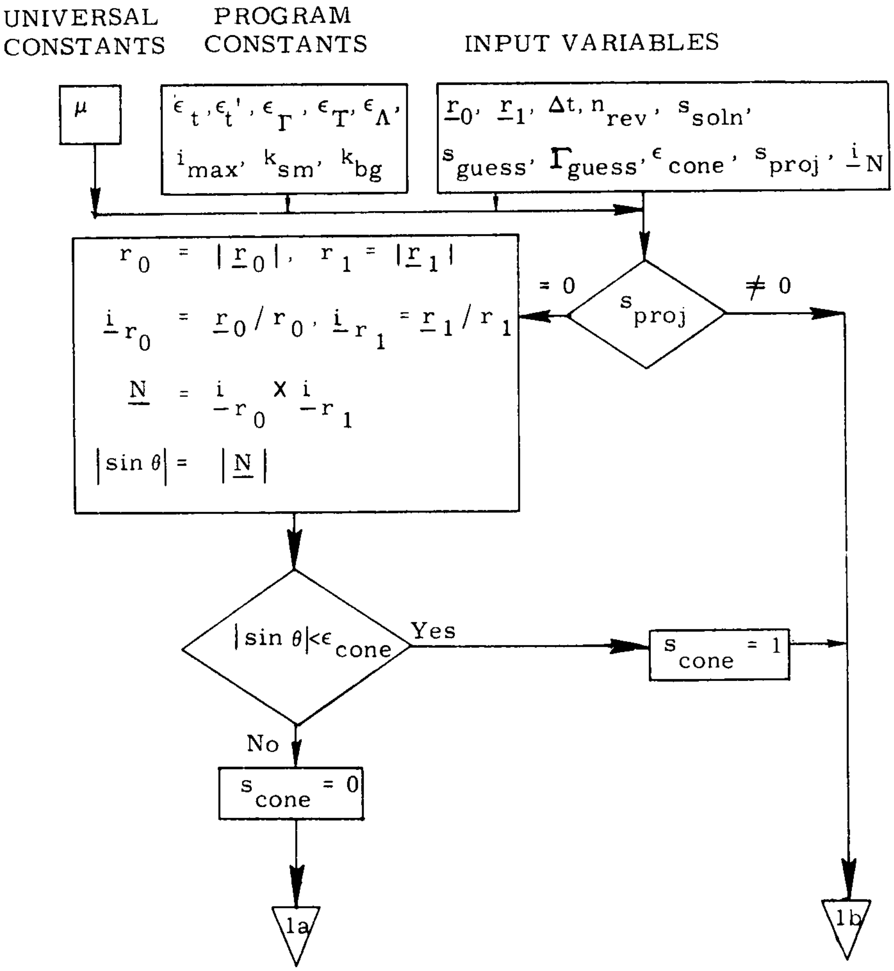
* The reader is reminded that the cross product operation always orients its result as if a right hand rotation were made through the smaller of the two angles from the initial to the target vector which lie in the plane of these vectors.

5. DETAILED FLOW DIAGRAMS

5.1 Multiple-Revolution Lambert Required Velocity Determination Routine

This routine utilizes the following subroutines or coding sequences, which are diagrammed in Section 5.3:

- Lambert Transfer Time Interval Subroutine
  - Marscher Equation Inversion Subroutine
  - Universal Kepler Equation Subroutine
- Secant Iterator
- Multi-revolution Bounds Adjustment Coding Sequence
  - Secant Minimum Iterator



UNIVERSAL CONSTANTS
PROGRAM CONSTANTS
INPUT VARIABLES
μ
$\epsilon_t, \epsilon_t', \epsilon_\Gamma, \epsilon_T, \epsilon_\Lambda, i_{max}, k_{sm}, k_{bg}$
$\underline{r}_0, \underline{r}_1, \Delta t, n_{rev}, s_{soln}, s_{guess}, \Gamma_{guess}, \epsilon_{cone}, s_{proj}, \underline{i}_N$
$r_0 = |\underline{r}_0|, r_1 = |\underline{r}_1|$
$\underline{i}_{r_0} = \underline{r}_0 / r_0, \underline{i}_{r_1} = \underline{r}_1 / r_1$
$\underline{N} = \underline{i}_{r_0} \times \underline{i}_{r_1}$
$|\sin\theta| = |\underline{N}|$
s_{proj}
= 0
≠ 0
$|\sin\theta| < \epsilon_{cone}$
Yes
No
$s_{cone} = 1$
$s_{cone} = 0$
1a
1b

Figure 3a. Multi-Revolution Lambert Routine Detailed Flow Diagram

9.7.1.1 Conic (continued)

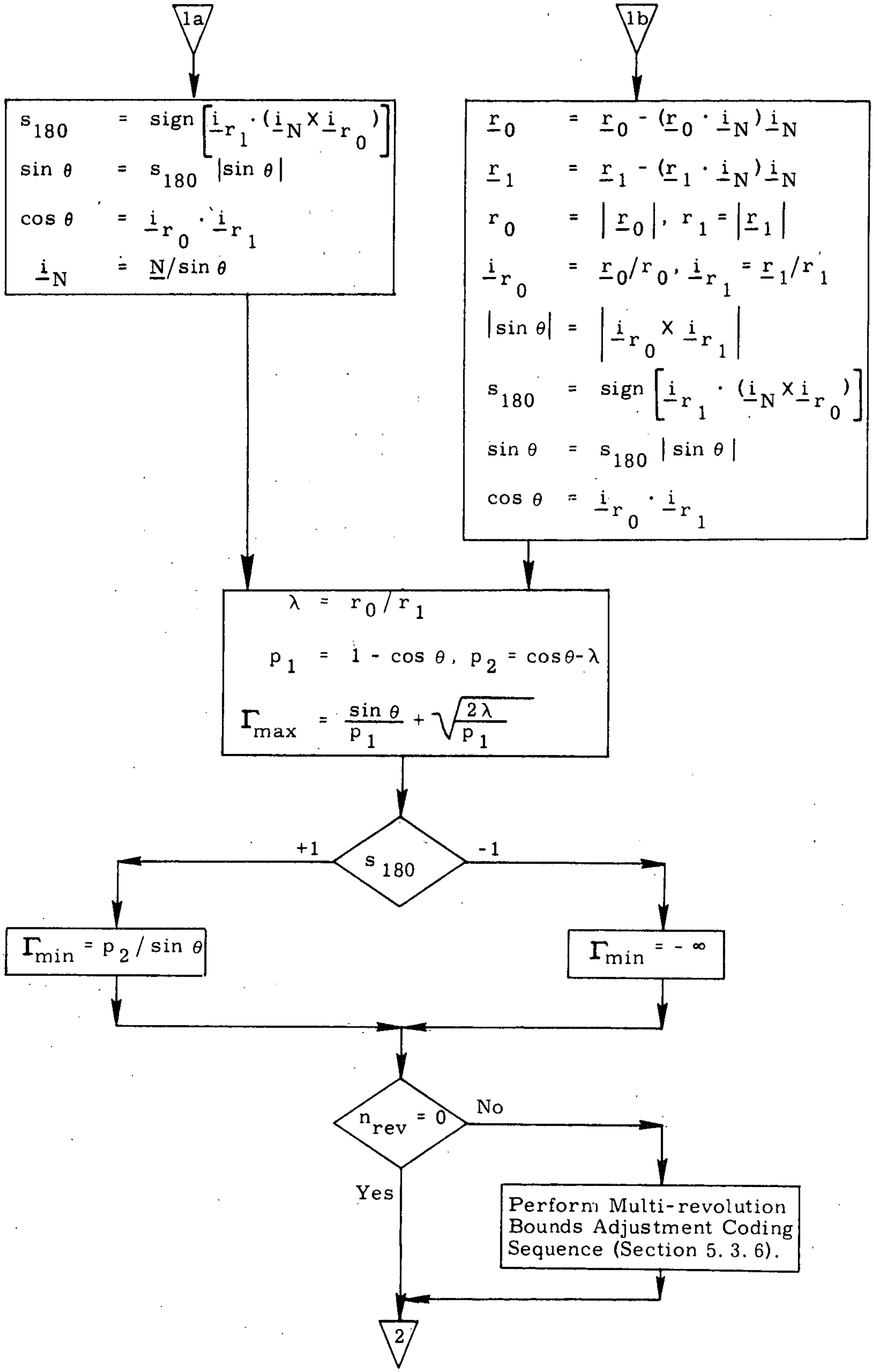

1a
1b
$s_{180} = \text{sign}\left[\underline{i}_{r_1} \cdot (\underline{i}_N \times \underline{i}_{r_0})\right]$
$\sin\theta = s_{180}\,|\sin\theta|$
$\cos\theta = \underline{i}_{r_0} \cdot \underline{i}_{r_1}$
$\underline{i}_N = \underline{N}/\sin\theta$
$\underline{r}_0 = \underline{r}_0 - (\underline{r}_0 \cdot \underline{i}_N)\underline{i}_N$
$\underline{r}_1 = \underline{r}_1 - (\underline{r}_1 \cdot \underline{i}_N)\underline{i}_N$
$r_0 = |\underline{r}_0|,\ r_1 = |\underline{r}_1|$
$\underline{i}_{r_0} = \underline{r}_0/r_0,\ \underline{i}_{r_1} = \underline{r}_1/r_1$
$|\sin\theta| = |\underline{i}_{r_0} \times \underline{i}_{r_1}|$
$s_{180} = \text{sign}\left[\underline{i}_{r_1} \cdot (\underline{i}_N \times \underline{i}_{r_0})\right]$
$\sin\theta = s_{180}\,|\sin\theta|$
$\cos\theta = \underline{i}_{r_0} \cdot \underline{i}_{r_1}$
$\lambda = r_0/r_1$
$p_1 = 1 - \cos\theta,\ p_2 = \cos\theta - \lambda$
$\Gamma_{max} = \frac{\sin\theta}{p_1} + \sqrt{\frac{2\lambda}{p_1}}$
+1
s_{180}
-1
$\Gamma_{min} = p_2/\sin\theta$
$\Gamma_{min} = -\infty$
$n_{rev} = 0$
No
Yes
Perform Multi-revolution Bounds Adjustment Coding Sequence (Section 5.3.6).
2

Figure 3b. Multi-Revolution Lambert Routine Detailed Flow Diagram

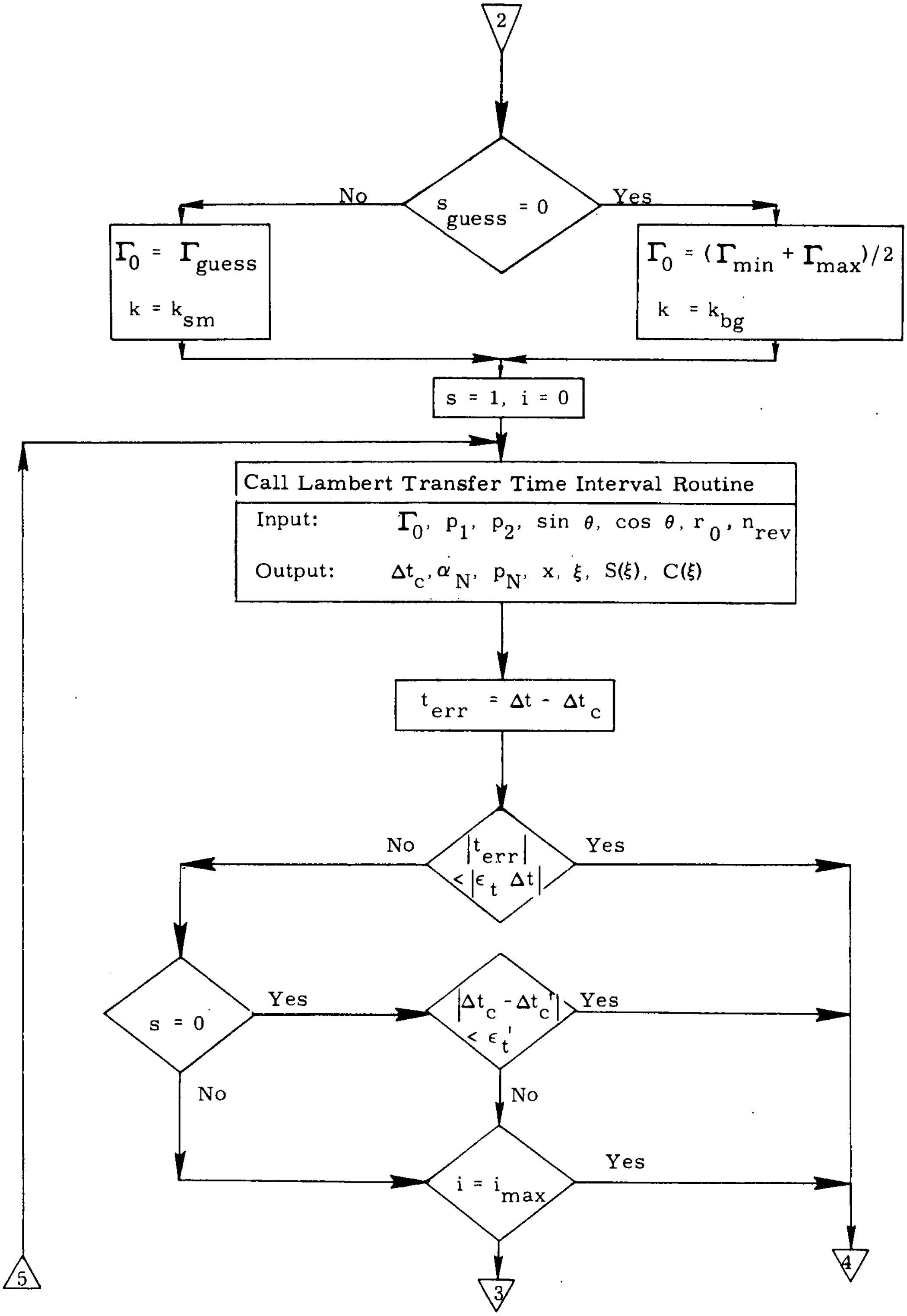

2
$s_{guess} = 0$
No
Yes
$\Gamma_0 = \Gamma_{guess}$
$k = k_{sm}$
$\Gamma_0 = (\Gamma_{min} + \Gamma_{max})/2$
$k = k_{bg}$
s = 1, i = 0
Call Lambert Transfer Time Interval Routine
Input: Γ_0, p_1, p_2, sin θ, cos θ, r_0, n_{rev}
Output: Δt_c, α_N, p_N, x, ξ, S(ξ), C(ξ)
$t_{err} = \Delta t - \Delta t_c$
$|t_{err}| < |\epsilon_t \Delta t|$
No
Yes
s = 0
Yes
No
$|\Delta t_c - \Delta t_c'| < \epsilon_t'$
Yes
No
$i = i_{max}$
Yes
5
3
4

Figure 3c. Multi-Revolution Lambert Routine Detailed Flow Diagram

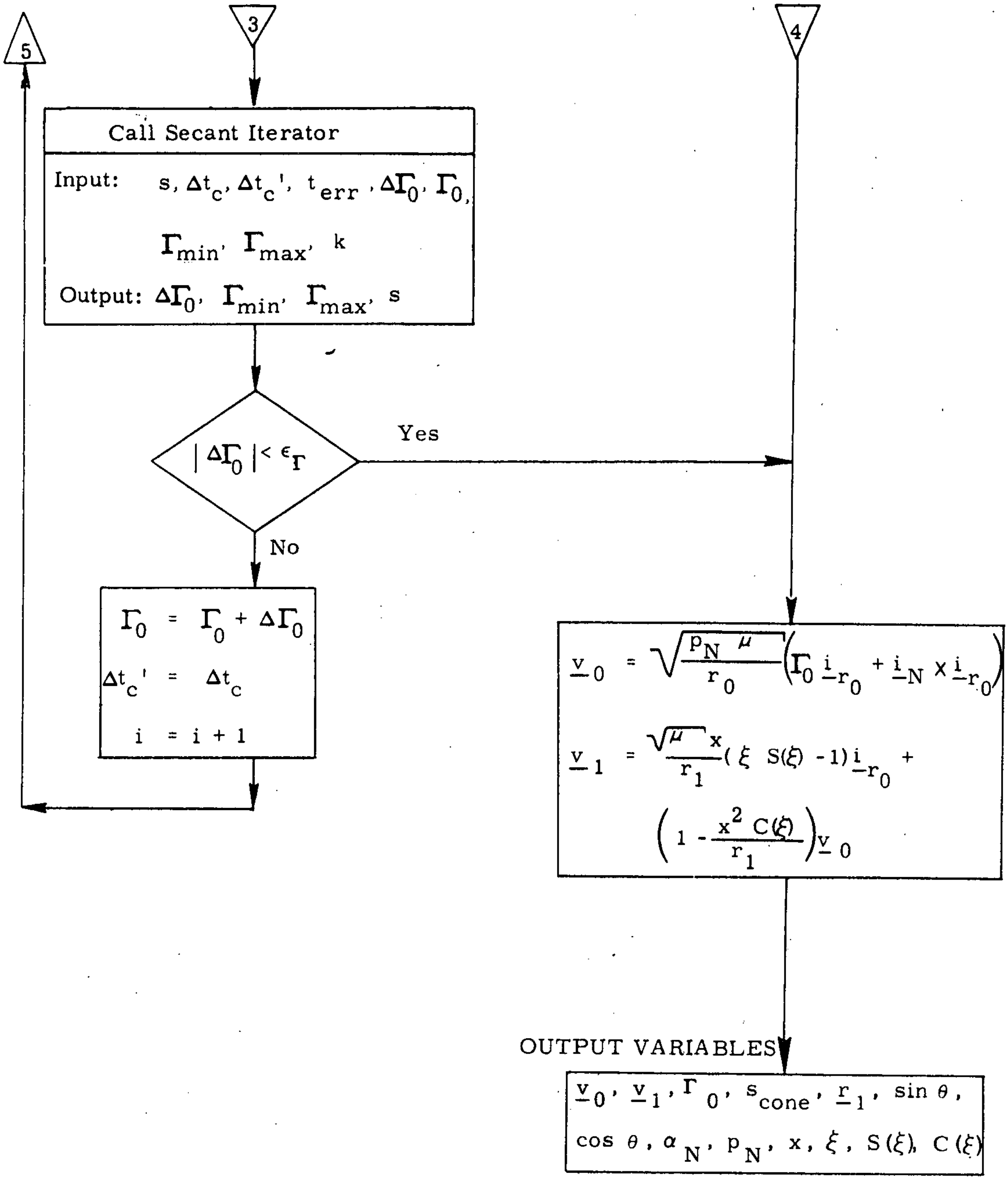

5
3
4
Call Secant Iterator
Input: $s, \Delta t_c, \Delta t_c', t_{err}, \Delta\Gamma_0, \Gamma_0, \Gamma_{min}, \Gamma_{max}, k$
Output: $\Delta\Gamma_0, \Gamma_{min}, \Gamma_{max}, s$
$|\Delta\Gamma_0| < \epsilon_\Gamma$
Yes
No
$\Gamma_0 = \Gamma_0 + \Delta\Gamma_0$
$\Delta t_c' = \Delta t_c$
$i = i + 1$
$\underline{v}_0 = \sqrt{\frac{p_N \mu}{r_0}} \left(\Gamma_0 \underline{i}_{r_0} + \underline{i}_N \times \underline{i}_{r_0} \right)$
$\underline{v}_1 = \frac{\sqrt{\mu} x}{r_1} (\xi S(\xi) - 1) \underline{i}_{r_0} + \left(1 - \frac{x^2 C(\xi)}{r_1} \right) \underline{v}_0$
OUTPUT VARIABLES
$\underline{v}_0, \underline{v}_1, \Gamma_0, s_{cone}, \underline{r}_1, \sin\theta, \cos\theta, \alpha_N, p_N, x, \xi, S(\xi), C(\xi)$

Figure 3d. Multi-Revolution Lambert Routine Detailed Flow Diagram

5.2 De-orbit Required Velocity Determination Routine

This routine utilizes the following subroutines which are diagrammed in Section 5.3:

- De-orbit Transfer Time Interval Subroutine
  - Marscher Equation Inversion Subroutine
  - Universal Kepler Equation Subroutine
- Secant Iterator

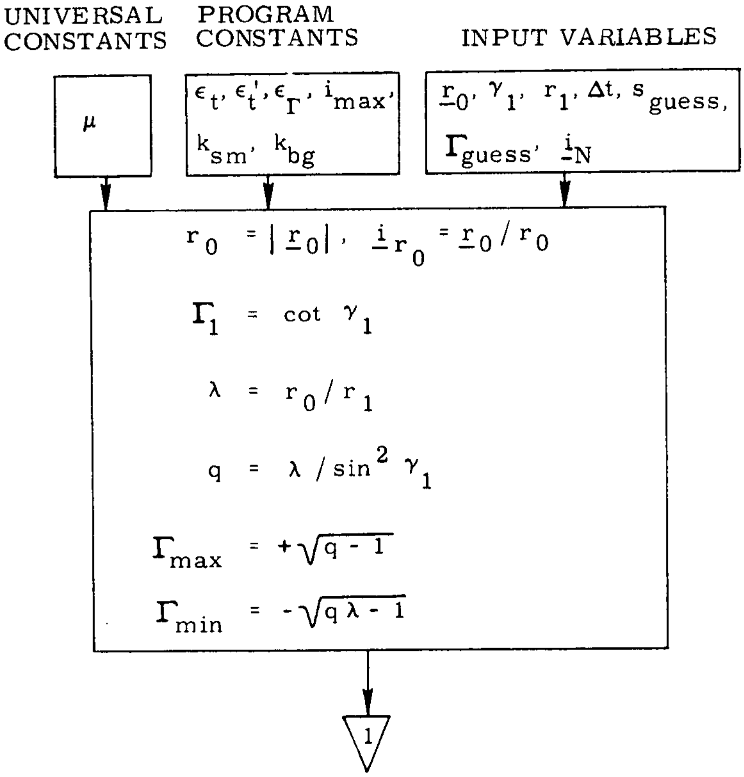

UNIVERSAL CONSTANTS
PROGRAM CONSTANTS
INPUT VARIABLES
μ
$\epsilon_t, \epsilon_t', \epsilon_\Gamma, i_{max}, k_{sm}, k_{bg}$
$\underline{r}_0, \gamma_1, r_1, \Delta t, s_{guess}, \Gamma_{guess}, \underline{i}_N$
$r_0 = |\underline{r}_0|, \underline{i}_{r_0} = \underline{r}_0 / r_0$
$\Gamma_1 = \cot \gamma_1$
$\lambda = r_0 / r_1$
$q = \lambda / \sin^2 \gamma_1$
$\Gamma_{max} = +\sqrt{q - 1}$
$\Gamma_{min} = -\sqrt{q\lambda - 1}$
1

Figure 4a. De-orbit Routine Detailed Flow Diagram

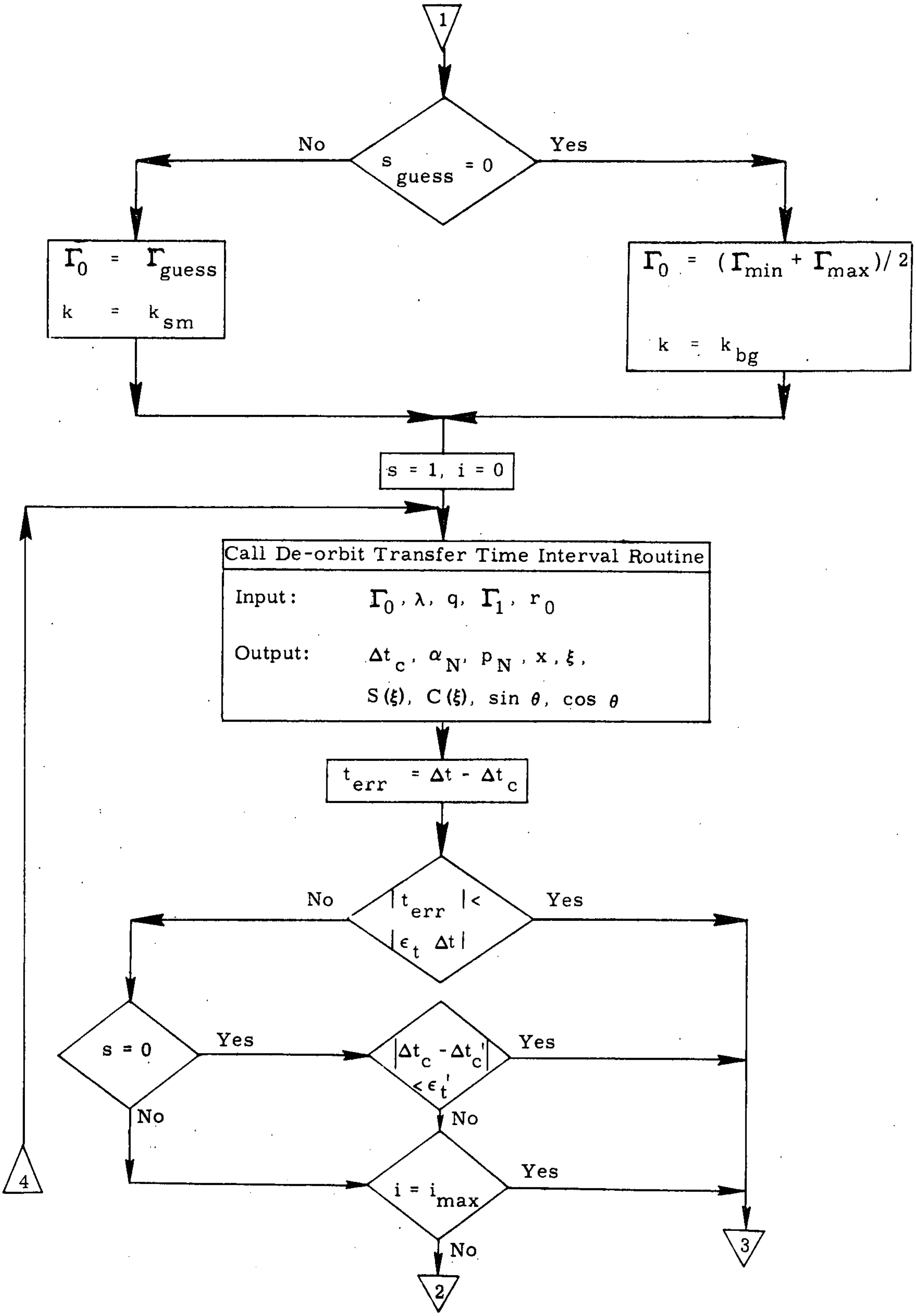

1
$s_{guess} = 0$
No
Yes
$\Gamma_0 = \Gamma_{guess}$
$k = k_{sm}$
$\Gamma_0 = (\Gamma_{min} + \Gamma_{max})/2$
$k = k_{bg}$
s = 1, i = 0
Call De-orbit Transfer Time Interval Routine
Input: Γ_0, λ, q, Γ_1, r_0
Output: Δt_c, α_N, p_N, x, ξ, S(ξ), C(ξ), sin θ, cos θ
$t_{err} = \Delta t - \Delta t_c$
$|t_{err}| < |\epsilon_t \Delta t|$
No
Yes
s = 0
Yes
No
$|\Delta t_c - \Delta t_c'| < \epsilon_t'$
Yes
No
$i = i_{max}$
Yes
No
4
2
3

Figure 4b. De-orbit Routine
Detailed Flow Diagram

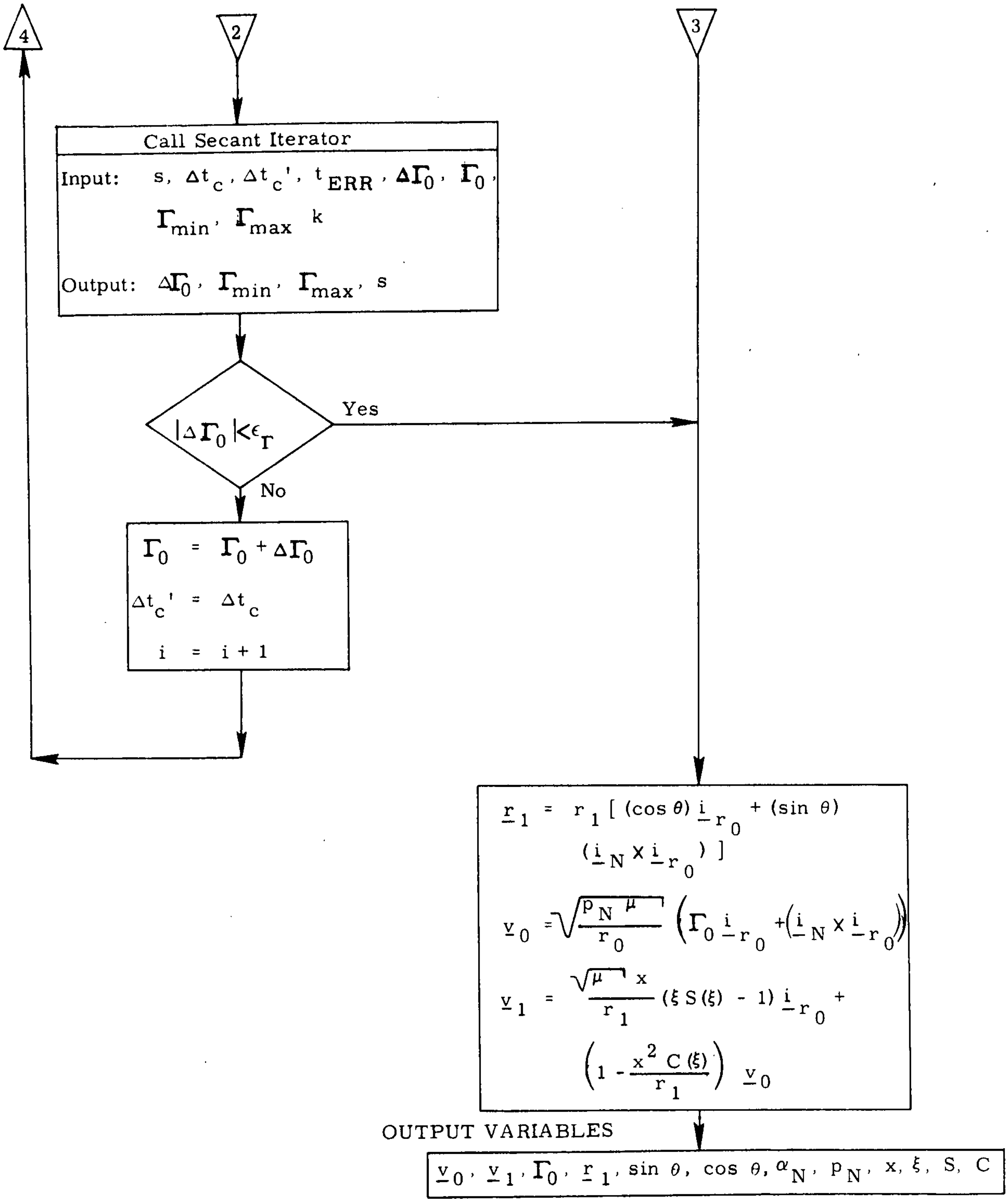

4
2
3
Call Secant Iterator
Input: $s, \Delta t_c, \Delta t_c', t_{ERR}, \Delta\Gamma_0, \Gamma_0, \Gamma_{min}, \Gamma_{max}, k$
Output: $\Delta\Gamma_0, \Gamma_{min}, \Gamma_{max}, s$
$|\Delta\Gamma_0| < \epsilon_\Gamma$
Yes
No
$\Gamma_0 = \Gamma_0 + \Delta\Gamma_0$
$\Delta t_c' = \Delta t_c$
$i = i + 1$
$\underline{r}_1 = r_1 [(\cos\theta)\underline{i}_{r_0} + (\sin\theta)(\underline{i}_N \times \underline{i}_{r_0})]$
$\underline{v}_0 = \sqrt{\frac{p_N \mu}{r_0}} \left(\Gamma_0 \underline{i}_{r_0} + (\underline{i}_N \times \underline{i}_{r_0})\right)$
$\underline{v}_1 = \frac{\sqrt{\mu}\, x}{r_1} (\xi S(\xi) - 1) \underline{i}_{r_0} + \left(1 - \frac{x^2 C(\xi)}{r_1}\right) \underline{v}_0$
OUTPUT VARIABLES
$\underline{v}_0, \underline{v}_1, \Gamma_0, \underline{r}_1, \sin\theta, \cos\theta, \alpha_N, p_N, x, \xi, S, C$

Figure 4c. De-orbit Routine
Detailed Flow Diagram

5.3 Subroutines or Coding Sequences used by the Conic Required Velocity Determination Routines

5.3.1 Lambert Transfer Time Interval Subroutine

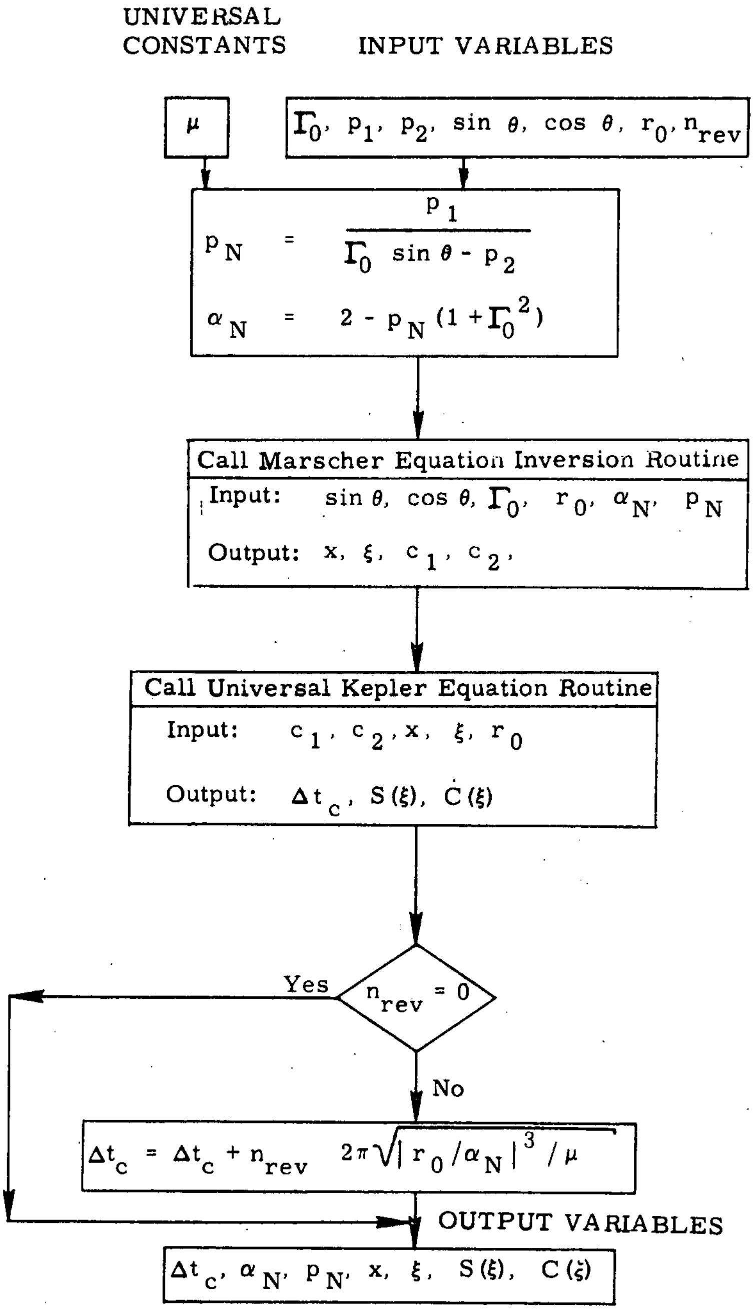
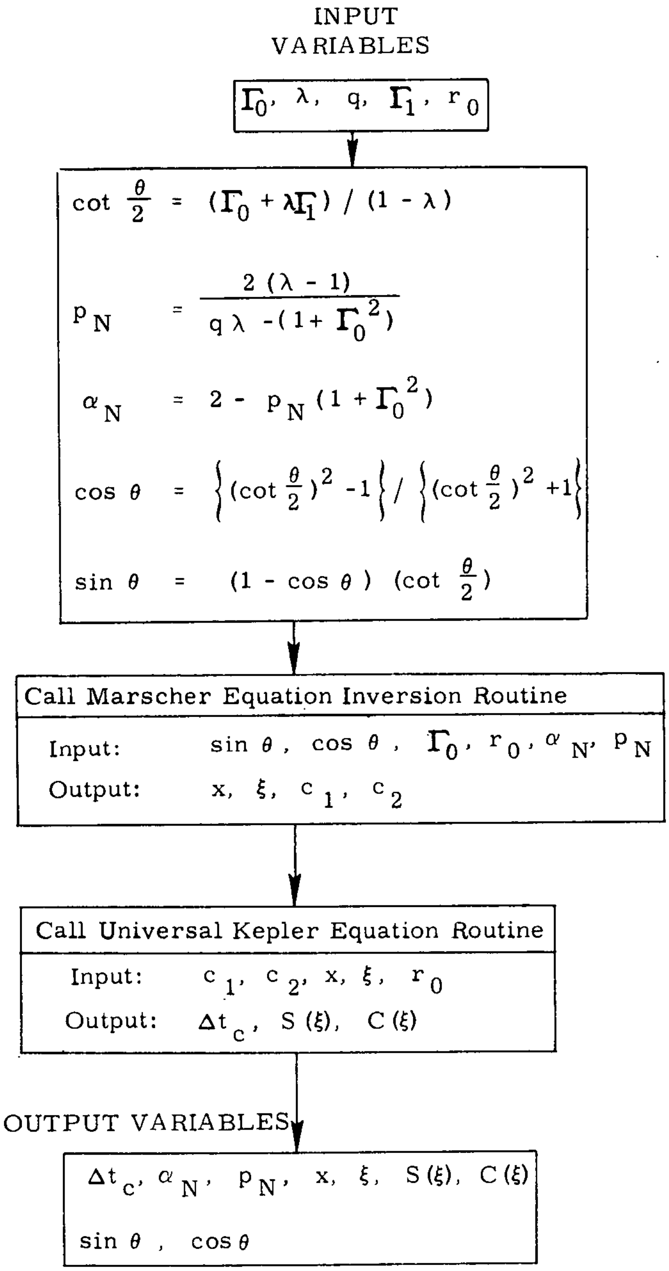

UNIVERSAL CONSTANTS
INPUT VARIABLES
μ
Γ_0, p_1, p_2, $\sin\theta$, $\cos\theta$, r_0, n_{rev}
$p_N = \dfrac{p_1}{\Gamma_0 \sin\theta - p_2}$
$\alpha_N = 2 - p_N(1 + \Gamma_0^2)$
Call Marscher Equation Inversion Routine
Input: $\sin\theta$, $\cos\theta$, Γ_0, r_0, α_N, p_N
Output: x, ξ, c_1, c_2,
Call Universal Kepler Equation Routine
Input: c_1, c_2, x, ξ, r_0
Output: Δt_c, $S(\xi)$, $\dot{C}(\xi)$
Yes
$n_{rev} = 0$
No
$\Delta t_c = \Delta t_c + n_{rev}\ 2\pi\sqrt{|r_0/\alpha_N|^3/\mu}$
OUTPUT VARIABLES
Δt_c, α_N, p_N, x, ξ, $S(\xi)$, $C(\xi)$

Figure 5. Lambert Transfer Time Interval Subroutine Detailed Flow Diagram

5.3.2 De-orbit Transfer Time Interval Subroutine



INPUT VARIABLES

$\Gamma_0, \lambda, q, \Gamma_1, r_0$

$$\cot\frac{\theta}{2} = (\Gamma_0 + \lambda\Gamma_1) / (1 - \lambda)$$

$$p_N = \frac{2(\lambda - 1)}{q\lambda - (1 + \Gamma_0^{\,2})}$$

$$\alpha_N = 2 - p_N(1 + \Gamma_0^{\,2})$$

$$\cos\theta = \left\{(\cot\frac{\theta}{2})^2 - 1\right\} / \left\{(\cot\frac{\theta}{2})^2 + 1\right\}$$

$$\sin\theta = (1 - \cos\theta)(\cot\frac{\theta}{2})$$

Call Marscher Equation Inversion Routine

Input: $\sin\theta, \cos\theta, \Gamma_0, r_0, \alpha_N, p_N$

Output: x, ξ, c_1, c_2

Call Universal Kepler Equation Routine

Input: c_1, c_2, x, ξ, r_0

Output: $\Delta t_c, S(\xi), C(\xi)$

OUTPUT VARIABLES

$\Delta t_c, \alpha_N, p_N, x, \xi, S(\xi), C(\xi)$

$\sin\theta, \cos\theta$

Figure 6. De-orbit Transfer Time Interval Subroutine Detailed Flow Diagram

5.3.3 Universal Kepler Eouation Subroutine

This subroutine is identical to the one used in the Kepler and Theta problems.

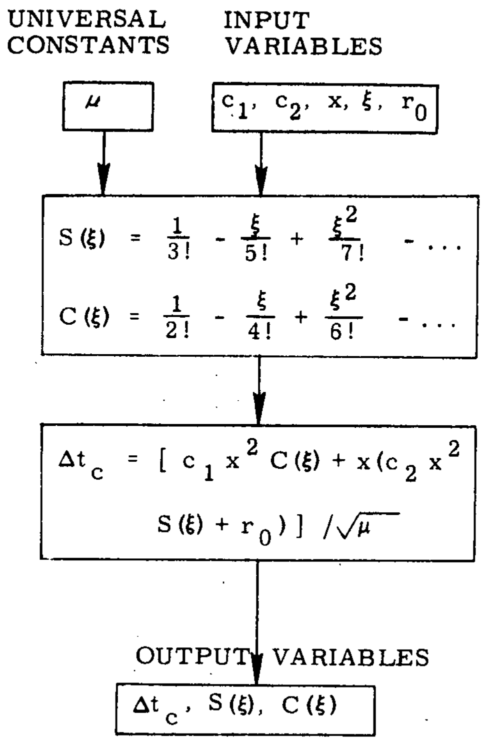

UNIVERSAL CONSTANTS
INPUT VARIABLES
μ
c_1, c_2, x, ξ, r_0
$S(\xi) = \frac{1}{3!} - \frac{\xi}{5!} + \frac{\xi^2}{7!} - \ldots$
$C(\xi) = \frac{1}{2!} - \frac{\xi}{4!} + \frac{\xi^2}{6!} - \ldots$
$\Delta t_c = [c_1 x^2 C(\xi) + x(c_2 x^2 S(\xi) + r_0)] / \sqrt{\mu}$
OUTPUT VARIABLES
$\Delta t_c, S(\xi), C(\xi)$

Figure 7. Universal Kepler Equation Subroutine Detailed Flow Diagram

5.3.4 Marscher Equation Inversion Subroutine

This subroutine is identical to the one used in the Theta problem.

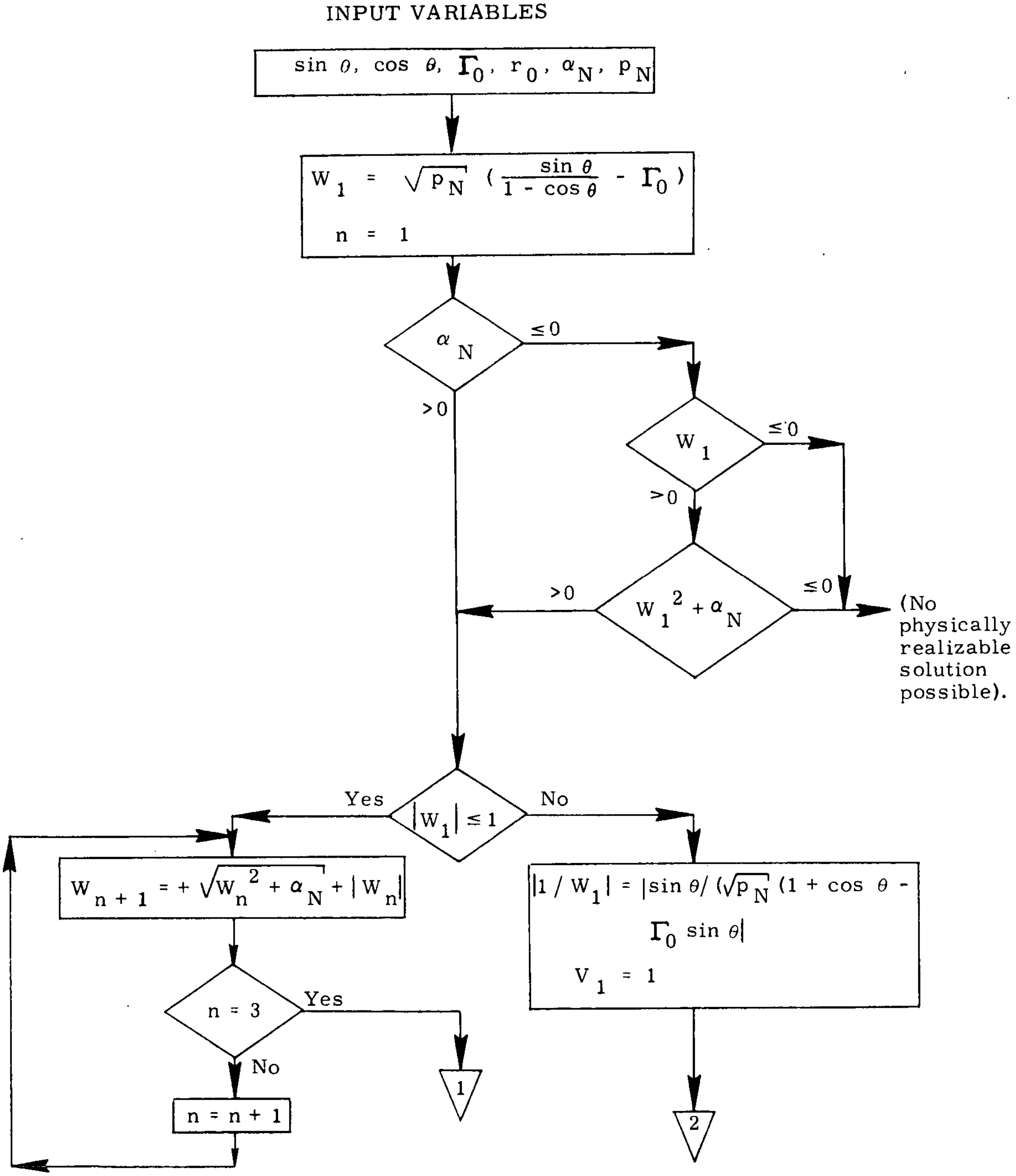

INPUT VARIABLES
sin θ, cos θ, Γ0, r0, αN, pN
W1 = √pN (sin θ / (1 - cos θ) - Γ0)
n = 1
αN
≤0
>0
W1
≤0
>0
W1^2 + αN
>0
≤0
(No physically realizable solution possible).
|W1| ≤ 1
Yes
No
Wn+1 = + √(Wn^2 + αN) + |Wn|
|1 / W1| = |sin θ/ (√pN (1 + cos θ - Γ0 sin θ|
V1 = 1
n = 3
Yes
No
n = n + 1
1
2

Figure 8a. Marscher Equation Inversion Subroutine Detailed Flow Diagram

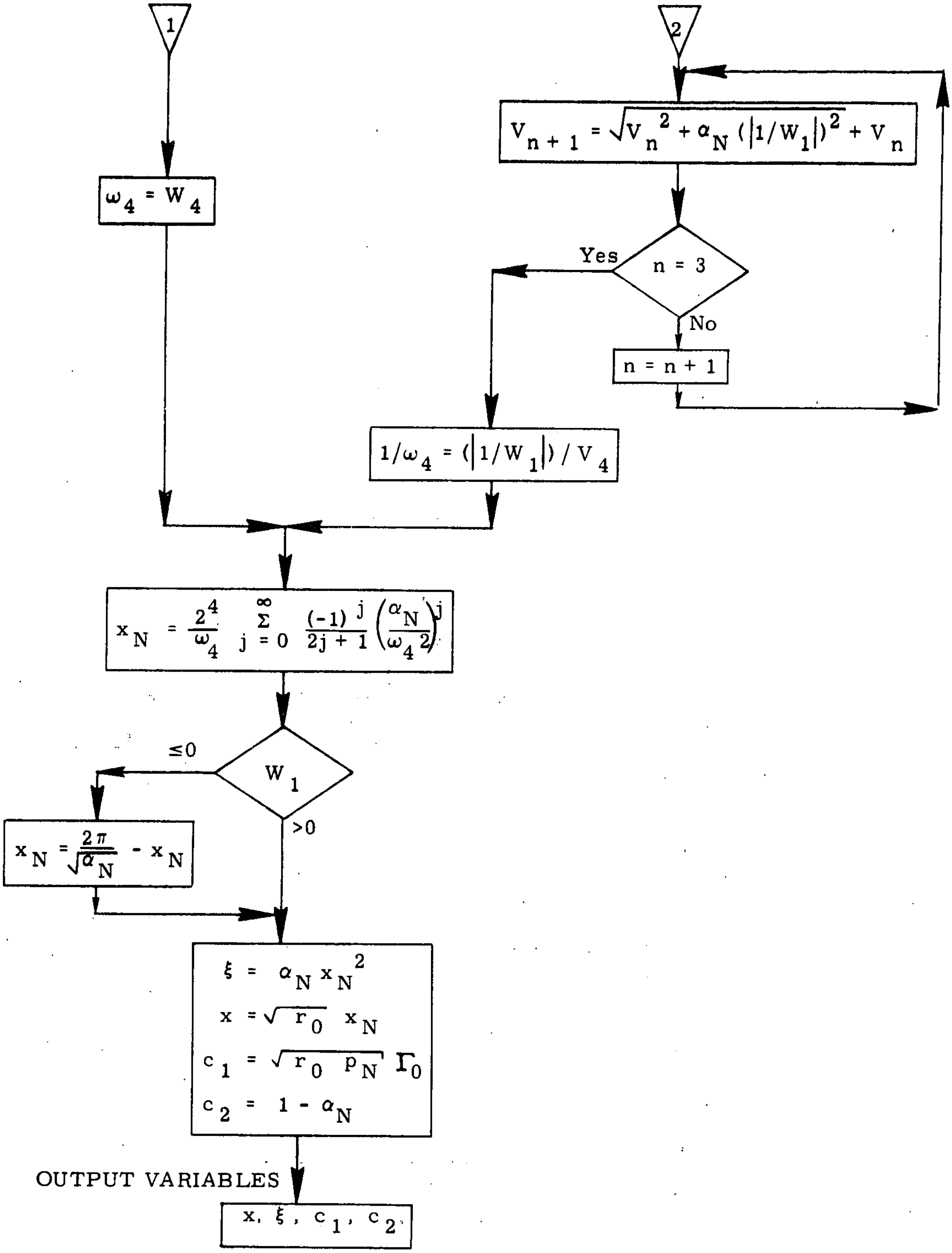

1
2
$\omega_4 = W_4$
$V_{n+1} = \sqrt{V_n^2 + \alpha_N (|1/W_1|)^2} + V_n$
$n = 3$
Yes
No
$n = n + 1$
$1/\omega_4 = (|1/W_1|)/V_4$
$x_N = \frac{2^4}{\omega_4} \sum_{j=0}^{\infty} \frac{(-1)^j}{2j+1} \left(\frac{\alpha_N}{\omega_4^2}\right)^j$
W_1
≤ 0
>0
$x_N = \frac{2\pi}{\sqrt{\alpha_N}} - x_N$
$\xi = \alpha_N x_N^2$
$x = \sqrt{r_0}\; x_N$
$c_1 = \sqrt{r_0 \; p_N}\; \Gamma_0$
$c_2 = 1 - \alpha_N$
OUTPUT VARIABLES
x, ξ, c_1, c_2

Figure 8b. Marscher Equation Inversion Subroutine Detailed Flow Diagram

5.3.5 Secant Iterator

This subroutine is identical (when k = 1/4) to the one used in the Theta problem.

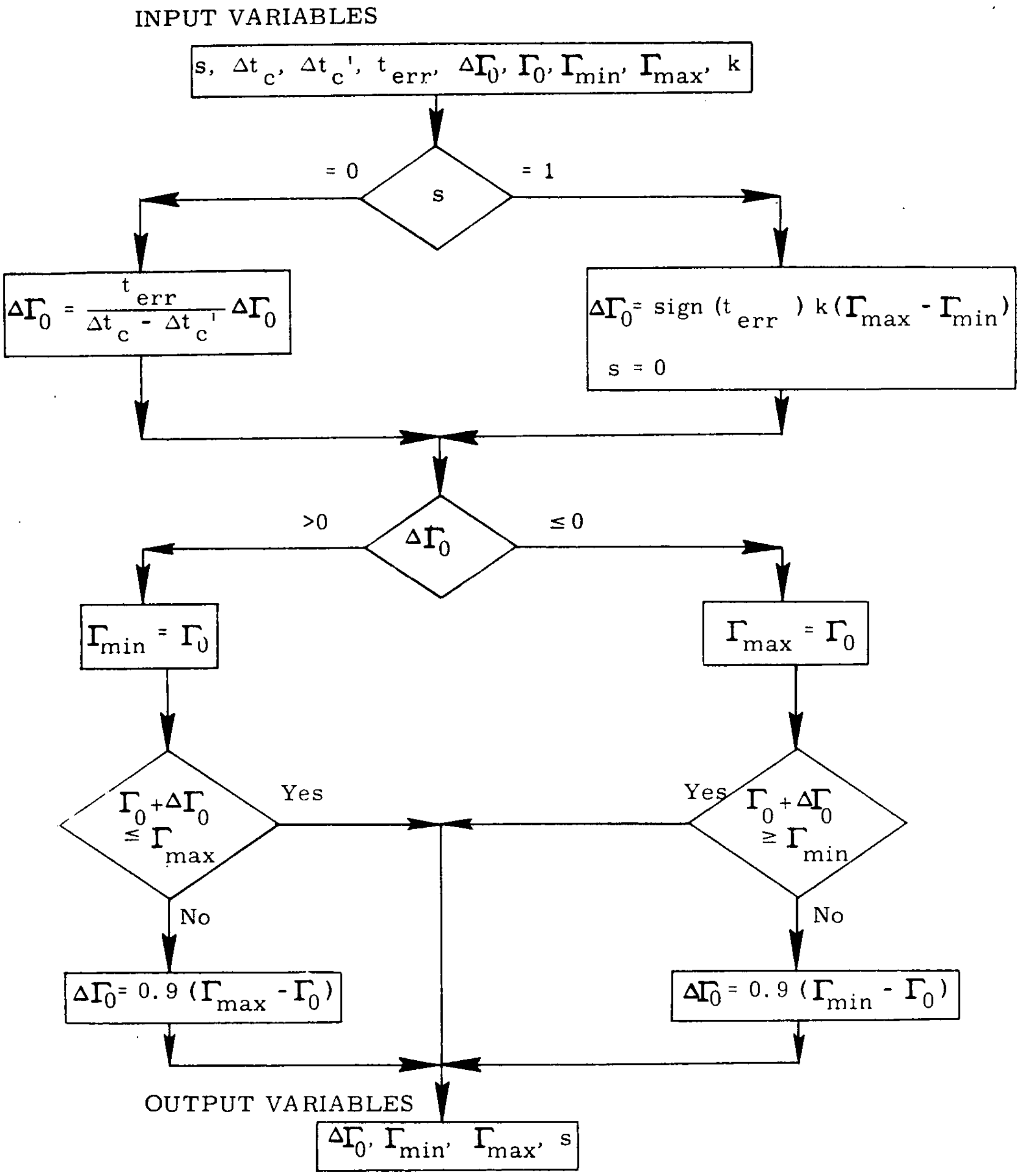

INPUT VARIABLES
s, Δt_c, $\Delta t_c'$, t_{err}, $\Delta\Gamma_0$, Γ_0, Γ_{min}, Γ_{max}, k
s
= 0
= 1
$\Delta\Gamma_0 = \frac{t_{err}}{\Delta t_c - \Delta t_c'} \Delta\Gamma_0$
$\Delta\Gamma_0 =$ sign (t_{err}) k ($\Gamma_{max} - \Gamma_{min}$)
s = 0
$\Delta\Gamma_0$
>0
≤ 0
$\Gamma_{min} = \Gamma_0$
$\Gamma_{max} = \Gamma_0$
$\Gamma_0 + \Delta\Gamma_0 \le \Gamma_{max}$
Yes
No
$\Gamma_0 + \Delta\Gamma_0 \ge \Gamma_{min}$
Yes
No
$\Delta\Gamma_0 = 0.9 (\Gamma_{max} - \Gamma_0)$
$\Delta\Gamma_0 = 0.9 (\Gamma_{min} - \Gamma_0)$
OUTPUT VARIABLES
$\Delta\Gamma_0$, Γ_{min}, Γ_{max}, s

Figure 9. Secant Iterator
Detailed Flow Diagram

5. 3. 6 Multi-revolution Bounds Adjustment Coding Sequence

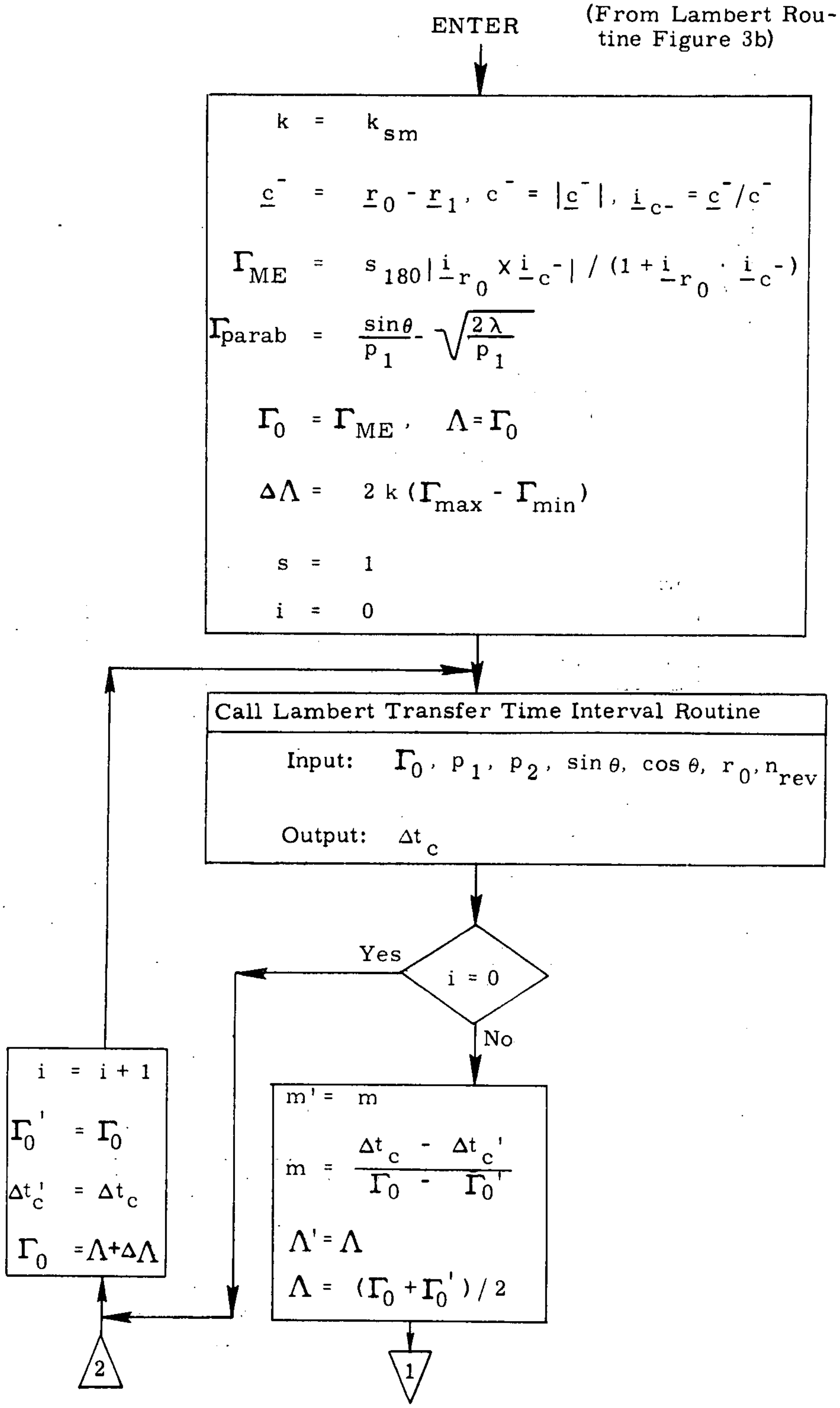

ENTER
(From Lambert Routine Figure 3b)
$k = k_{sm}$
$\underline{c}^- = \underline{r}_0 - \underline{r}_1,\ c^- = |\underline{c}^-|,\ \underline{i}_{c^-} = \underline{c}^-/c^-$
$\Gamma_{ME} = s_{180}|\underline{i}_{r_0} \times \underline{i}_{c^-}| / (1 + \underline{i}_{r_0} \cdot \underline{i}_{c^-})$
$\Gamma_{parab} = \frac{\sin\theta}{p_1} - \sqrt{\frac{2\lambda}{p_1}}$
$\Gamma_0 = \Gamma_{ME},\ \Lambda = \Gamma_0$
$\Delta\Lambda = 2k(\Gamma_{max} - \Gamma_{min})$
$s = 1$
$i = 0$
Call Lambert Transfer Time Interval Routine
Input: $\Gamma_0,\ p_1,\ p_2,\ \sin\theta,\ \cos\theta,\ r_0, n_{rev}$
Output: Δt_c
$i = 0$
Yes
No
$i = i + 1$
$\Gamma_0' = \Gamma_0$
$\Delta t_c' = \Delta t_c$
$\Gamma_0 = \Lambda + \Delta\Lambda$
$m' = m$
$m = \frac{\Delta t_c - \Delta t_c'}{\Gamma_0 - \Gamma_0'}$
$\Lambda' = \Lambda$
$\Lambda = (\Gamma_0 + \Gamma_0')/2$
2
1

Figure 10a. Multi-Revolution Bounds Adjustment Coding Sequence Detailed Flow Diagram

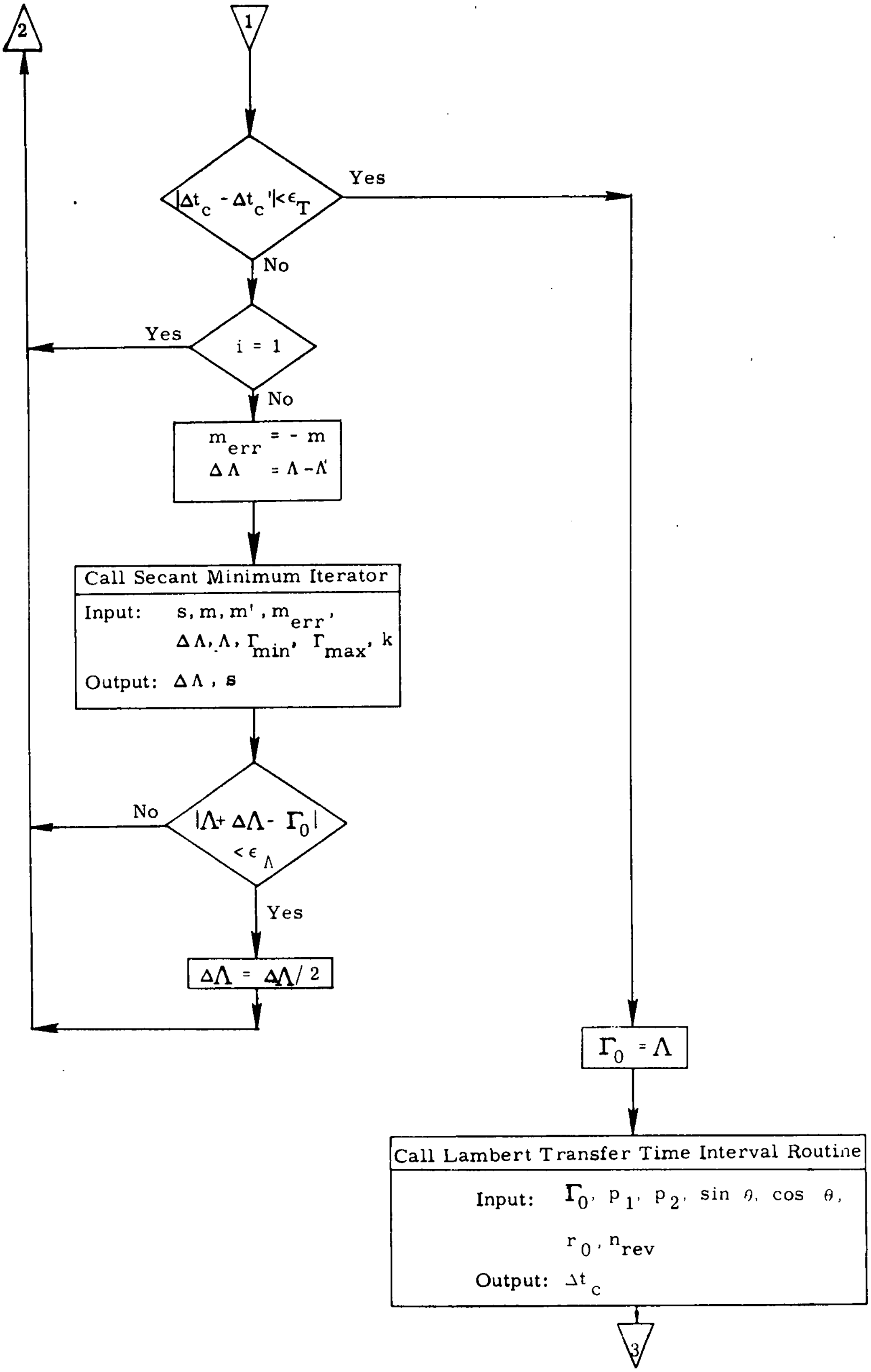

2
1
$|\Delta t_c - \Delta t_c'| < \epsilon_T$
Yes
No
Yes
$i = 1$
No
$m_{err} = - m$
$\Delta\Lambda = \Lambda - \Lambda'$
Call Secant Minimum Iterator
Input: $s, m, m', m_{err}, \Delta\Lambda, \Lambda, \Gamma_{min}, \Gamma_{max}, k$
Output: $\Delta\Lambda$, s
No
$|\Lambda + \Delta\Lambda - \Gamma_0| < \epsilon_\Lambda$
Yes
$\Delta\Lambda = \Delta\Lambda / 2$
$\Gamma_0 = \Lambda$
Call Lambert Transfer Time Interval Routine
Input: $\Gamma_0, p_1, p_2, \sin\theta, \cos\theta, r_0, n_{rev}$
Output: Δt_c
3

Figure 10b. Multi-Revolution Bounds Adjustment Coding Sequence Detailed Flow Diagram

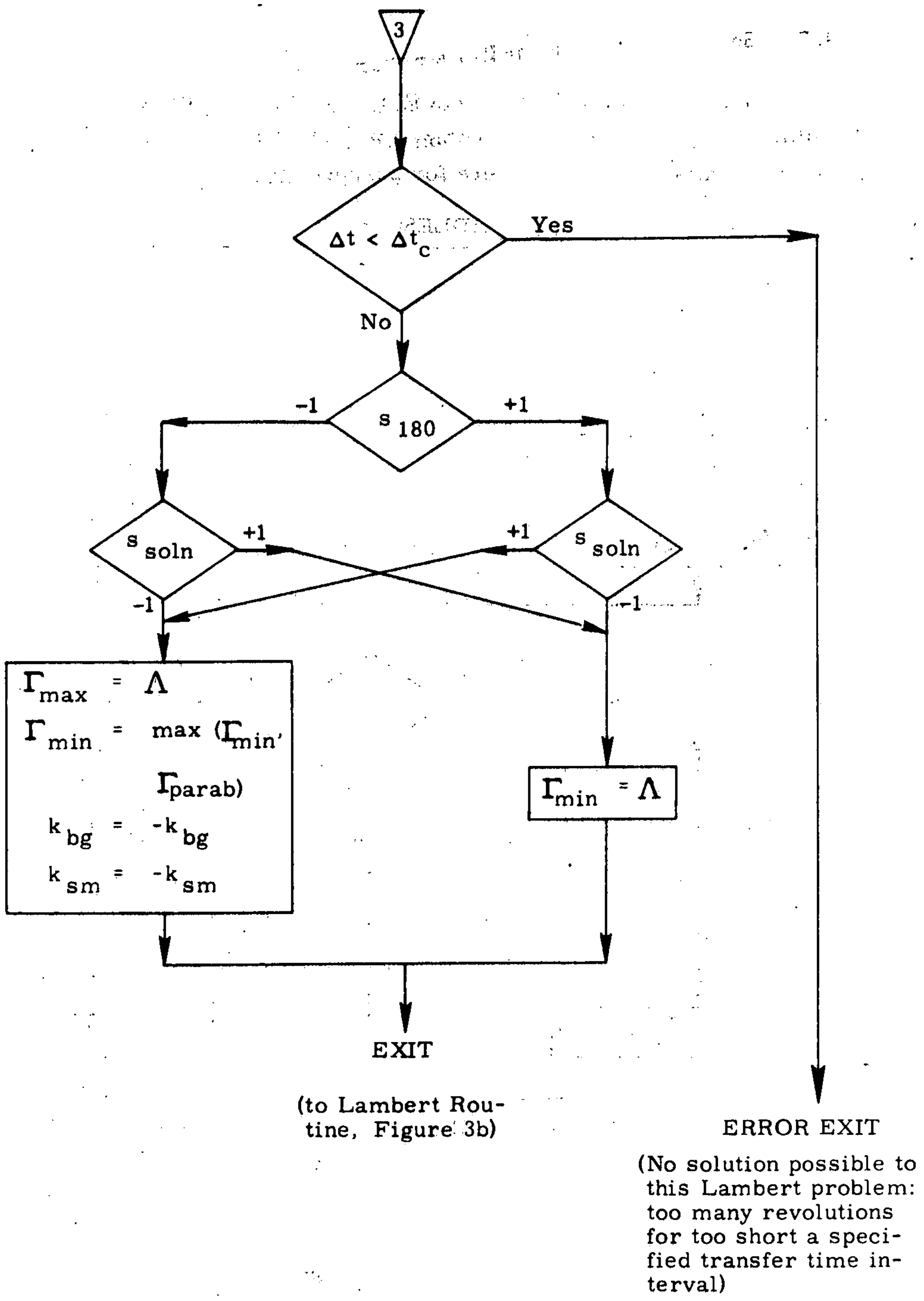

3
$\Delta t < \Delta t_c$
Yes
No
s_{180}
-1
+1
s_{soln}
+1
-1
s_{soln}
+1
-1
$\Gamma_{max} = \Lambda$
$\Gamma_{min} = \max (\Gamma_{min}, \Gamma_{parab})$
$k_{bg} = -k_{bg}$
$k_{sm} = -k_{sm}$
$\Gamma_{min} = \Lambda$
EXIT
(to Lambert Routine, Figure 3b)
ERROR EXIT
(No solution possible to this Lambert problem: too many revolutions for too short a specified transfer time interval)

Figure 10c. Multi-Revolution Bounds Adjustment Coding Sequence Detailed Flow Diagram

5.3.7 Secant Minimum Iterator

This subroutine is very similar, though not identical, to the Secant Iterator. They can easily be combined into one routine, although they have been diagrammed separately here for purposes of clarity.

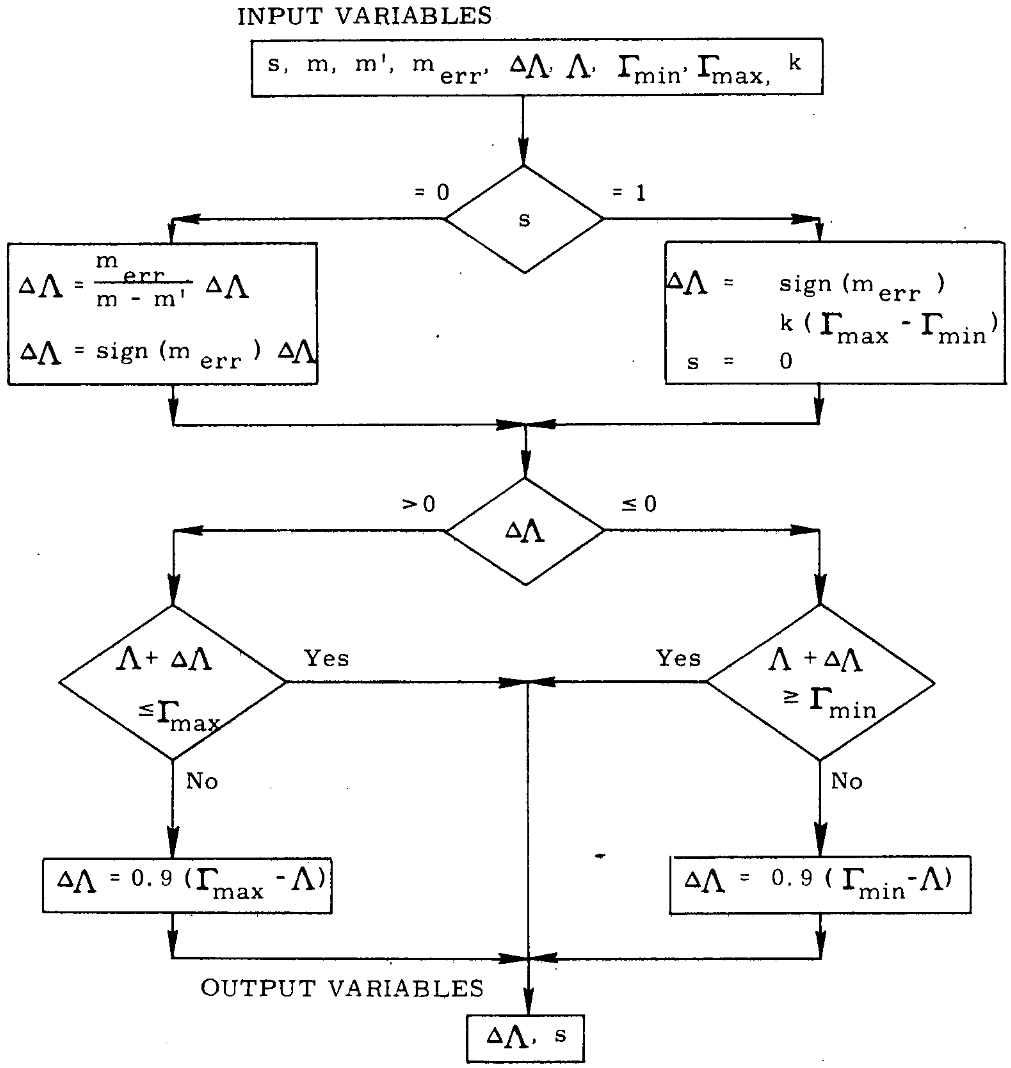

INPUT VARIABLES
s, m, m', m_{err}, $\Delta\Lambda$, Λ, Γ_{min}, Γ_{max}, k
= 0
s
= 1
$\Delta\Lambda = \frac{m_{err}}{m - m'} \Delta\Lambda$
$\Delta\Lambda$ = sign (m_{err}) $\Delta\Lambda$
$\Delta\Lambda$ = sign (m_{err}) k ($\Gamma_{max} - \Gamma_{min}$)
s = 0
> 0
$\Delta\Lambda$
≤ 0
$\Lambda + \Delta\Lambda \leq \Gamma_{max}$
Yes
No
Yes
$\Lambda + \Delta\Lambda \geq \Gamma_{min}$
No
$\Delta\Lambda = 0.9 (\Gamma_{max} - \Lambda)$
$\Delta\Lambda = 0.9 (\Gamma_{min} - \Lambda)$
OUTPUT VARIABLES
$\Delta\Lambda$, s

Figure 11. Secant Minimum Iterator
Detailed Flow Diagram

6. SUPPLEMENTARY INFORMATION

The formulation of the solutions of the Lambert and De-orbit problems in terms of an iteration on the cotangent of the initial flight path angle was conceived by Marscher (1965), who developed all the fundamental relationships including the bounds and the remarkable Marscher Equation:

$$\cot\frac{\theta}{2} = \frac{r_0}{\sqrt{p\,a}}\cot\frac{\Delta E}{2} + \cot\gamma_0 \quad \text{(elliptic)}$$

$$\cot\frac{\theta}{2} = \frac{r_0}{\sqrt{-pa}}\coth\frac{\Delta H}{2} + \cot\gamma_0 \quad \text{(hyperbolic)}$$

or in universal form:

$$\cot\frac{\theta}{2} = \frac{r_0}{\sqrt{p}}\,\frac{[1-\alpha x^2 S(\alpha x^2)]}{x\,C(\alpha x^2)} + \cot\gamma_0 \ .$$

Marscher's original solutions of the Lambert and De-orbit problems, however, involved a double loop iterative procedure in which the universal Marscher Equation was solved iteratively for x in an inner loop during each pass through the outer loop iteration on $\cot\gamma_0$ and the transfer time interval. Marscher's solutions were simplified to a single loop iteration through the inversion of Marscher's Equation which was derived by Robertson (1967). The expressions for the Universal Kepler Equation and the terminal velocity vector are well known and are given in Battin (1964).

Krause organized the details of the computation in both routines. He also developed the two secant iterators (Krause, 1967).

The formulation of the Lambert and De-orbit problems given in the preceeding sections is essentially that used in the Apollo program, and hence has been thoroughly exercised. Among the major advantages of this formulation are: (1) the use of an independent variable which is simply related to a single physical quantity of practical interest (the flight-path angle), and (2) the sharing of large portions of the calculation between the Lambert, De-orbit, Kepler, and Theta Routines in the form of common subroutines, permitting a considerable reduction in the amount of computer memory required for the storage of the instructions. A limitation of the formulation is that it cannot handle rectilinear transfers since the independent variable Γ_0 is infinite for all such cases.

The value of Γ_0 corresponding to the minimum energy transfer may be derived from a consideration of problem 3.4 of Battin (1964).

REFERENCES

1. Battin, R. H., 1964, Astronautical Guidance, McGraw-Hill.

2. Brand, T.J., 1971, Precision Required Velocity Determination, M.I.T./DL Space Shuttle GN& C Equation Document No. 13 (Revision 1), September 1971.

3. Krause, K. W., 1967, Generalized Slope Iterator, MIT/DL Space Guidance Analysis Memo No. 4-67.

4. Krause, K. W., 1968, A Unified Method of Solving Initial Value and Boundary Value Conic Trajectory Problems, TRW Interoffice Correspondence No. 3424.9-15 (January 1968).

5. Marscher, W., 1965, A Unified Method of Generating Conic Sections, MIT/DL Report R-479, February 1965.

6. Robertson, W. M., 1967, Explicit Universal Series Solutions for the Universal Variable x, MIT/DL Space Guidance Analysis Memo No. 8-67.

7. Robertson, W. M., 1971 Conic State Extrapolation M.I.T./DL Space Shuttle GN & C Equation Document No. 3, February 1971.

9.7.1.2 Required Velocity Determination, Precision

SPACE SHUTTLE

GN&C SOFTWARE EQUATION SUBMITTAL

Software Equation Section Required velocity determination (precision)

Submittal No. 27A

Function: To provide initialization of powered flight guidance equations for short finite-length burns.

Module No. OG-3 Function No. 1,2,3 (MSC 03690)
OG-5 1,2

Submitted by: T. Brand Co. MIT No. 13 (Rev. 1)

Date: 21 Oct. 1971

NASA Contract: C. Lively Organization EG-2

Approved by Panel III K.J. Cof Date 10/21/71

Summary Description: This routine provides targeting for powered flight guidance during short burns in a non-Keplerian gravity field. The effects of finite burn times and of perturbations from a conic gravity field are accommodated by an offset target such that during the manuever, simple conic computations may be employed with great accuracy. The revision includes logic (1) to preclude difficulties for transfer angles near 180° and (2) to permit disabling gravity-perturbation computations.

Shuttle Configuration: The software is essentially independent of configuration.

Comments:

(Design Status)

(Verification Status)

Panel Comments:

Revision: A. Prior submittal Aug. 1971

1. INTRODUCTION

Calculation of the precision required velocity which satisfies terminal position and time-of-flight constraints in a non-Keplerian gravity field is a computation time consuming process, especially in an on-board computer. Therefore, targeting calculations prior to a maneuver are customarily used to predict and compensate for the effects of the perturbations from a conic gravity field, so that during the maneuver only the much simpler conic related computations will have to be performed.

For Lambert aim point maneuvers (described in Reference 2) an adjustment to the terminal (target) position vector will suffice to provide this compensation. This adjusted terminal position, referred to as an offset target, must compensate for gravity perturbations throughout both the maneuver and subsequent coasting flight. Then the required velocity determined by the Lambert routine to intercept the offset target in a conic gravity field is identical to the velocity required to intercept the true target in the non-Keplerian field.

The traditional technique of predicting the effects of gravitational perturbations over the trajectory involves approximating the maneuver by an impulsive velocity change, and hence assuming a coasting trajectory between the initial (ignition) and target positions. However, due to the non-zero length of the maneuver, the actual trajectory will not follow the path predicted by the impulsive approximation, but rather a neighboring path. The difference in the perturbing acceleration between the two paths accumulates over the entire trajectory, resulting in a miss at the target. Since the coasting portion of the trajectory is generally much longer than the thrusting portion, it is important to accurately predict the perturbing effects over this portion of the trajectory. This is accomplished by determining the initial conditions for a coasting trajectory which is coincident with the actual trajectory after thrust termination. A detailed derivation of this technique can be found in Brand (1971) (Reference 1), and a functional description of the procedure follows.

NOMENCLATURE

a_T Estimated magnitude of the thrust acceleration

d Number of columns of navigation filter weighting matrix (set to 0 in this routine since the matrix is not required)

f Thrust

f_{ACS} Magnitude of the attitude control system translational thrust

f_{OMS} Magnitude of the nominal orbital maneuvering system engine thrust

$\underline{i}_N$ Unit normal to the trajectory plane (in the direction of the angular momentum at ignition)

m Current estimated vehicle mass

n Iteration counter

n_{max} Iteration limit

n_{rev} Integral number of complete 360° revolutions to be made in the desired transfer

$\underline{r}_0$ Initial (ignition) position

$\underline{r}'_0$ Adjusted initial position used to define coasting trajectory

$\underline{r}_1$ Target position (input to the routine)

$\underline{r}'_1$ Terminal position (output of the routine)

$\underline{r}_{1c}$ Offset target position

s_{cone} Switch set in the Lambert routine to indicate transfer is near 180° (see Reference 4 for complete description)

9.7.1.2 Precision (continued)

s_{eng}	Engine select switch
s_{fail}	Switch set to indicate non-convergence
s_{guess}	Switch set to indicate an estimate of independent variable Γ will be input to the Conic Required Velocity Determination Routine
s_{pert}	Switch set to indicate which perturbing accelerations should be included in the offset target calculation ($s_{pert} = 0$ indicates only conic calculations; see Reference 3 for complete description of other switch settings)
s_{proj}	Switch set when the target vector must be projected into the plane defined by $\underline{i}_N$
s_{soln}	Switch indicating which of two physically possible solutions is desired in the multi-revolution transfer (see Reference 4 for complete description)
t_0	Ignition time
t_1	Target time of arrival
$\underline{v}_0$	Initial (ignition) velocity
$\underline{v}_0'$	Initial (and required) velocity on the coasting trajectory
$\underline{v}_{1c}$	Terminal velocity of a conic trajectory
$\underline{v}_1'$	Terminal velocity (output of the routine)
Γ_{guess}	Guess of the independent variable Γ used in the Conic Required Velocity Determination Routine
$\Delta\underline{r}$	Target miss resulting from perturbations
Δr_{proj}	Out-of-plane target miss due to projection of the target vector

9.7.1.2 Precision (continued)

Δt Transfer time $(t_1 - t_0)$

$\Delta \underline{v}_0$ Required velocity change

Δv_0 Magnitude of the required velocity change

ϵ_{conv} Convergence criterion: target miss of the numerically integrated trajectory

$\epsilon_{\theta T}$ Tolerance criterion establishing a cone around the minus $\underline{r}_0$ direction inside of which the target vector will be projected into the plane $\underline{i}_N$ $\left[\epsilon_{\theta T} = \sin \text{(half cone angle)}\right]$

θ Transfer angle (true anomaly difference) at the start of the thrusting maneuver

θ_T Approximate central angle traversed during the thrusting maneuver

θ_1 Approximate transfer angle to the target at the termination of the thrusting maneuver $\left[\theta_1 = \theta - \theta_T\right]$

ω Approximate orbital rate

2. FUNCTIONAL FLOW DIAGRAM

A functional flow diagram describing the calculations necessary to determine the precision required velocity and offset target is presented in Figure 1. Since this technique compensates for the non-impulsive nature of the maneuver, it requires an estimate of the expected thrust acceleration. Then the initial position can be offset from the actual position such that a coasting trajectory which is coincident with the actual trajectory after thrust termination can be defined. Figure 2 illustrates the concept.

The calculation of the coasting trajectory initial position requires an estimate of the required velocity change, and therefore two passes are made through the Lambert routine before numerically integrating to determine the effects of gravitational perturbations. The first Lambert solution is used to determine the impulsive velocity change required. Based upon this, an estimate of the initial position for the coasting trajectory can be calculated. Then the second Lambert solution determines the velocity required from the adjusted initial position, thus defining the coasting trajectory.

For transfers angles which are odd multiples of 180°, Lambert's problem has a partial physical singularity in that the plane of the transfer becomes indeterminate. A detailed description of this singularity can be found in Reference 4. To prevent possible problems in both targeting and guiding a maneuver whose transfer angle lies near this singularity, logic has been included in this routine to determine whether the transfer angle approaches this singularity at any time during the maneuver. If this is the case, the target vector is projected into the orbital plane defined by the premaneuver position and velocity, thus preventing any plane change.

If only conic calculations are desired, the routine is exited after the two Lambert solutions are completed. If not, subsequent numerical integration determines the target miss resulting from the effects of gravitational perturbations over this path. To compensate for these effects, the target vector for the Lambert routine is offset from the actual target by the negative of the miss vector. Since the adjusted initial position, target offset, and effects of gravitational perturbations are all interdependent, the process is repeated until changes in the offset target position are small enough to indicate convergence. Three passes (two iterations) are normally sufficient to establish the offset within a few feet.

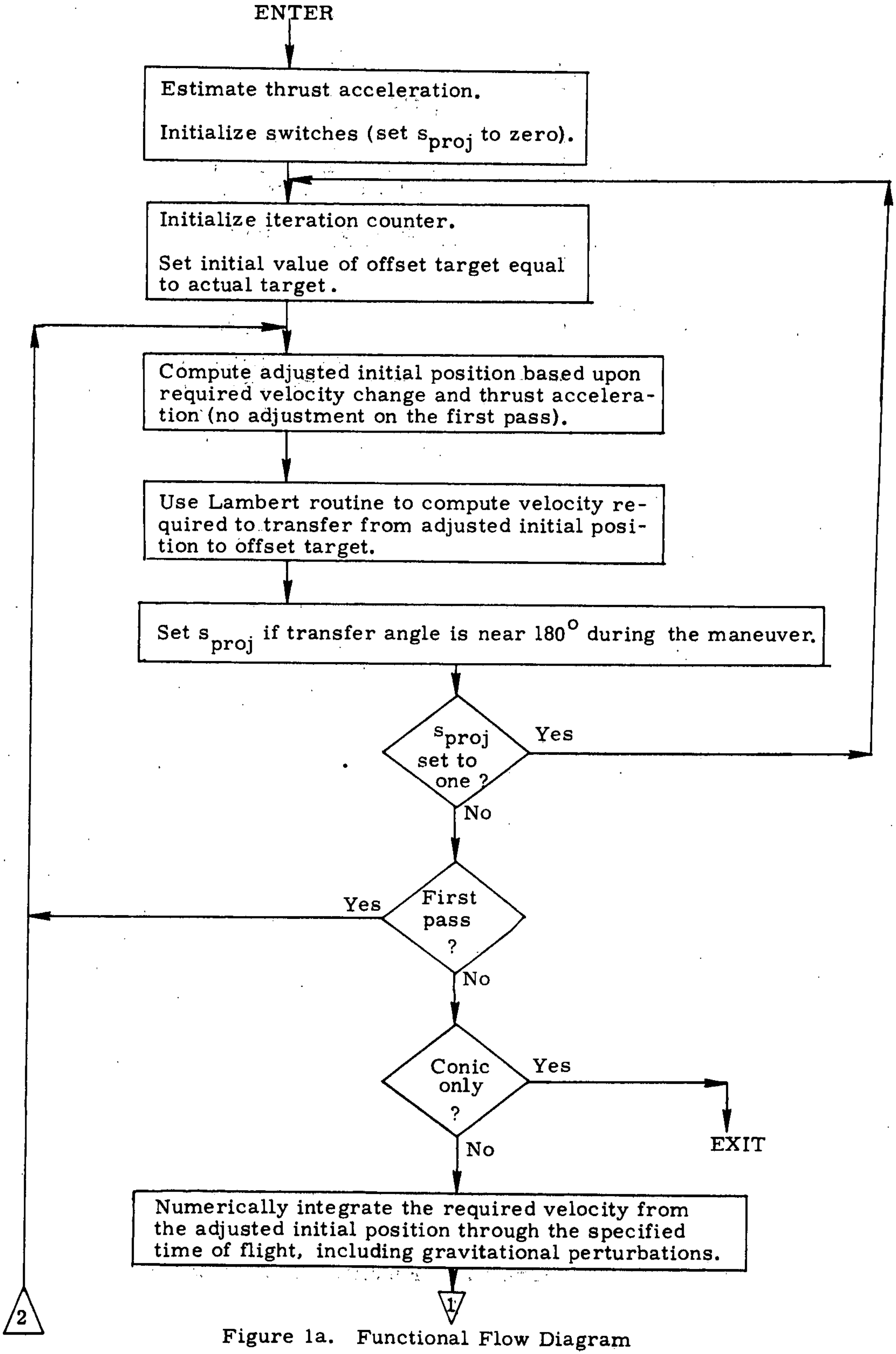

ENTER
Estimate thrust acceleration.
Initialize switches (set s_{proj} to zero).
Initialize iteration counter.
Set initial value of offset target equal to actual target.
Compute adjusted initial position based upon required velocity change and thrust acceleration (no adjustment on the first pass).
Use Lambert routine to compute velocity required to transfer from adjusted initial position to offset target.
Set s_{proj} if transfer angle is near 180° during the maneuver.
s_{proj} set to one ?
Yes
No
First pass ?
Yes
No
Conic only ?
Yes
No
EXIT
Numerically integrate the required velocity from the adjusted initial position through the specified time of flight, including gravitational perturbations.
1
2

Figure 1a. Functional Flow Diagram

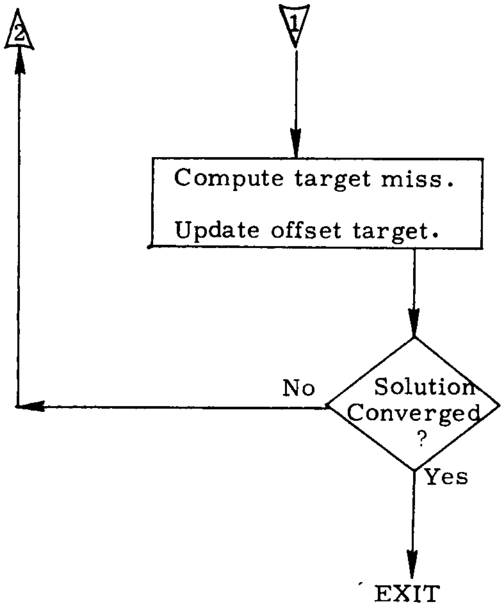


Figure 1b. Functional Flow Diagram

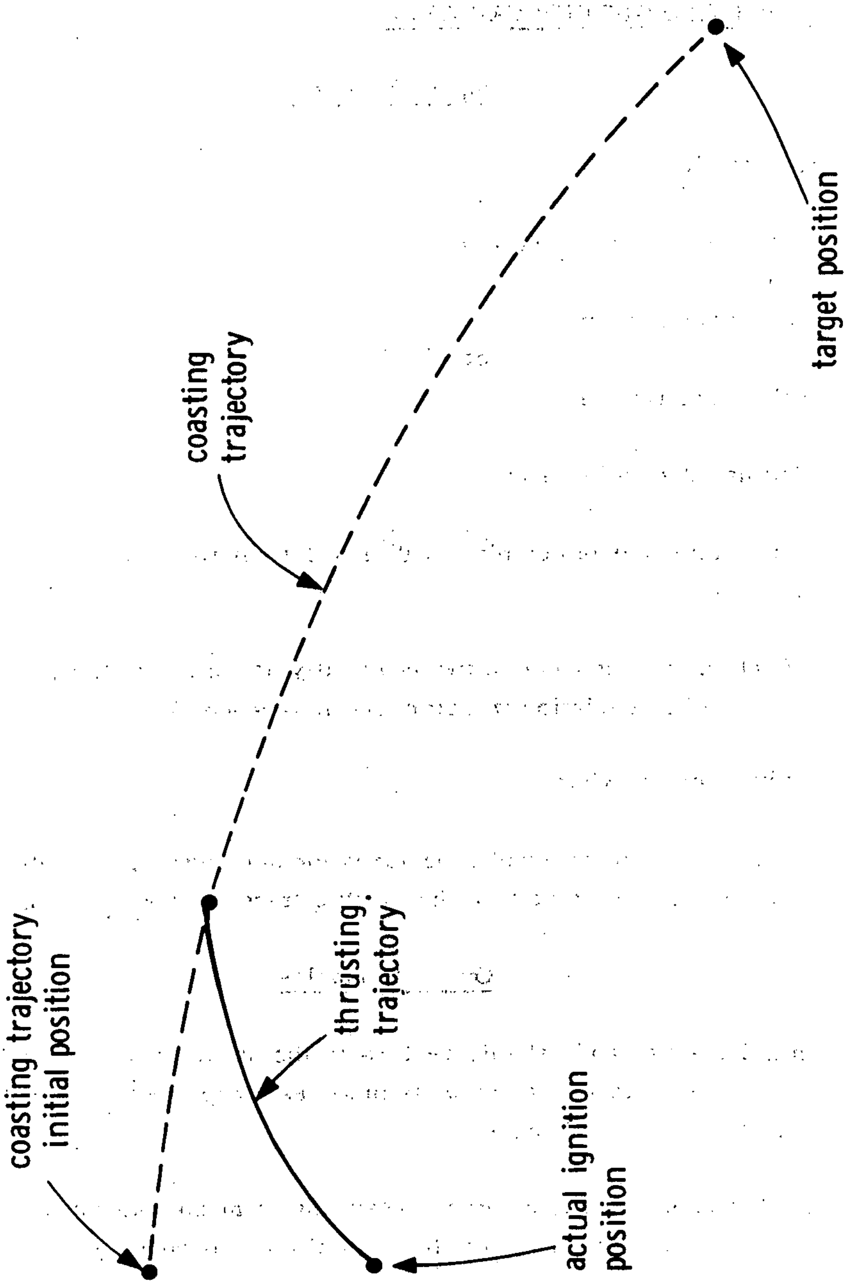


Figure 2. Coasting Trajectory Illustration

3. INPUT AND OUTPUT VARIABLES

Input Variables

t_0 Ignition time

$\underline{r}_0$, $\underline{v}_0$ State vector at ignition time t_0

t_1 Target time of arrival

$\underline{r}_1$ Target position vector

m Estimated vehicle mass

n_{rev} Integral number of complete 360° revolutions to be made in the desired transfer

s_{soln} Switch indicating which of two physically possible solutions is desired in the multi-revolution transfer (see Reference 4)

s_{eng} Engine select switch

s_{pert} Switch set to indicate which perturbation accelerations are desired to determine required velocity on the coasting trajectory ($s_{pert} = 0$ indicates conic only)

Output Variables

$\underline{r}_0'$, $\underline{v}_0'$ Initial position and velocity vectors on the coasting trajectory (differencing v_0' and the premaneuver velocity provides a precise measure of the required velocity change)

$\underline{r}_1'$, $\underline{v}_1'$ Position and velocity at time t_1 resulting from the maneuver (includes the effects of projection into the orbital plane if required)

9.7.1.2 Precision (continued)

$\underline{r}_{1c}$* Offset target position (identical to $\underline{r}_1'$ when only conic calculations are desired)

$\underline{i}_N$* Unit normal to the premaneuver orbital plane

s_{proj}* Switch set to indicate that the transfer angle is near an odd multiple of 180° and therefore the target has been projected into the premaneuver orbital plane

Δr_{proj} Out-of-plane target miss due to projection of the target vector into the premaneuver orbital plane ($\Delta r_{proj} = 0$ when projection is unnecessary)

s_{fail} Switch set if convergence difficulties are encountered in the iterative scheme used to compensate for gravitational perturbations ($s_{fail} = 0$ indicates no convergence difficulties)

* These outputs are required by the powered flight guidance to perform the Lambert aimpoint maneuver.

4. DESCRIPTION OF EQUATIONS

The computational process used to calculate a precision required velocity and offset target makes extensive use of the Conic Required Velocity Determination Routine (Lambert routine) and the Precision State Extrapolation Routine (coasting integration). The calculation of the precision required velocity requires at least two Lambert solutions so that the concept of a coasting trajectory can be used to compensate for the finite length of the maneuver. In addition, if an offset target is desired to compensate for gravitational perturbations a straight forward iterative technique involving successive Lambert solutions followed by precision integration is used. Since these techniques are described in Section 2, the remainder of this section will discuss the treatment of the singularity in the Lambert solution for transfer angles which are odd multiples of 180°.

The reader should be familiar with the discussion in subsection 4.7 of Reference 4. From that discussion, it is evident that if the initial and target position vectors used to define the transfer plane are nearly colinear, small changes in either of these position vectors can cause large changes in the transfer plane. However, even small changes in the transfer plane cause large changes in the required Δv. Thus during the targeting process, when successive solutions of Lambert's problem are necessary, the adjustments being made to the initial and target positions can result in radically different transfer planes. Likewise, during the powered maneuver, out-of-plane thrust transients can cause changes in the initial position which substantially alter the transfer plane.

Since plane change maneuvers are most efficiently performed when the transfer angle to the target is 90 or 270 degrees, it is not practical to make plane changes when the transfer angle is near 180 degrees. To prevent costly plane changes, therefore, logic has been included in both the Lambert routine and this routine to force the solution into the premaneuver orbital plane, which is defined by the unit normal $\underline{i}_N$. During the targeting process this routine determines whether the transfer angle during the maneuver is likely to lie in the region of 180 degrees (or 540, 900, etc.). If this is the case, the switch s_{proj} is set. This forces all solutions of Lambert's problem for this maneuver (both for this routine and the powered flight guidance) to be based on initial and target positions which have been projected into the premaneuver orbital plane.

Since target vectors for Lambert transfers in the region of 180 degrees should logically be limited to the premaneuver orbital plane, this projection should have no effect. However, during the process of calculating an offset target, numerical integration may indicate an out-of -plane target miss due to gravitational perturbations. Normally, compensation for this is accomplished by an offset target equivalently out-of-plane in the opposite direction. Near the 180 degree singularity, however, this would cause large changes in the transfer plane. Therefore when the switch s_{proj} is set, indicating transfers near the singularity, no compensation for the out-of-plane effects of gravitational perturbations is allowed (or practical). In this case compensation is limited to in-plane effects and accomplished by an in-plane offset target. The out-of-plane miss Δr_{proj} resulting from this technique is returned to the calling routine for possible display.

During initialization of this routine, the switch s_{proj} is set to zero. After every Lambert solution, tests are made to determine if the transfer angle will lie near the singularity during the maneuver. If transfers near the singularity are detected, the switch s_{proj} is set and the routine is reinitialized, thus 'locking' all subsequent solutions into the premaneuver orbital plane.

To determine if the transfer angle will lie near the singularity during the maneuver, the logic described by Figure 3b of the Detailed Flow Diagram is used. First the switch s_{proj} is checked to see if previous solutions have indicated the transfer will lie near the singularity. If it is not set, then the switch s_{cone}, returned by the Lambert routine, will indicate whether the transfer angle θ at ignition lies in a small cone about the singularity. If s_{cone} is not set, it is still necessary to determine if the transfer angle is likely to move into the region of the singularit y during the maneuver. To accomplish this, the central angle θ_T traversed during the maneuver is estimated by multiplying the approximate maneuver time $(\Delta v/a_T)$ by the approximate orbital rate (v_0/r_0). Then the transfer angle θ_1 remaining at the completion of the maneuver can be tested $(\theta_1 = \theta - \theta_T)$. If θ_1 lies near the singularity, defined by the sine of the half cone angle $\epsilon_{\theta T}$, the switch s_{proj} is set and the routine is reinitialized.

5. DETAILED FLOW DIAGRAM

This section contains a detailed flow diagram of the routine for determining the required velocity and target offset.

Fach input and output variable in the routine and subroutine call statements can be followed by a symbol in brackets. This symbol identifies the notation for the corresponding variable in the detailed description and flow diagrams of the called routine. When identical notation is used, the bracketed symbol is omitted.

9.7.1.2 Precision (continued)

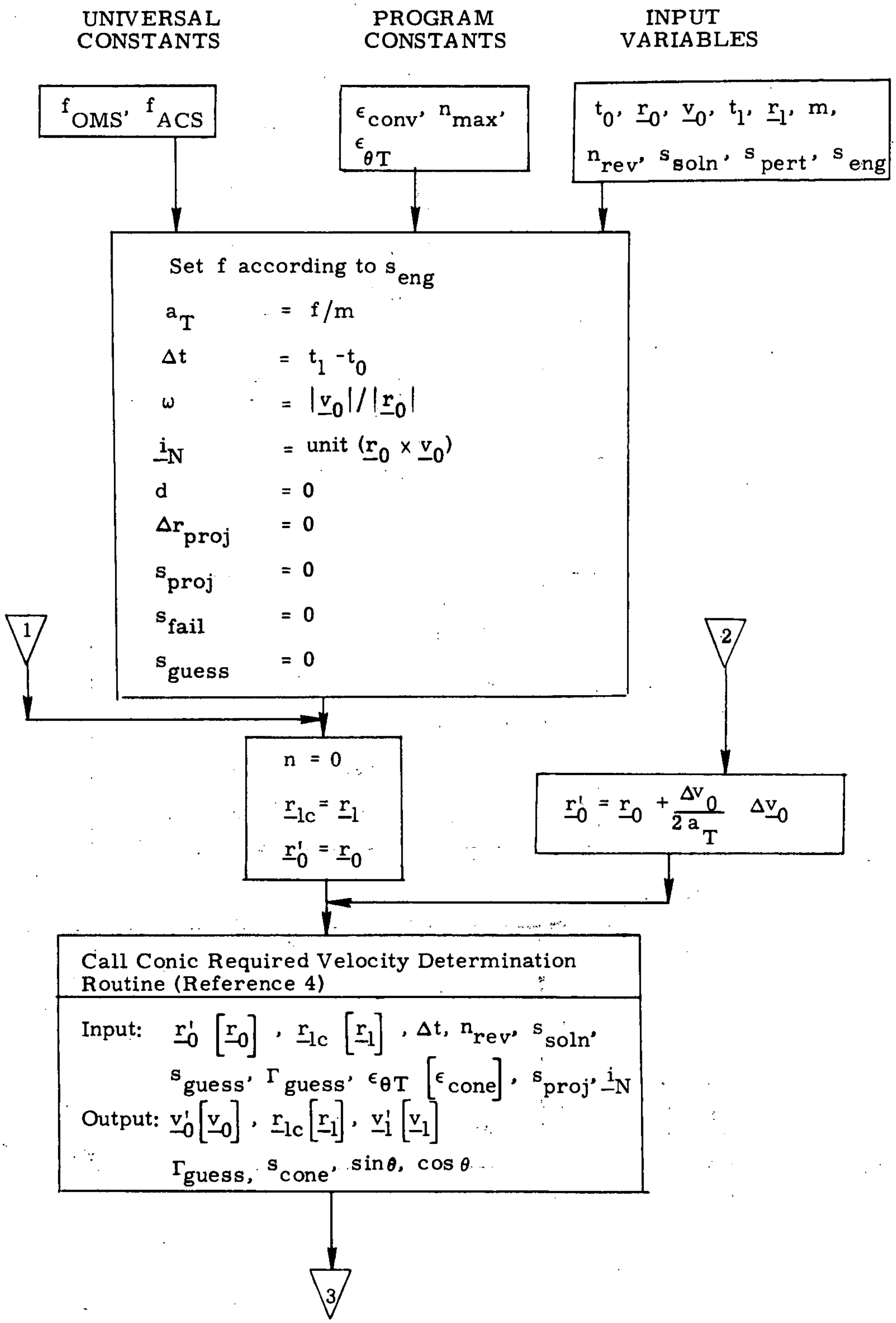

UNIVERSAL CONSTANTS
PROGRAM CONSTANTS
INPUT VARIABLES
f_{OMS}, f_{ACS}
ϵ_{conv}, n_{max}, $\epsilon_{\theta T}$
t_0, $\underline{r}_0$, $\underline{v}_0$, t_1, $\underline{r}_1$, m, n_{rev}, s_{soln}, s_{pert}, s_{eng}
Set f according to s_{eng}
$a_T = f/m$
$\Delta t = t_1 - t_0$
$\omega = |\underline{v}_0| / |\underline{r}_0|$
$\underline{i}_N = \text{unit}(\underline{r}_0 \times \underline{v}_0)$
$d = 0$
$\Delta r_{proj} = 0$
$s_{proj} = 0$
$s_{fail} = 0$
$s_{guess} = 0$
1
2
$n = 0$
$\underline{r}_{1c} = \underline{r}_1$
$\underline{r}'_0 = \underline{r}_0$
$\underline{r}'_0 = \underline{r}_0 + \frac{\Delta v_0}{2 a_T} \Delta\underline{v}_0$
Call Conic Required Velocity Determination Routine (Reference 4)
Input: $\underline{r}'_0 \left[\underline{r}_0\right]$, $\underline{r}_{1c} \left[\underline{r}_1\right]$, Δt, n_{rev}, s_{soln}, s_{guess}, Γ_{guess}, $\epsilon_{\theta T} \left[\epsilon_{cone}\right]$, s_{proj}, $\underline{i}_N$
Output: $\underline{v}'_0 \left[\underline{v}_0\right]$, $\underline{r}_{1c} \left[\underline{r}_1\right]$, $\underline{v}'_1 \left[\underline{v}_1\right]$, Γ_{guess}, s_{cone}, $\sin\theta$, $\cos\theta$
3

Figure 3a. Detailed Flow Diagram

9.7.1.2 Precision (continued)

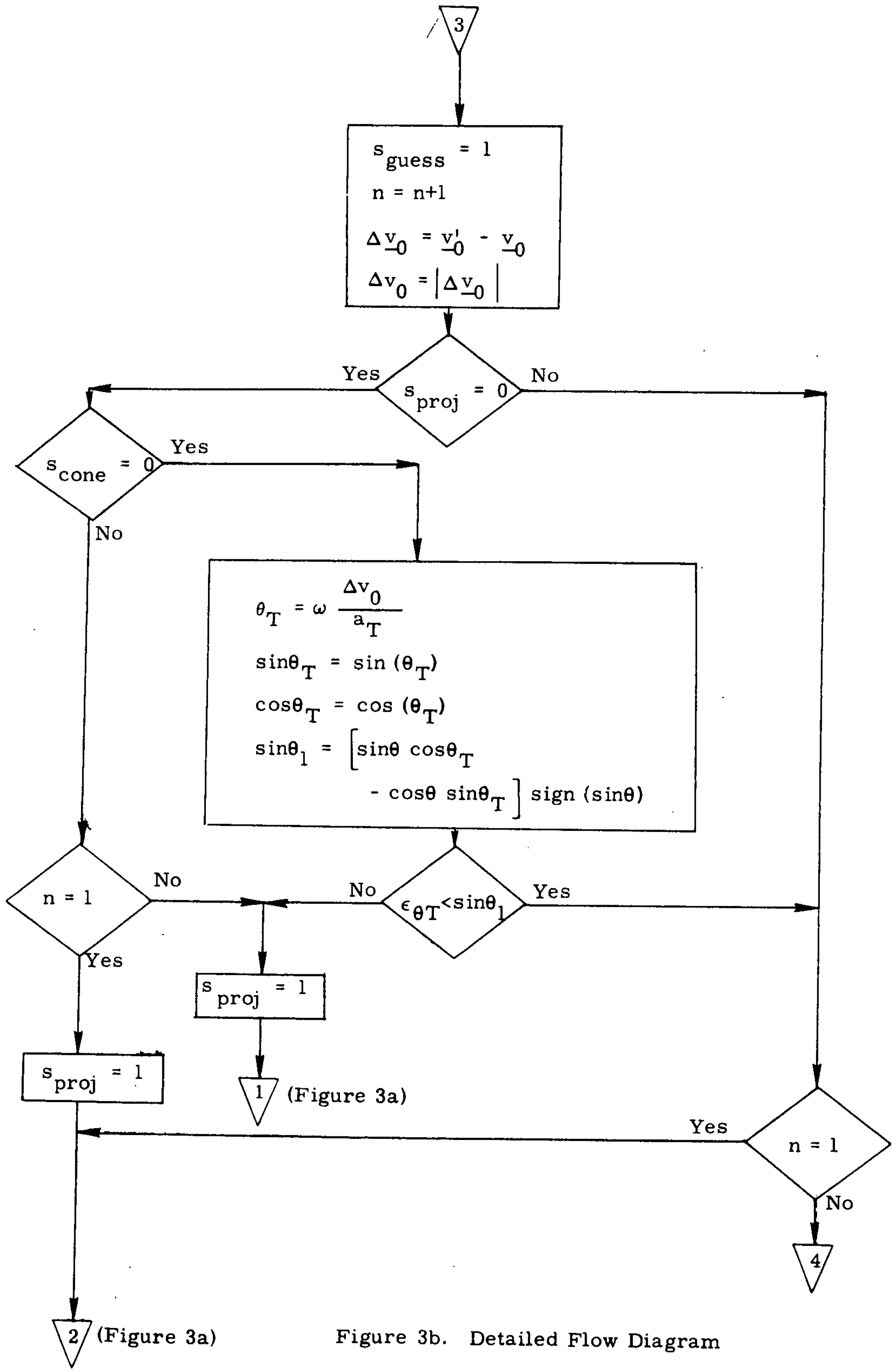

3
$s_{guess} = 1$
$n = n+1$
$\Delta \underline{v}_0 = \underline{v}'_0 - \underline{v}_0$
$\Delta v_0 = |\Delta \underline{v}_0|$
Yes
$s_{proj} = 0$
No
$s_{cone} = 0$
Yes
No
$\theta_T = \omega \frac{\Delta v_0}{a_T}$
$\sin\theta_T = \sin(\theta_T)$
$\cos\theta_T = \cos(\theta_T)$
$\sin\theta_1 = \left[\sin\theta \cos\theta_T - \cos\theta \sin\theta_T\right] \text{sign}(\sin\theta)$
$n = 1$
No
Yes
No
$\epsilon_{\theta T} < \sin\theta_1$
Yes
$s_{proj} = 1$
$s_{proj} = 1$
1
(Figure 3a)
Yes
$n = 1$
No
4
2
(Figure 3a)

Figure 3b. Detailed Flow Diagram

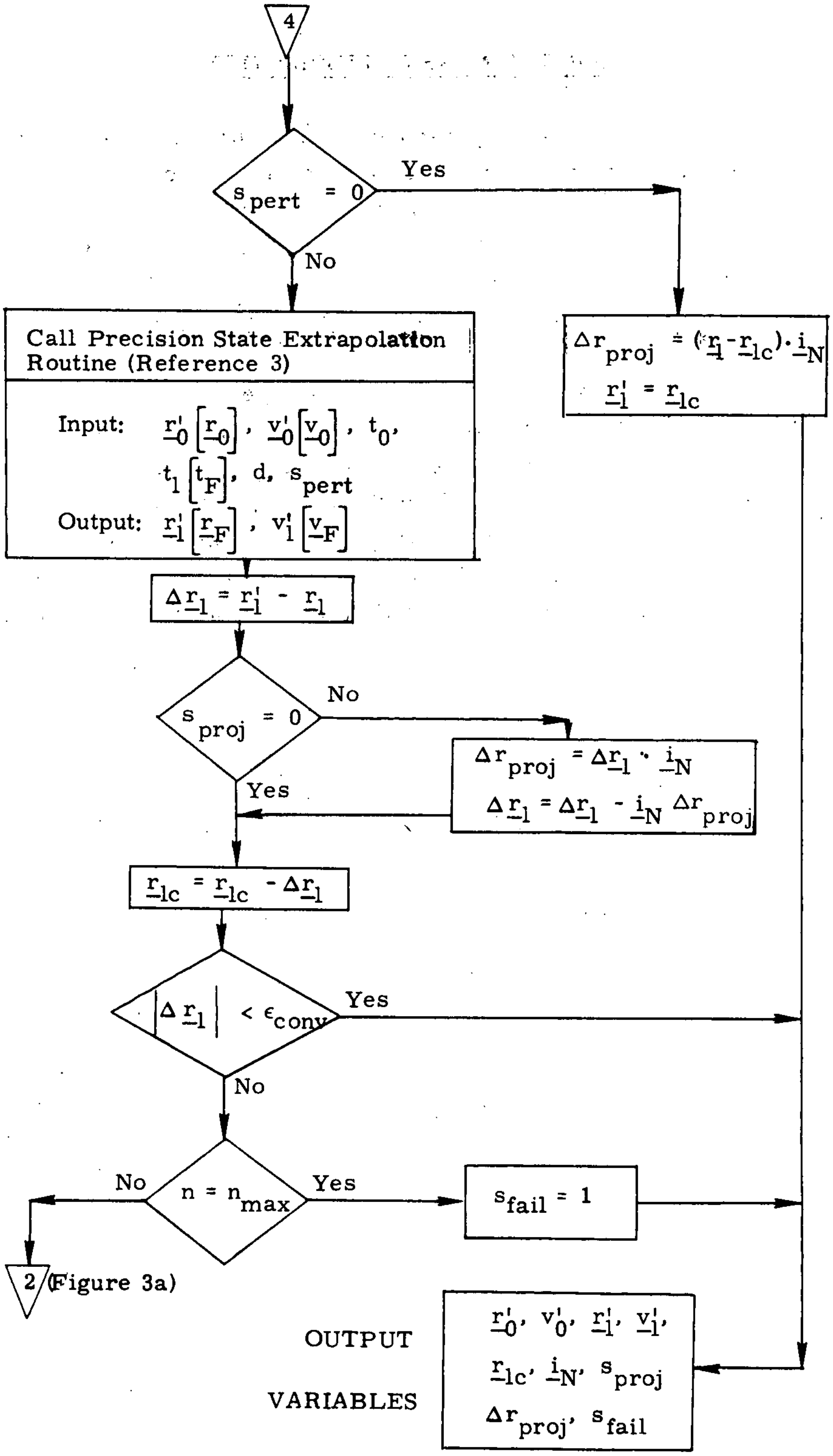

4
$s_{pert} = 0$
Yes
No
Call Precision State Extrapolation Routine (Reference 3)
Input: $\underline{r}'_0 [\underline{r}_0]$, $\underline{v}'_0 [\underline{v}_0]$, t_0, $t_1 [t_F]$, d, s_{pert}
Output: $\underline{r}'_1 [\underline{r}_F]$, $\underline{v}'_1 [\underline{v}_F]$
$\Delta r_{proj} = (\underline{r}_1 - \underline{r}_{1c}) \cdot \underline{i}_N$
$\underline{r}'_1 = \underline{r}_{1c}$
$\Delta \underline{r}_1 = \underline{r}'_1 - \underline{r}_1$
$s_{proj} = 0$
No
Yes
$\Delta r_{proj} = \Delta \underline{r}_1 \cdot \underline{i}_N$
$\Delta \underline{r}_1 = \Delta \underline{r}_1 - \underline{i}_N \Delta r_{proj}$
$\underline{r}_{1c} = \underline{r}_{1c} - \Delta \underline{r}_1$
$|\Delta \underline{r}_1| < \epsilon_{conv}$
Yes
No
$n = n_{max}$
No
Yes
$s_{fail} = 1$
2 (Figure 3a)
OUTPUT VARIABLES
$\underline{r}'_0$, v'_0, $\underline{r}'_1$, $\underline{v}'_1$, $\underline{r}_{1c}$, $\underline{i}_N$, s_{proj}, Δr_{proj}, s_{fail}

Figure 3c. Detailed Flow Diagram

6. SUPPLEMENTARY INFORMATION

The scheme presented here for predicting and compensating for gravitational perturbations is slightly different from the APOLLO design. However, the performance is considerably improved and the ideas are compatible with the Powered Flight Guidance Routines proposed in Reference 2, which also utilize the concept of a coasting trajectory to advantage. The treatment of the 180 degree Lambert singularity has also been improved.

The convergence tolerance is yet to be determined, but should be consistant with orbital navigation accuracy. It should be noted that because of the sequence of computations in this routine, a tolerance of 1000 feet can result in an actual error of about 10 feet since experience has shown that the target miss resulting from numerical integration (and the corresponding incremental improvement in the target offset) decreases by a factor of about 1/100 per iteration. An iteration limit was included since any iterative loop should have some contingency exit in the event of non-convergence.

9.7.1.2 Precision (continued)

REFERENCES

1. Brand, T., "A New Approach to Lambert Guidance", R-694, M.I.T./DL, June 1971.

2. Pu, C., Higgins, J., Brand, T., "Powered Flight Guidance", Space Shuttle GN & C Equation Document No. 11, Revision 1, M.I.T./DL, September 1971.

3. Robertson, W., "Precision State and Filter Weighting Matrix Extrapolation", Space Shuttle GN & C Equation Document No. 4, Revision 1, M.I.T./DL, to be published.

4. Robertson, W., "Conic Required Velocity Determination", Space Shuttle GN & C Equation Document No. 10, Revision 1, M.I.T./DL, September 1971.

9.7.2 Navigation (same as 9.2.1)

9.7.3 Guidance

SPACE SHUTTLE

GN&C SOFTWARE EQUATION SUBMITTAL

Software Equation Section: Powered Flight Guidance

Submittal No. 25A

Function: To issue proper steering and engine cutoff commands so as to satisfy desired maneuver cutoff conditions.

Mosule No. OG2 Function No. 2,4,5 & 6 (MSC 03690)

Submitted by: Pu, Higgins, & Brand Co. MIT No. 11 (Rev. 1)

Date: 21 October 1971

NASA Contract: C. Lively Organization: EG2

Approved by Panel III: K.J. Cox Date: 10/21/71

Summary Description: This routine augments the Apollo External Delta-V Maneuver Guidance Mode and the Apollo Lambert Aim-Point Maneuver Mode with (1) guidance during the maneuver based on navigation in a spherical gravity field, and (2) required velocity determined in consideration of a finite maneuver length. These additions considerably improve the accuracy of the maneuvers. This revision provides (1) equations required to explicity control re-entry angle for deorbit maneuvers and (2) logic to avoid difficulties for transfer angles near 180°.

Comments: P30 (Apollo) External ΔV Maneuver Guidance is included as a subset of this submittal and can be removed from Appendix A of Volume III.

(Design Status) ____________________

(Verification Status) ____________________

Panel Comments:

Revision A: Prior submittal July 1971.

1. INTRODUCTION

The objective of the Powered Flight Guidance Routines is to issue the proper steering and engine cutoff commands such that the desired terminal conditions of the maneuver are satisfied. The basic powered flight guidance law used in the orbiter is a velocity-to-be-gained concept with cross-product steering.

The two principle modes of the Powered Flight Guidance Routines are:

1. Delta-V Maneuver Guidance Mode
2. Real-Time Required Velocity Updating Guidance Mode.

The Delta-V Maneuver Guidance Mode is essentially equivalent to the External Delta-V Maneuver Guidance Mode used in APOLLO. The input desired velocity change is modified to compensate for the estimated central angle to be traversed during the maneuver. Then the object of the powered phase is simply to steer the vehicle to achieve this velocity change.

The Real-Time Required Velocity Updating Mode is a generalized version of the Lambert Aim Point Maneuver Mode used in APOLLO. The object of these maneuvers is to place the vehicle on a coasting trajectory which will intercept a specified target at a specified time. Two new concepts which greatly improve the accuracy of these maneuvers are introduced. First, guidance during the maneuver is based on a state vector navigated from ignition in a spherical (Keplerian) gravity field. Second, the required velocity is not determined using the present vehicle position but rather an offset position which accounts for the finite length of the maneuver. Since this is primarily an equations document, these new concepts are treated only briefly in the text. A detailed description and derivation can be found in Reference 5.

Because the calculation of required velocity can be a lengthy process, the ability to update the required velocity every major cycle is dependent upon the speed of the computer. The APOLLO Guidance Computer required portions of several major cycles to complete the solution. The guidance equations described here will assume that the orbiter computer will also need portions of several major cycles to complete the solution for required velocity. A faster computer would not alter the basic concepts presented here, but would simplify the mechanization somewhat.

The Real-Time Required Velocity Updating Mode may select a specific required velocity routine to accomplish one of the following maneuvers:

1. Lambert Aim Point Maneuver
2. Deorbit Maneuver
3. Other maneuvers such as a maneuver to an orbit with certain specified constraints (TBD).

The required velocity routines will be subjects of separate documents. Since this report is mainly concerned with the documentation of guidance equations, logic or computations concerned with monitoring or controlling system operation will not be presented.

NOMENCLATURE

a_T	Estimated magnitude of thrust acceleration
C	Matrix to rotate the target vector to compensate for earth rotation due to change in time of flight during deorbit maneuver
d	Dimension of navigation filter weighting matrix (d = 0 in this routine since the matrix is not used)
f	Thrust
f_{OMS}	Magnitude of orbital maneuvering system engine thrust
f_{ACS}	Magnitude of attitude control system engine translational thrust
$\underline{g}$	Gravity vector in the oblate gravity field
$\underline{g}_s$	Gravity vector in the spherical gravity field
$\underline{i}_N$	Unit vector in the direction of the angular momentum vector normal to the transfer plane
$\underline{i}_x, \underline{i}_y, \underline{i}_z$	Unit vectors of local vertical coordinates
$\underline{i}_{TD}$	Unit vector of desired thrust direction
k_γ	Sensitivity used in computing the desired change in flight time to control entry angle during deorbit
k_{steer}	Steering gain
k_{tgo}	Intermediate variable in t_{go} computation
m	Current estimated vehicle mass
n	Number of guidance cycles used in thrust acceleration magnitude filter

n_{rev}	Integer number of 360° revolutions used in Conic Required Velocity Determination Routine
p_N	Normalized semi-latus rectum of conic transfer orbit
p_γ	Parameter defining the desired terminal flight path angle
p_γ'	Parameter defining the projected terminal flight path angle
$\underline{r}$	Position vector navigated in the oblate gravity field
$\underline{r}'$	Position vector on the coasting trajectory
$\underline{r}_s$	Position vector navigated in the spherical gravity field
$\underline{r}(t_2)$	Offset target vector at t_2
$\Delta \underline{r}$	Initial position offset
s_{cone}	Switch in the Conic Required Velocity Determination Routine to indicate if the transfer is near 180° (see Ref. 3 for details)
$s_{cut\text{-}off}$	Engine cut-off switch $\begin{pmatrix} = 0 & \text{command not issued} \\ = 1 & \text{command issued} \end{pmatrix}$
s_{guess}	Switch to indicate whether estimate of independent variable Γ will be input to the Conic Required Velocity Determination Routine (see Ref. 3 for details)

s_{eng} Engine select switch

s_{enable} Steering enable switch

$$\left(\begin{array}{ll} = 0 & \text{inhibit} \\ = 1 & \text{enable} \end{array}\right)$$

s_{pert} Switch indicating the perturbing accelerations to be included in Precision State and Filter Weighting Matrix Extrapolation Routine (see Ref. 2 for details)

s_{proj} Switch indicating whether the initial and target position vectors are to be projected into the plane defined by $\underline{i}_N$ (see Ref. 6 for details)

s_{soln} Switch indicating which of two possible solutions is desired in the multi-revolution case (see Ref. 3 for details)

s_{steer} Steering switch

$$\left(\begin{array}{ll} = 0 & \text{no steering} \\ = 1 & \text{active steering permitted} \end{array}\right)$$

t Current state vector time (during thrusting phase, this is the time at which the accelerometers are read)

Δt Guidance cycle time step

$\Delta t'$ Dummy transfer time set to 0

$\Delta t_{cut\text{-}off}$ Value of t_{go} used to define time to issue engine cut-off command and terminate active steering

Δt_{enable} Value of t_{go} which distinguishes between long or short maneuver

Δt_{t0} Time interval before t_{ig} to start thrusting phase computations

Δt_{t1} Time interval prior to t_{ig} when initial t_{go} prediction is made

9.7.3 Guidance (continued)

Δt_{t2} Time interval after t_{ig} when steering is permitted

$\Delta t_{tail\text{-}off}$ Time interval representing the duration of a burn at maximum thrust equivalent to the tail-off impulse after the engine-off signal is issued

$\Delta t_{tail\text{-}off,\ OMS}$ $\Delta t_{tail\text{-}off}$ of orbital maneuvering system engine

$\Delta t_{tail\text{-}off,\ ACS}$ $\Delta t_{tail\text{-}off}$ of attitude control system engine for translational maneuver

δt_2 Change in time of arrival required to satisfy terminal flight path angle in a deorbit maneuver

t_2 Time of arrival at $\underline{r}(t_2)$

t_{go} Time-to-go before engine cut-off

t_{ig} Nominal engine ignition time

$\underline{v}$ Velocity vector navigated in the oblate gravity field

$\underline{v}_s$ Velocity vector navigated in the spherical gravity field

$\underline{v}_g$ Velocity-to-be-gained vector

v_g Magnitude of $\underline{v}_g$

$\underline{v}'_{req}$ Required velocity vector at the offset initial position (defines the coasting trajectory)

$\underline{v}_{req}$ Required velocity at current position (no initial position offset)

$\Delta\underline{v}$ Measured velocity increment vector due to thrust in one guidance cycle

Δv Magnitude of $\Delta\underline{v}$

9.7.3 Guidance (continued)

$\Delta \underline{v}_{LV}$	Input desired velocity change vector to Delta-V Guidance Mode
$\Delta \underline{v}_{xz}$	In-plane components of $\Delta \underline{v}_{LV}$
$\Delta \underline{v}_{c}$	Compensated in-plane components of $\Delta \underline{v}_{LV}$
$\Delta v_x, \Delta v_y,$ Δv_z	Components of $\Delta \underline{v}_{LV}$
v_{exh}	Exhaust velocity
$v_{exh', OMS}$	Exhaust velocity of the orbital maneuvering system engine
$v_{exh', ACS}$	Exhaust velocity of the attitude control system engine for translational maneuver
α_N	Reciprocal of normalized semi-major axis of conic transfer orbit
$\epsilon_{\theta G}$	Tolerance criterion establishing a cone around the negative target position direction inside of which the Conic Required Velocity Determination Routine will define the transfer plane by $\underline{i}_N$
γ_{t2}	Projected terminal flight path angle with respect to local horizontal (negative downward)
Γ	Converged value of iteration variable used in Conic Required Velocity Determination Routine
Γ_P	Previous value of Γ
Γ_{guess}	Estimated value of Γ
$\dot{\Gamma}$	Time rate of change of Γ
κ	Ratio of $\left\lvert \underline{r}(t_2) \right\rvert$ to $\left\lvert \underline{r}(t) \right\rvert$
θ_T	Estimated central angle during thrusting maneuver in Delta-V Guidance Mode
μ	Earth's gravitational constant

9.7.3 Guidance (continued)

τ	Time associated with current required velocity
τ_P	Previous value of τ
$\underline{\omega}_c$	Angular velocity command
ω_{earth}	Magnitude of the earth's angular velocity

2. FUNCTIONAL FLOW DIAGRAM

Powered Flight Guidance involves both the prethrust and thrusting phases of the maneuver. The prethrust computations, shown in Figure 1, are a single step process performed several minutes prior to the maneuver to prepare the vehicle for thrusting. They are required to process targeting parameters to determine the desired vehicle attitude at ignition. In addition, the state vector is advanced to a specified time prior to ignition. At this time, an integral number of major cycles prior to ignition, the thrusting phase computations, including Powered Flight Navigation, are initiated. Of course, the attitude maneuver necessary to align the vehicle to the desired attitude at ignition should be completed before entering the thrusting phase computations.

The sequence of functions performed during the powered flight phase is illustrated in Figure 2. The guidance computer program known as the Servicer Routine, which controls the various subroutines to create a powered flight sequence, is not included in this document.

Each guidance cycle begins with the reading of the accelerometers and is followed by the updating of the state vector in the Powered Flight Navigation Routine. Then the velocity-to-be-gained is updated in the Cross-Product Steering Routine. If steering is permitted, the latter also computes the time-to-go and the steering command beginning at a fixed time after ignition.

The targeting calculations used to predict and compensate for gravitational perturbations establish an offset target which assumes that the vehicle is under the influence of only a spherical gravity field after the expected ignition time. Therefore, in the Real-Time Required Velocity Updating Mode, it is necessary to maintain an additional state vector navigated in a spherical gravity field. This dual navigation should begin at the ignition time assumed in the targeting program if it differs from the actual.

Figure 2 shows the sequencing of the main branch of the guidance routine during the thrusting phase. In the Real-Time Required Velocity Mode, another branch of the Powered Flight Guidance Routines involving the calculation of required velocity operates as a separate branch independent of the main guidance cycle. This separate branch, called the Velocity-to-be-Gained Routine, is initiated and controlled by the Servicer Routine and may require portions of several major guidance cycles to complete its solution. Of course, simple velocity-to-be-gained updates computed by decrementing the previous value by the sensed velocity change continue in the Cross-Product Steering Routine every major cycle. Normally, the Velocity-to-be-Gained Routine operates on a lower priority than the main guidance loop so that the new velocity-to-be-gained vector is not used by the Cross-Product Steering Routine until the next guidance cycle.

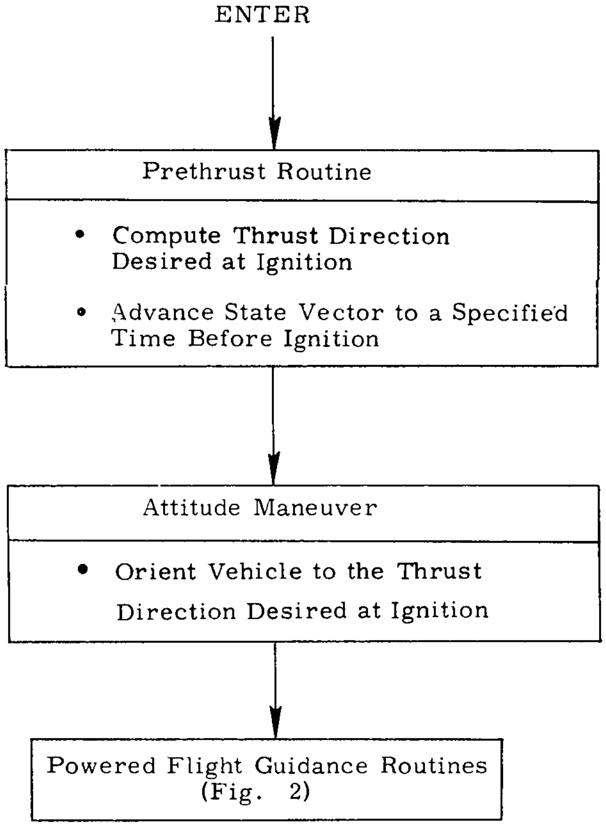

ENTER
Prethrust Routine
• Compute Thrust Direction Desired at Ignition
• Advance State Vector to a Specified Time Before Ignition
Attitude Maneuver
• Orient Vehicle to the Thrust Direction Desired at Ignition
Powered Flight Guidance Routines (Fig. 2)

Figure 1. Powered Flight Program

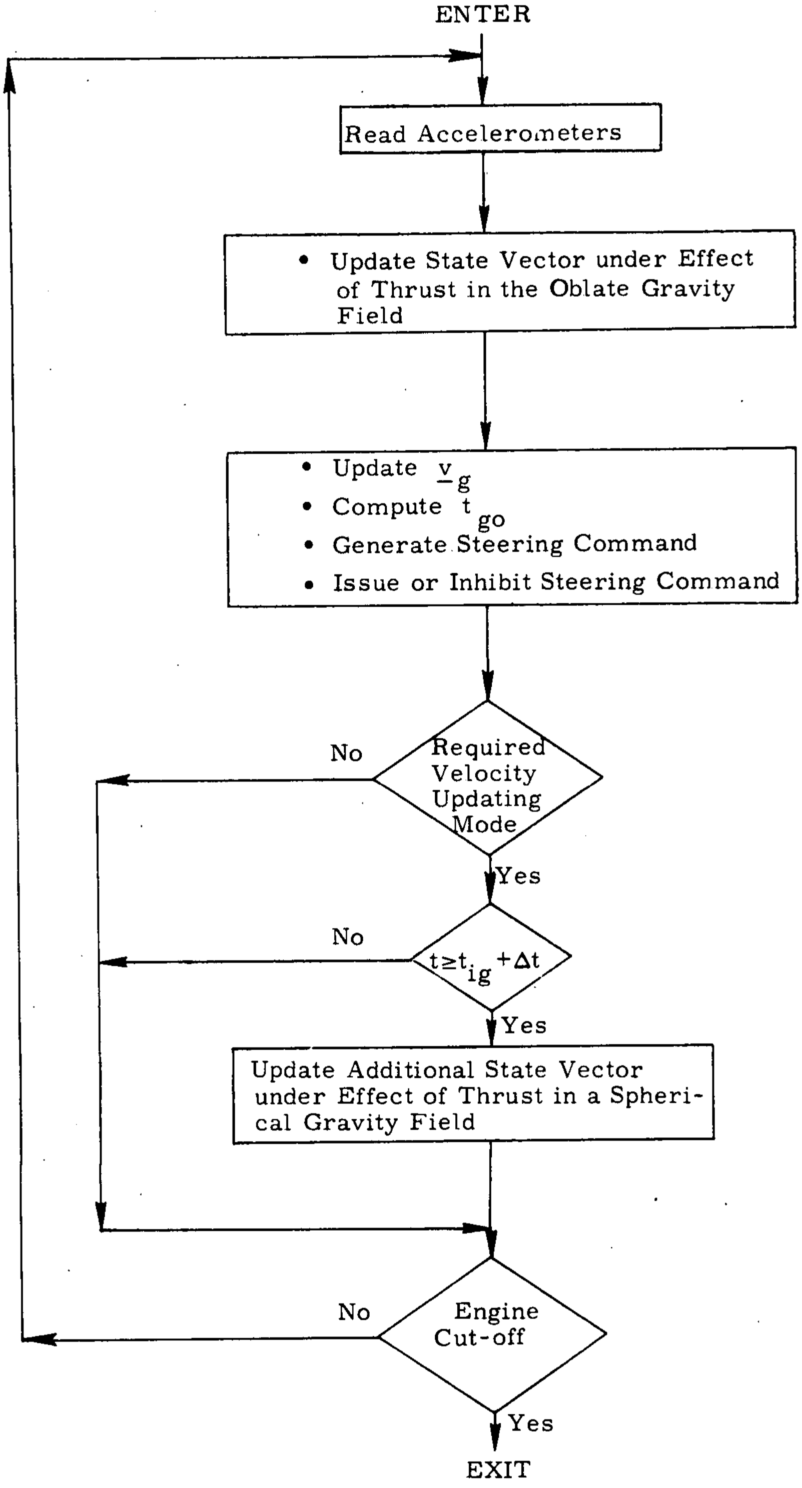

ENTER
Read Accelerometers
• Update State Vector under Effect of Thrust in the Oblate Gravity Field
• Update $\underline{v}_g$
• Compute t_{go}
• Generate Steering Command
• Issue or Inhibit Steering Command
Required Velocity Updating Mode
No
Yes
$t \geq t_{ig} + \Delta t$
No
Yes
Update Additional State Vector under Effect of Thrust in a Spherical Gravity Field
Engine Cut-off
No
Yes
EXIT

Figure 2. Powered Flight Guidance Routines

The characteristic of the transfer in the Real-Time Required Velocity Updating Mode in relation to the singularity cone of the Lambert problem is determined by the targeting program before the powered phase is initiated. This information is passed on to this guidance program through the s_{proj} switch and is used by the Conic Required Velocity Determination Routine to define the transfer plane. (See Ref. 3 for a detailed explanation of the singularity cone and Ref. 6 for the targeting procedure).

If the s_{proj} switch has been set, the transfer will take place in the plane defined by the unit vector $\underline{i}_N$ in the direction of the angular momentum vector at ignition. If this switch has not been set, there are two possibilities. Under normal circumstances the transfer will take place in the plane defined by the vehicle and target position vectors. However, unexpected degradation in engine performance during flight may prolong the powered maneuver to such an extent that the input position vector to the Conic Required Velocity Determination Routine is inside the singularity cone. The procedure to cope with this situation is presented below.

If the s_{proj} switch has not been set by the targeting program, the s_{cone} switch, which is an output of the Conic Required Velocity Determination Routine, is checked at each guidance cycle. If it is found that this switch has been set. indicating that the input position vector is inside the singularity cone, the Servicer Routine is directed to bypass the Velocity-to-be-gained Routine for the remainder of the powered maneuver. In other words, the remaining powered maneuver will be completed simply by decrementing the previous value of the velocity-to-be-gained by the sensed velocity change as is done in the Delta-V Mode.

When the time-to-go becomes less than some predetermined value, active steering is suspended and an engine cut-off command is set to be issued at the proper time.

3. INPUT AND OUTPUT VARIABLES

Input Variables

t Current time of the state vector

$\underline{r}(t)$, $\underline{v}(t)$ Current state vector

t_{ig} Nominal ignition time

m Current estimated vehicle mass

s_{eng} Engine select switch

s_{pert} Switch indicating if certain perturbations should be included in Precision State Extrapolation (see Ref. 2)

Delta-V Mode:

$\Delta\underline{v}_{LV}$ Input desired velocity change vector in local vertical coordinates

Real-Time Required Velocity Updating Mode:

$\underline{i}_N$ Unit vector in the direction of angular momentum normal to the transfer plane

p_γ Parameter defining desired terminal flight path angle in a deorbit maneuver

n_{rev} Integer number of complete 360° revolutions to be made in the desired transfer

$\underline{r}(t_2)$ Offset target vector at time t_2

s_{proj} Switch indicating whether the initial and target position vectors are to be projected into the plane defined by $\underline{i}_N$

s_{soln} Switch indicating which of two physically possible solutions is desired in the multi-revolution case

$\epsilon_{\theta G}$ Tolerance criterion establishing a cone inside which the Lambert routine will define the plane of transfer by $\underline{i}_N$ ($\epsilon_{\theta G}$ = sine of one-half of the cone angle)

Output Variables

$\underline{\omega}_c$ Angular velocity command in inertial coordinates

Engine cut-off Command

4. DESCRIPTION OF EQUATIONS

The Powered Flight Guidance Routines have two major phases, the prethrust phase and the powered flight phase.

4.1 Prethrust Phase

4.1.1 Delta-V Mode

The position and velocity vectors, $\underline{r}(t)$ and $\underline{v}(t)$, are extrapolated to t_{ig} using the Precision State Extrapolation Routine described in Reference 2.

The local vertical coordinate system at ignition is defined by the unit vectors as follows:

$$\underline{i}_x = \text{unit} \left\{ [\, \underline{r}(t_{ig}) \times \underline{v}(t_{ig}) \,] \times \underline{r}(t_{ig}) \right\}$$

$$\underline{i}_y = \text{unit} \; [\, \underline{v}(t_{ig}) \times \underline{r}(t_{ig}) \,]$$

$$\underline{i}_z = -\,\text{unit} \; [\, \underline{r}(t_{ig}) \,] .$$

The Prethrust Routine accepts an input $\Delta\underline{v}_{LV}$ in local vertical coordinates

$$\Delta\underline{v}_{LV} = (\Delta v_x, \Delta v_y, \Delta v_z).$$

The in plane components of $\Delta\underline{v}_{LV}$ in inertial coordinates are given by

$$\Delta\underline{v}_{xz} = \Delta v_x \, \underline{i}_x + \Delta v_z \, \underline{i}_z .$$

The estimated central angle traversed by the vehicle during the powered maneuver is computed from

$$\theta_T = \frac{\left| \underline{r}(t_{ig}) \times \underline{v}(t_{ig}) \right| \Delta v_{LV} \; m}{r^2(t_{ig}) \; f} .$$

The in plane components of $\Delta \underline{v}_{LV}$ are then rotated by half of the estimated central angle, as shown in Figure 3, to give

$$\Delta \underline{v}_c = \Delta v_{xz} \left[\text{unit } (\Delta \underline{v}_{xz}) \cos\left(\frac{\theta_T}{2}\right) + \text{unit } (\Delta \underline{v}_{xz} X \, \underline{i}_y) \sin\left(\frac{\theta_T}{2}\right) \right].$$

The total compensated velocity-to-be-gained vector at t_{ig} is

$$\underline{v}_g (t_{ig}) = \Delta \underline{v}_c + \Delta v_y \underline{i}_y$$

and the unit vector of the desired thrust direction is

$$\underline{i}_{TD} = \text{unit } [\underline{v}_g (t_{ig})] .$$

4. 1. 2 Real-Time Required Velocity Updating Mode

The following discussion of the prethrust computations necessary in the Real-Time Required Velocity Updating Mode introduces the concept of defining the required velocity on a coasting trajectory. Since a similar set of equations and a more complete description can be found in the discussion of the Velocity-to-be-Gained Routine, only a brief description will be included here.

As is done in the Delta-V Mode, the first step in the prethrust process is the extrapolation of the position and velocity vectors, $\underline{r}(t)$ and $\underline{v}(t)$, to the ignition time, t_{ig}, using the Precision State Extrapolation Routine (Ref. 2). The Conic Required Velocity Determination Routine (Ref. 3) is called to compute the required velocity at ignition, $\underline{v}_{req} (t_{ig})$. An estimate of the velocity-to-be-gained vector at ignition, $\underline{v}_g (t_{ig})$, based on an impulsive maneuver, is obtained by

$$\underline{v}_g (t_{ig}) = \underline{v}_{req} (t_{ig}) - \underline{v} (t_{ig}) .$$

This estimate of velocity-to-be-gained is then used to compute an initial position offset, $\Delta \underline{r} (t_{ig})$, which is an estimate of the position deviation of a coasting trajectory from the finite-thrust trajectory. This coasting trajectory is coincident with the finite-thrust trajectory at thrust termination.

A prethrust estimate of the thrust acceleration, also required to compute the offset, is given by

$$a_T = \frac{f}{m} ,$$

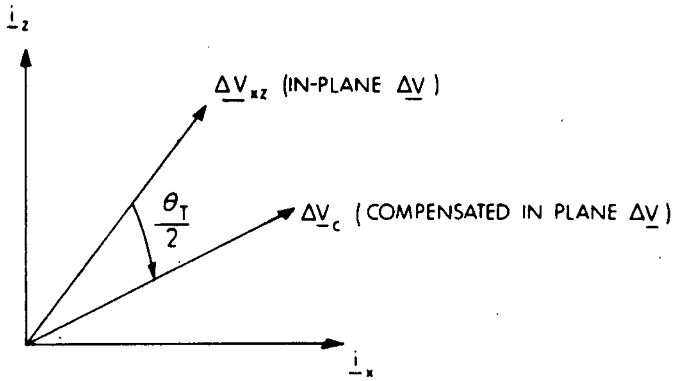

$\underline{i}_z$
$\Delta \underline{V}_{xz}$ (IN-PLANE $\Delta \underline{V}$)
$\frac{\theta_T}{2}$
$\Delta \underline{V}_c$ (COMPENSATED IN PLANE $\Delta \underline{V}$)
$\underline{i}_x$

Figure 3. Compensation of In-plane Components of Desired Velocity Change for Estimated Central Angle in Delta-V Maneuver Guidance Mode

where f and m are the nominal thrust and current estimated vehicle mass.

Using the velocity-to-be-gained and thrust acceleration estimates, the initial position offset is defined by

$$\Delta \underline{r}(t_{ig}) = -\frac{|\underline{v}_g(t_{ig})|}{2\ a_T}\ \underline{v}_g(t_{ig})$$

and the initial position of the coasting trajectory, $\underline{r}'(t_{ig})$, is simply

$$\underline{r}'(t_{ig}) = \underline{r}(t_{ig}) + \Delta \underline{r}(t_{ig}).$$

The Conic Required Velocity Determination Routine is again called to compute the required velocity, $\underline{v}'_{req}(t_{ig})$, from the offset ignition position. Then the final value of velocity-to-be-gained and desired thrust direction at ignition are given by

$$\underline{v}_g(t_{ig}) = \underline{v}'_{req}(t_{ig}) - \underline{v}(t_{ig})$$

$$\underline{i}_{TD} = \text{unit}\ [\ \underline{v}_g(t_{ig})\ ]\ .$$

The final step in the prethrust phase is the extrapolation of the state vector to a predetermined time $(t_{ig} - \Delta t_{t0})$ prior to ignition. This is the time at which the thrusting phase calculations, including powered flight navigation, are initiated.

4.2 Powered Flight Phase

4.2.1 Time-to-go Prediction

An initial estimate of the duration of engine thrust is required to determine if active steering is allowed. The computation is done at a predetermined time before ignition using $\underline{v}_g(t_{ig})$, and estimated mass and mass flow rate. Any correction in $\underline{v}_g$ due to ullage prior to this time will be accounted for in the Cross-Product Steering Routine. It is only necessary to correct $\underline{v}_g$ for ullage from this time on. The corrected $\underline{v}_g$ is used to determine certain parameters so that the maneuver may be classified as either short or long. Active steering is inhibited $(s_{enable} = 0)$ for short maneuver. In this case, t_{go} is computed and a cut-off command is issued to take place at $t_{ig} + t_{go}$. Active steering is permitted $(s_{enable} = 1)$ for a long maneuver and t_{go} is not computed.

Detailed t_{go} computation for short burn will be given at a later date when engine characteristics become available.

4.2.2 Acceleration Magnitude Filter

In order to compute an initial position offset vector $\Delta \underline{r}(t)$ in the Velocity-To-Be-Gained Routine, an estimate of the thrust acceleration a_T is required. During the prethrust phase, an estimate of a_T is given by

$$a_T = \frac{f}{m} .$$

This value is also used during the powered flight phase until several guidance cycles after actual engine ignition. From that time on, a_T is computed by an averaging process over a number of guidance cycles, i.e.,

$$a_T = \sum_{i=1}^{n} \frac{|\Delta \underline{v}_i|}{n \; \Delta t} .$$

The minimum number of guidance cycles, n, required to smooth out fluctuation is to be determined.

4.2.3 Cross-Product Steering Routine

The first function of the Cross-Product Steering Routine is to update the velocity-to-be-gained vector by subtracting the measured velocity change, $\Delta \underline{v}$.

$$\underline{v}_g(t) = \underline{v}_g(t - \Delta t) - \Delta \underline{v}$$

In the Delta-V Mode, the $\underline{v}_g(t - \Delta t)$ is the $\underline{v}_g$ of the previous guidance cycle. However, in the Real-Time Required Velocity Updating Mode, this term may have been updated by the Velocity-to-be-Gained Routine in the previous guidance cycle.

If active steering is enabled ($s_{enable} = 1$), the steering switch s_{steer} is set to 1 to permit active steering at a specified time after actual engine ignition and reset to 0 when t_{go} becomes less than some predetermined value, $\Delta t_{cut\text{-}off}$. When $s_{steer} = 1$, t_{go} is computed by the equation

$$t_{go} = k_{tgo} \left| \frac{\underline{v}_g \cdot \Delta \underline{v}}{\Delta \underline{v} \cdot \Delta \underline{v}} \right| \Delta t + \Delta t_{tail\text{-}off} ,$$

where

$$k_{tgo} = 1 - \frac{1}{2} \frac{|\underline{v}_g \cdot \Delta \underline{v}|}{v_{exh} \quad \Delta v} .$$

The steering command computed is an angular velocity command in inertial coordinates and is given by

$$\underline{\omega}_c = - k_{steer} \frac{\underline{v}_g \times \Delta \underline{v}}{v_g \quad \Delta v} .$$

It may be required that the actual steering command to the autopilot be in the navigation base coordinates in which case a coordinate transformation of $\underline{\omega}_c$ will be required.

If active steering is inhibited ($s_{enable} = 0$), the steering switch remains at $s_{steer} = 0$. In this case, there is no t_{go} computation or active steering.

An engine off command is issued when t_{go} becomes less than $\Delta t_{cut\text{-}off}$, and active steering is terminated at that time.

4.2.4 Velocity-to-be-Gained Routine

The Velocity-to-be-Gained Routine is entered every major guidance cycle after completion of the Cross-Product Steering Routine. Since the calculation of required velocity via a solution of Lambert's problem in the Conic Required Velocity Determination Routine is an iterative process, it may not be completed in one major guidance cycle. If this is the case, when it is recalled by the Servicer Routine, it is entered at the point from which it was exited in the previous guidance cycle.

Since the computations in this routine may require more than the time available in one major guidance cycle, it is necessary to distinguish between the current state vector time t, which is updated every major cycle, and the time associated with the solution for required velocity. Thus the time reference, τ, is used to define the time associated with the current (in process) solution for required velocity.

The targeting calculations completed prior to the thrusting maneuver determine an offset target which assumes that the vehicle is under the influence of spherical gravity after ignition. Therefore, the state vector input to the Velocity-to-be-Gained Routine is navigated in a spherical gravity field after t_{ig}, the ignition time assumed by the targeting program. Throughout this section, reference to the vehicle's state vector will imply the spherically navigated one unless specifically stated otherwise.

The position vector used as the initial condition for the Lambert solution is offset from the actual position to account for the non-impulsive nature of the maneuver. A graphical description of this guidance concept is shown in Figure 4. The coasting trajectory, with the offset initial position, is shown by ① ; ② is the powered trajectory (spherical gravity field); and ③ is the actual trajectory (oblate gravity field). The offset initial position is selected such that the resulting coasting trajectory, defined by the Lambert required velocity, will be coincident with the powered trajectory at thrust termination. The velocity-to-be-gained is defined as the difference between the (required) velocity on the coasting trajectory and the velocity on the powered trajectory. As the velocity-to-be-gained is driven to zero, the powered trajectory approaches the coasting trajectory. The powered trajectory will then follow a path to intercept the offset target while, at the same time, the actual trajectory will follow a path to intercept the true target.

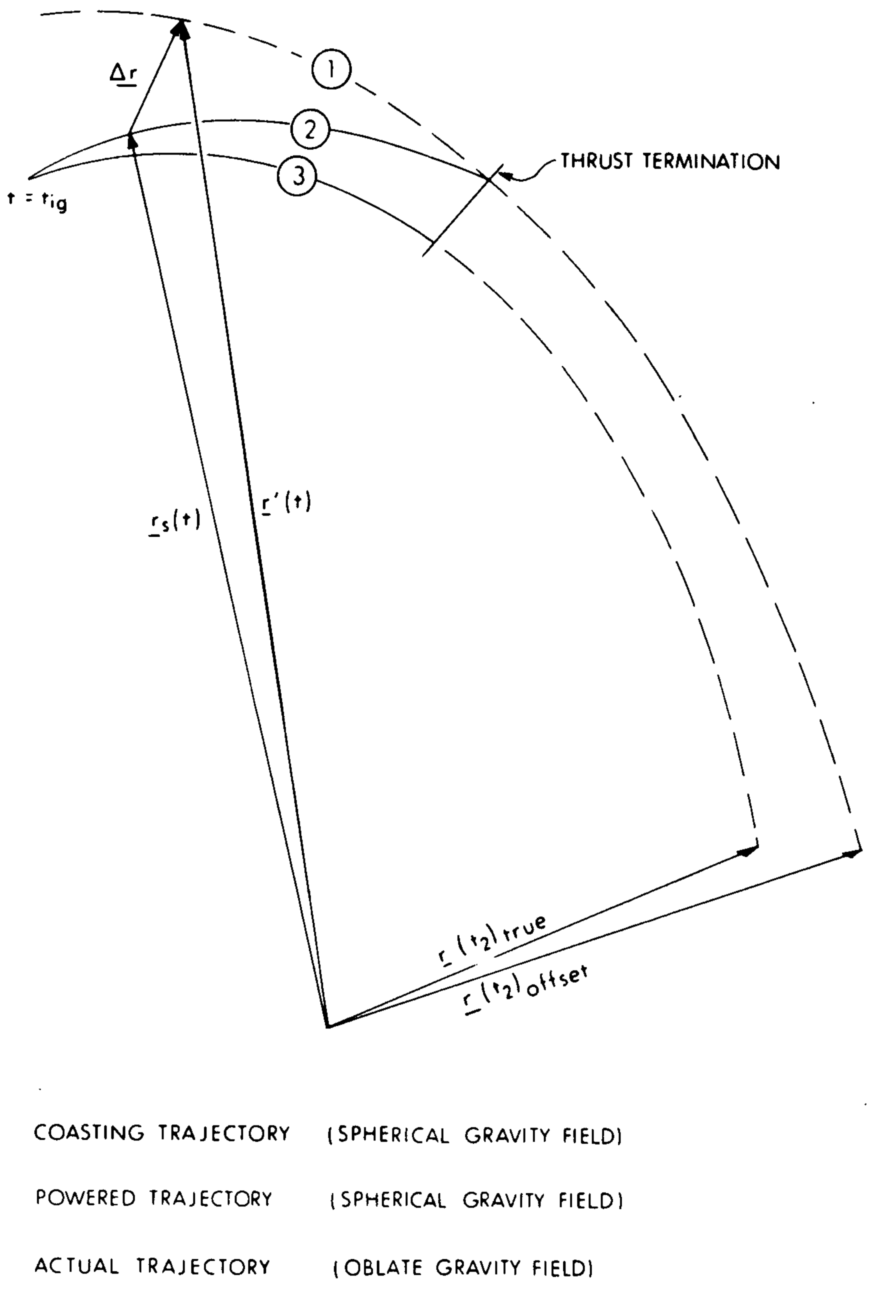
The initial position difference, $\Delta \underline{r}$, between the coasting and powered trajectories can be computed by the following equation:

$$\Delta \underline{r} = - \frac{|\underline{v}_g(\tau)|}{2\, a_T(\tau)} \; \underline{v}_g(\tau) ,$$

where $\underline{v}_g(\tau)$ is the extrapolated value of the velocity-to-be-gained vector computed in the Cross-Product Steering Routine, and $a_T(\tau)$ is the current estimate of the thrust acceleration magnitude. The offset position vector, $\underline{r}'(\tau)$, is then computed from

$$\underline{r}'(\tau) = \underline{r}_s(\tau) + \Delta \underline{r} \; ,$$

where $\underline{r}_s(\tau)$ is the position vector on the powered trajectory. Using this offset position vector, a Lambert solution for the required velocity on the coasting trajectory is obtained from the Conic Required Velocity Determination Routine.



1
2
3
Δr
THRUST TERMINATION
t = t_ig
r_s(t)
r'(t)
r(t_2) true
r(t_2) offset
1 COASTING TRAJECTORY (SPHERICAL GRAVITY FIELD)
2 POWERED TRAJECTORY (SPHERICAL GRAVITY FIELD)
3 ACTUAL TRAJECTORY (OBLATE GRAVITY FIELD)

Figure 4. Graphical Description of Trajectories used in Real-Time Required Velocity Updating Guidance Mode

If the Lambert solution for required velocity cannot be completed in one guidance cycle, the required velocity must be extrapolated forward before computing the velocity-to-be-gained. Since the required velocity is defined on a coasting trajectory, only the spherical gravitational acceleration need be considered. First the current state vector time, t, must be obtained. Then

$$\underline{v}'_{req}(t) = \underline{v}'_{req}(\tau) + \frac{1}{2}(t - \tau)[\underline{g}_s(t) + \underline{g}_s(\tau)]$$

where $\underline{g}_s(t)$ and $\underline{g}_s(\tau)$ are the gravity vectors on the powered trajectory at t and τ, respectively. Note that the required velocity is extrapolated forward by numerically integrating the average gravitational acceleration on the powered trajectory rather than on the coasting trajectory. This approximation, which eliminates the need to compute the gravitational acceleration on the powered trajectory, can be justified for two reasons: first, the extrapolation error does not accumulate over the entire maneuver due to the repetitive nature of the guidance scheme; second, towards the end of the maneuver when accuracy becomes important, the gravitational difference between the powered and coasting trajectories becomes insignificant.

After the required velocity has been extrapolated to the time associated with the current velocity vector, the velocity-to-be-gained can be computed from the following equation

$$\underline{v}_g(t) = \underline{v}'_{req}(t) - \underline{v}_s(t) .$$

The Conic Required Velocity Determination Routine has a mode in which a guess of the independent variable, Γ, can be input to the routine in order to reduce the number of iterations required to solve Lambert's problem. The iterated value of the independent variable is then returned on the output list so that it can be used to determine a guess for the next Lambert solution. The Velocity-to-be-Gained Routine utilizes this mode. It uses the last two iterated values of the independent variable to compute the time derivative of the independent variable. This derivative is used to extrapolate the independent variable to the time associated with the next Lambert solution. This extrapolated value is then input to the Conic Required Velocity Determination Routine as the new guess of the independent variable.

4.2.5 Deorbit Targeting Modification

During a deorbit maneuver degradation in engine performance is compensated by modifying the time of arrival so that the desired terminal flight path angle is maintained. Beginning with the second pass through the Velocity-to-be-gained Routine the desired change in the time of arrival is computed in the following manner.

First, a parameter p_γ', which is a function of the projected terminal (entry interface) flight path angle, is computed by the relation

$$p_\gamma' \triangleq \sec^2 (\gamma_{t_2})$$

$$= \left[2 - \alpha_N \kappa\right] \frac{\kappa}{p_N}$$

where

γ_{t_2} = Projected terminal flight path angle with respect to local horizontal (negative downward)

$\underline{r}(t_2)$ = Target position vector

$\underline{r}(t)$ = Current vehicle position vector tion vector

$$\kappa = \frac{|\underline{r}(t_2)|}{|\underline{r}(t)|}$$

α_N, p_N = Outputs of Conic Required Velocity Determination Routine (see Ref. 3)

The desired change in the time of arrival is then given by

$$\delta t_2 = k_\gamma \left[p_\gamma' - p_\gamma\right] (t_2 - t)$$

where

k_γ = Sensitivity used to compute δt_2

p_γ = Desired value of $\sec^2 (\gamma_{t_2})$

p_γ is an input variable. k_γ is a predetermined constant which is an approximation to the partial derivative of the change in the time of arrival with respect to the change in $\sec^2 (\gamma_{t_2})$ divided by the time of flight to entry interface. Simulations have shown that its value is nearly constant for typical deorbit maneuvers.

Next, the inertial target vector is rotated to compensate for target movement due to earth rotation and change in the time of arrival.

$$C = \begin{bmatrix} \cos(\omega_{earth} \; \delta t_2) & \sin(\omega_{earth} \; \delta t_2) & 0 \\ -\sin(\omega_{earth} \; \delta t_2) & \cos(\omega_{earth} \; \delta t_2) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\underline{r}(t_2 + \delta t_2) = C \; \underline{r}(t_2)$$

The time of arrival is updated by

$$t_2 = t_2 + \delta t_2$$

The updated target vector and time of arrival are used in the Velocity-to-be-gained Routine in the next cycle.

4.3 Sequence of Events

To aid the reader in understanding the sequence of events which make up a powered maneuver, a chronological list is included in Figure 5.

C.5

9.7.3 Guidance (continued)

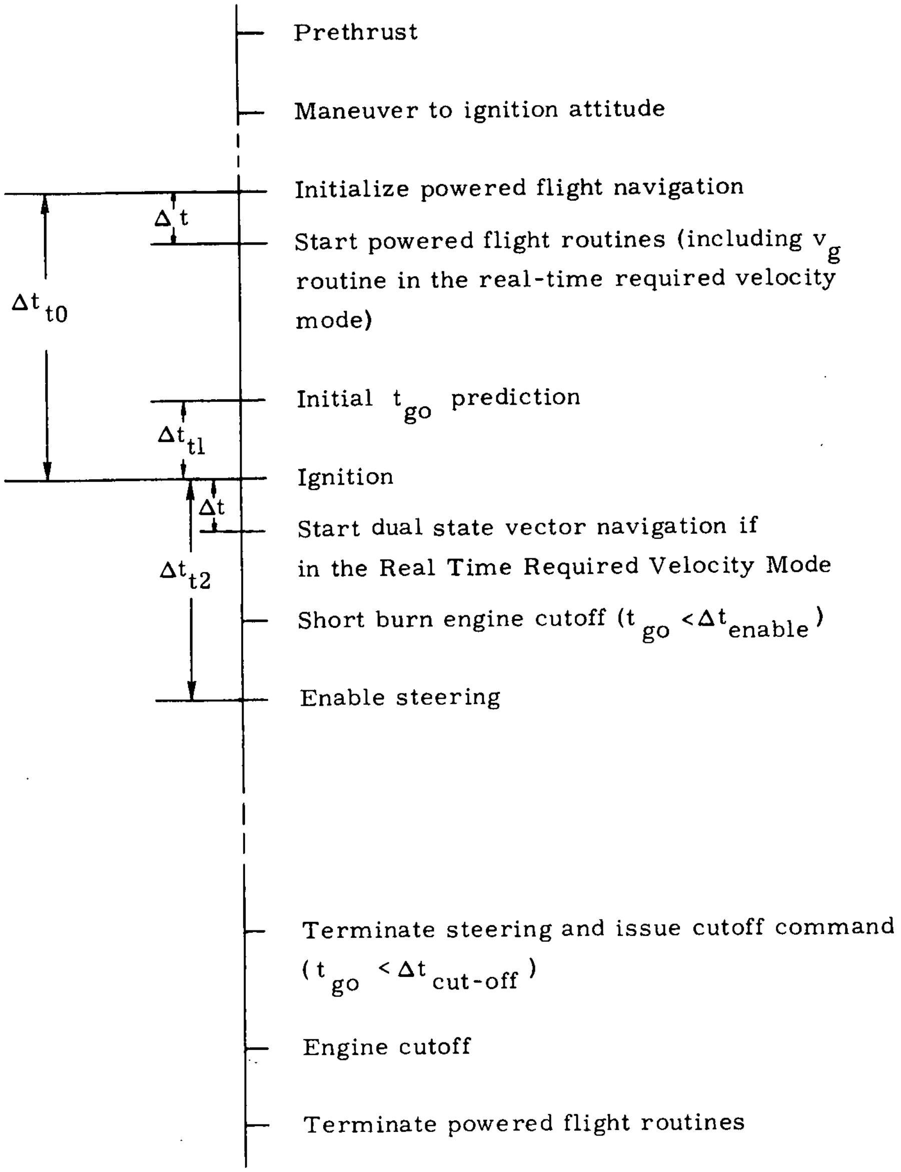

Prethrust
Maneuver to ignition attitude
Initialize powered flight navigation
$\Delta' t$
Start powered flight routines (including v_g routine in the real-time required velocity mode)
Δt_{t0}
Initial t_{go} prediction
Δt_{t1}
Ignition
Δt
Start dual state vector navigation if in the Real Time Required Velocity Mode
Δt_{t2}
Short burn engine cutoff ($t_{go} < \Delta t_{enable}$)
Enable steering
Terminate steering and issue cutoff command ($t_{go} < \Delta t_{cut\text{-}off}$)
Engine cutoff
Terminate powered flight routines

Figure 5. Sequence of Events

5. DETAILED FLOW DIAGRAMS

This section contains detailed flow diagrams of the routines used for powered flight guidance.

Each input and output variable in the routine and subroutine call statements can be followed by a symbol in brackets. This symbol identifies the notation in the corresponding variable in the detailed description and flow diagrams of the called routine. When identical notation is used, the bracketed symbol is omitted.

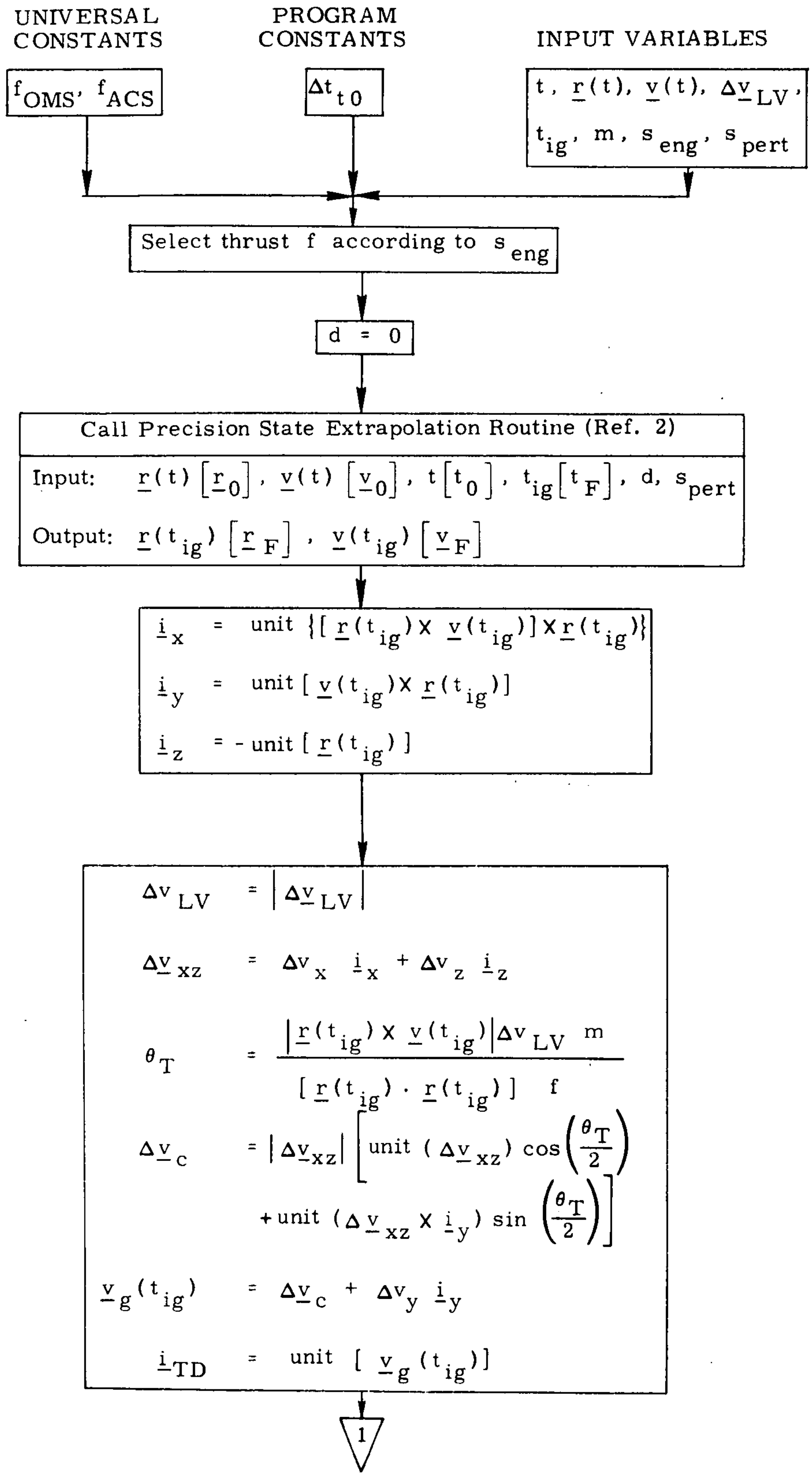

UNIVERSAL CONSTANTS
f_{OMS}, f_{ACS}
PROGRAM CONSTANTS
Δt_{t0}
INPUT VARIABLES
$t, \underline{r}(t), \underline{v}(t), \Delta\underline{v}_{LV}, t_{ig}, m, s_{eng}, s_{pert}$
Select thrust f according to s_{eng}
$d = 0$
Call Precision State Extrapolation Routine (Ref. 2)
Input: $\underline{r}(t)\left[\underline{r}_0\right], \underline{v}(t)\left[\underline{v}_0\right], t\left[t_0\right], t_{ig}\left[t_F\right]$, d, s_{pert}
Output: $\underline{r}(t_{ig})\left[\underline{r}_F\right], \underline{v}(t_{ig})\left[\underline{v}_F\right]$
$\underline{i}_x = \text{unit}\,\{[\underline{r}(t_{ig}) \times \underline{v}(t_{ig})] \times \underline{r}(t_{ig})\}$
$\underline{i}_y = \text{unit}\,[\underline{v}(t_{ig}) \times \underline{r}(t_{ig})]$
$\underline{i}_z = -\,\text{unit}\,[\underline{r}(t_{ig})]$
$\Delta v_{LV} = \left|\Delta\underline{v}_{LV}\right|$
$\Delta\underline{v}_{xz} = \Delta v_x\,\underline{i}_x + \Delta v_z\,\underline{i}_z$
$\theta_T = \dfrac{\left|\underline{r}(t_{ig}) \times \underline{v}(t_{ig})\right| \Delta v_{LV}\, m}{[\underline{r}(t_{ig}) \cdot \underline{r}(t_{ig})]\; f}$
$\Delta\underline{v}_c = \left|\Delta\underline{v}_{xz}\right| \left[\text{unit}\,(\Delta\underline{v}_{xz}) \cos\left(\dfrac{\theta_T}{2}\right) + \text{unit}\,(\Delta\underline{v}_{xz} \times \underline{i}_y) \sin\left(\dfrac{\theta_T}{2}\right)\right]$
$\underline{v}_g(t_{ig}) = \Delta\underline{v}_c + \Delta v_y\,\underline{i}_y$
$\underline{i}_{TD} = \text{unit}\,[\underline{v}_g(t_{ig})]$
1

Figure 6a. Prethrust Phase of Delta-V Maneuver Guidance Mode

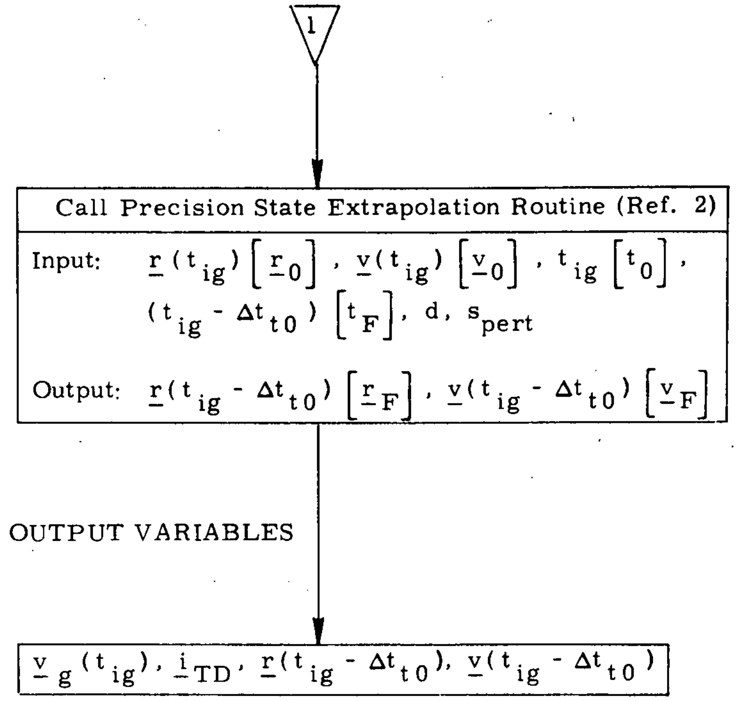

1
Call Precision State Extrapolation Routine (Ref. 2)
Input: $\underline{r}(t_{ig}) \left[\underline{r}_0\right]$, $\underline{v}(t_{ig}) \left[\underline{v}_0\right]$, $t_{ig} \left[t_0\right]$, $(t_{ig} - \Delta t_{t0}) \left[t_F\right]$, d, s_{pert}
Output: $\underline{r}(t_{ig} - \Delta t_{t0}) \left[\underline{r}_F\right]$, $\underline{v}(t_{ig} - \Delta t_{t0}) \left[\underline{v}_F\right]$
OUTPUT VARIABLES
$\underline{v}_g(t_{ig})$, $\underline{i}_{TD}$, $\underline{r}(t_{ig} - \Delta t_{t0})$, $\underline{v}(t_{ig} - \Delta t_{t0})$

Figure 6b. Prethrust Phase of Delta-V Maneuver Guidance Mode

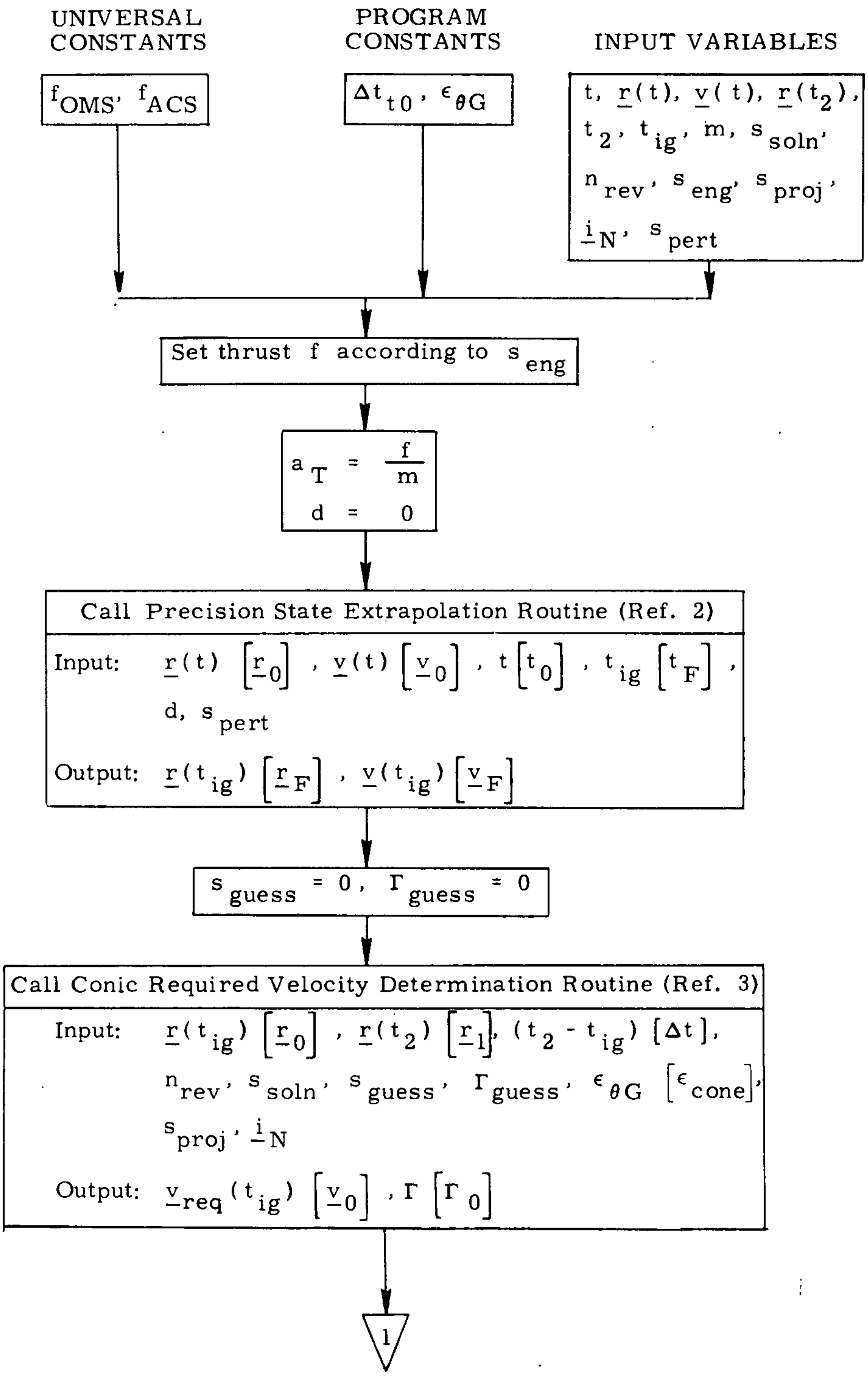

UNIVERSAL CONSTANTS
f_{OMS}, f_{ACS}
PROGRAM CONSTANTS
Δt_{t0}, $\epsilon_{\theta G}$
INPUT VARIABLES
t, $\underline{r}(t)$, $\underline{v}(t)$, $\underline{r}(t_2)$, t_2, t_{ig}, m, s_{soln}, n_{rev}, s_{eng}, s_{proj}, $\underline{i}_N$, s_{pert}
Set thrust f according to s_{eng}
$a_T = \frac{f}{m}$
$d = 0$
Call Precision State Extrapolation Routine (Ref. 2)
Input: $\underline{r}(t)$ $[\underline{r}_0]$, $\underline{v}(t)$ $[\underline{v}_0]$, t $[t_0]$, t_{ig} $[t_F]$, d, s_{pert}
Output: $\underline{r}(t_{ig})$ $[\underline{r}_F]$, $\underline{v}(t_{ig})$ $[\underline{v}_F]$
$s_{guess} = 0$, $\Gamma_{guess} = 0$
Call Conic Required Velocity Determination Routine (Ref. 3)
Input: $\underline{r}(t_{ig})$ $[\underline{r}_0]$, $\underline{r}(t_2)$ $[\underline{r}_1]$, $(t_2 - t_{ig})$ $[\Delta t]$, n_{rev}, s_{soln}, s_{guess}, Γ_{guess}, $\epsilon_{\theta G}$ $[\epsilon_{cone}]$, s_{proj}, $\underline{i}_N$
Output: $\underline{v}_{req}(t_{ig})$ $[\underline{v}_0]$, Γ $[\Gamma_0]$
1

Figure 7a. Prethrust Phase of Real-Time Required Velocity Updating Guidance Mode

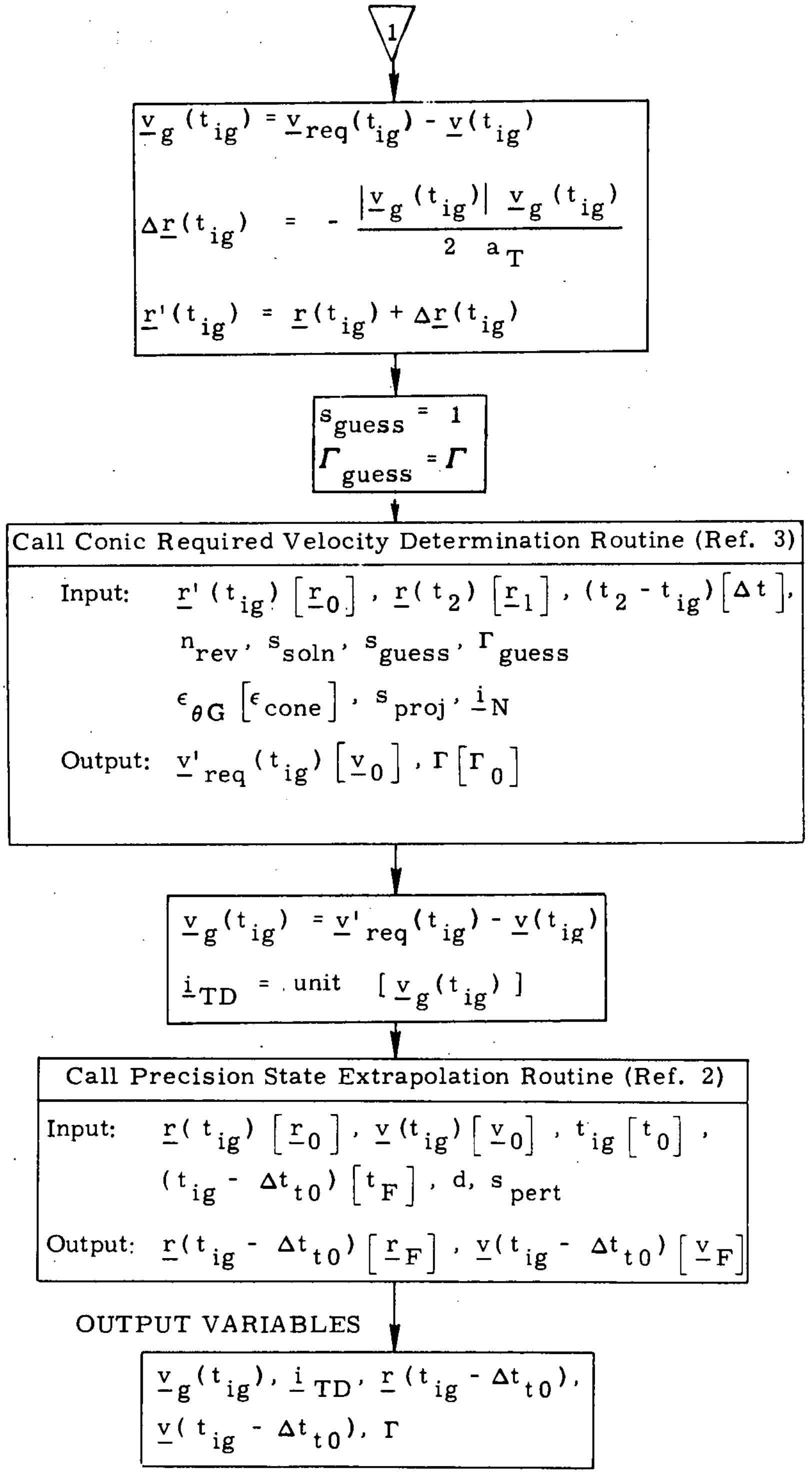
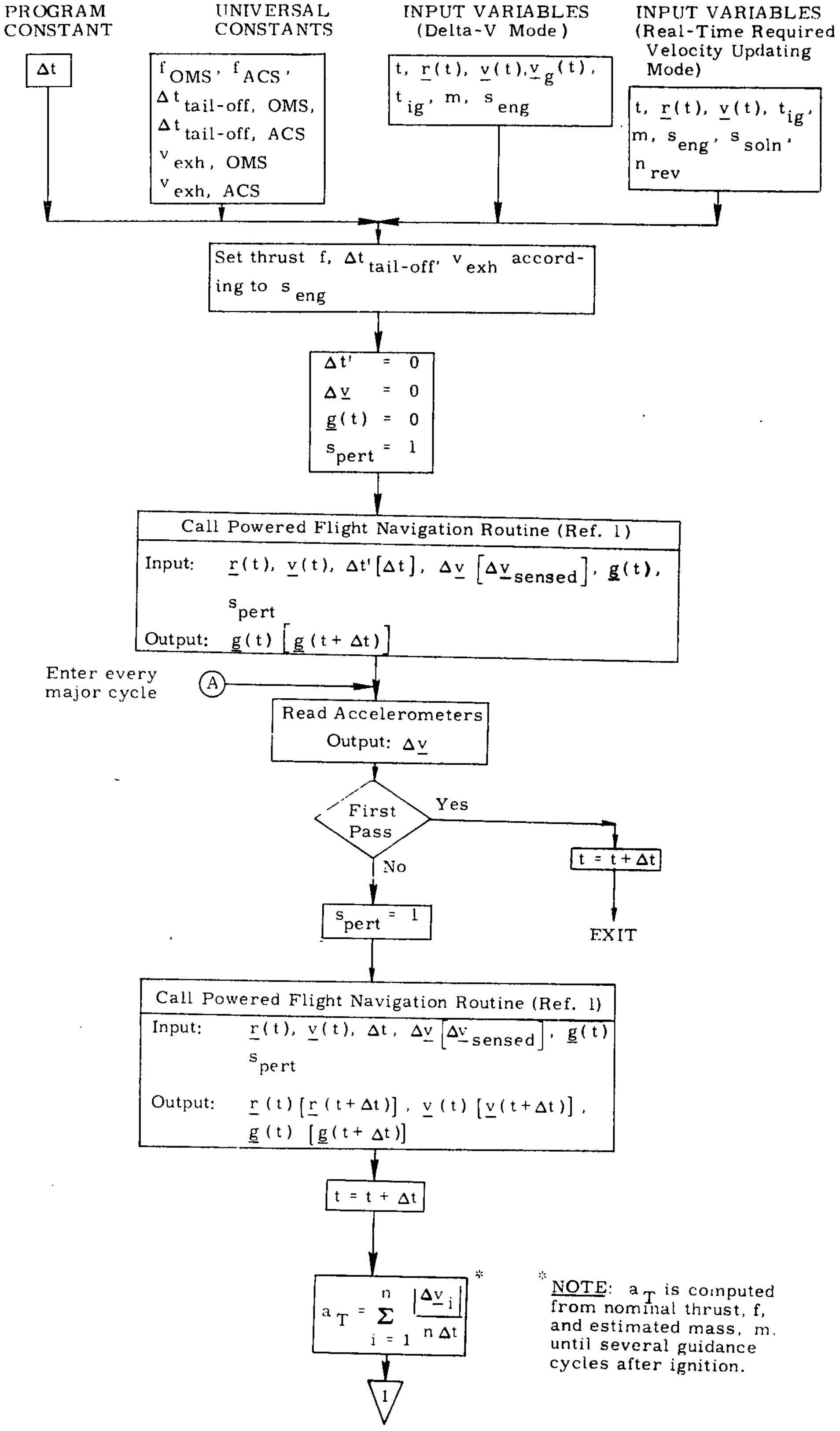

1
$\underline{v}_g(t_{ig}) = \underline{v}_{req}(t_{ig}) - \underline{v}(t_{ig})$
$\Delta\underline{r}(t_{ig}) = - \frac{|\underline{v}_g(t_{ig})| \ \underline{v}_g(t_{ig})}{2 \ a_T}$
$\underline{r}'(t_{ig}) = \underline{r}(t_{ig}) + \Delta\underline{r}(t_{ig})$
$s_{guess} = 1$
$\Gamma_{guess} = \Gamma$
Call Conic Required Velocity Determination Routine (Ref. 3)
Input: $\underline{r}'(t_{ig}) \left[\underline{r}_0\right], \underline{r}(t_2) \left[\underline{r}_1\right], (t_2 - t_{ig}) \left[\Delta t\right],$
$n_{rev}, s_{soln}, s_{guess}, \Gamma_{guess}$
$\epsilon_{\theta G} \left[\epsilon_{cone}\right], s_{proj}, \underline{i}_N$
Output: $\underline{v}'_{req}(t_{ig}) \left[\underline{v}_0\right], \Gamma \left[\Gamma_0\right]$
$\underline{v}_g(t_{ig}) = \underline{v}'_{req}(t_{ig}) - \underline{v}(t_{ig})$
$\underline{i}_{TD} = \text{unit} \ [\underline{v}_g(t_{ig})]$
Call Precision State Extrapolation Routine (Ref. 2)
Input: $\underline{r}(t_{ig}) \left[\underline{r}_0\right], \underline{v}(t_{ig}) \left[\underline{v}_0\right], t_{ig} \left[t_0\right],$
$(t_{ig} - \Delta t_{t0}) \left[t_F\right], d, s_{pert}$
Output: $\underline{r}(t_{ig} - \Delta t_{t0}) \left[\underline{r}_F\right], \underline{v}(t_{ig} - \Delta t_{t0}) \left[\underline{v}_F\right]$
OUTPUT VARIABLES
$\underline{v}_g(t_{ig}), \underline{i}_{TD}, \underline{r}(t_{ig} - \Delta t_{t0}),$
$\underline{v}(t_{ig} - \Delta t_{t0}), \Gamma$

Figure 7b. Prethrust Phase of Real-Time Required Velocity Updating Guidance Mode


PROGRAM CONSTANT
Δt
UNIVERSAL CONSTANTS
f_{OMS}, f_{ACS}, $\Delta t_{tail-off, OMS}$, $\Delta t_{tail-off, ACS}$, $v_{exh, OMS}$, $v_{exh, ACS}$
INPUT VARIABLES (Delta-V Mode)
t, $\underline{r}(t)$, $\underline{v}(t)$, $\underline{v}_g(t)$, t_{ig}, m, s_{eng}
INPUT VARIABLES (Real-Time Required Velocity Updating Mode)
t, $\underline{r}(t)$, $\underline{v}(t)$, t_{ig}, m, s_{eng}, s_{soln}, n_{rev}
Set thrust f, $\Delta t_{tail-off}$, v_{exh} according to s_{eng}
$\Delta t' = 0$
$\Delta \underline{v} = 0$
$\underline{g}(t) = 0$
$s_{pert} = 1$
Call Powered Flight Navigation Routine (Ref. 1)
Input: $\underline{r}(t)$, $\underline{v}(t)$, $\Delta t'[\Delta t]$, $\Delta \underline{v}\ [\Delta \underline{v}_{sensed}]$, $\underline{g}(t)$, s_{pert}
Output: $\underline{g}(t)\ [\underline{g}(t+\Delta t)]$
Enter every major cycle
A
Read Accelerometers
Output: $\Delta \underline{v}$
First Pass
Yes
$t = t + \Delta t$
EXIT
No
$s_{pert} = 1$
Call Powered Flight Navigation Routine (Ref. 1)
Input: $\underline{r}(t)$, $\underline{v}(t)$, Δt, $\Delta \underline{v}\ [\Delta \underline{v}_{sensed}]$, $\underline{g}(t)$, s_{pert}
Output: $\underline{r}(t)\ [\underline{r}(t+\Delta t)]$, $\underline{v}(t)\ [\underline{v}(t+\Delta t)]$, $\underline{g}(t)\ [\underline{g}(t+\Delta t)]$
$t = t + \Delta t$
$a_T = \sum_{i=1}^{n} \frac{|\Delta \underline{v}_i|}{n \Delta t}$ *
1

* NOTE: a_T is computed from nominal thrust, f, and estimated mass, m, until several guidance cycles after ignition.

Figure 8a. Powered Flight Routines

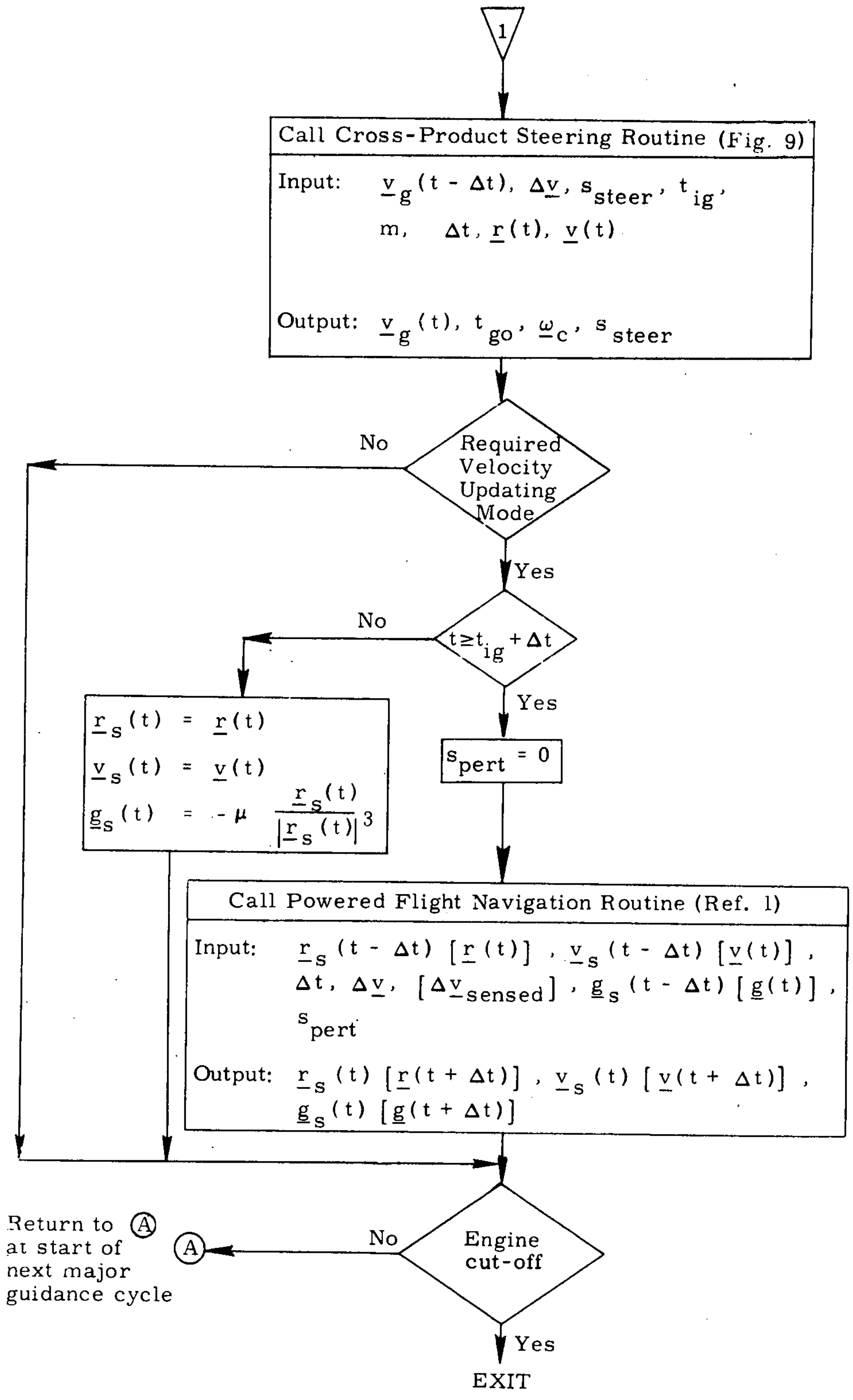
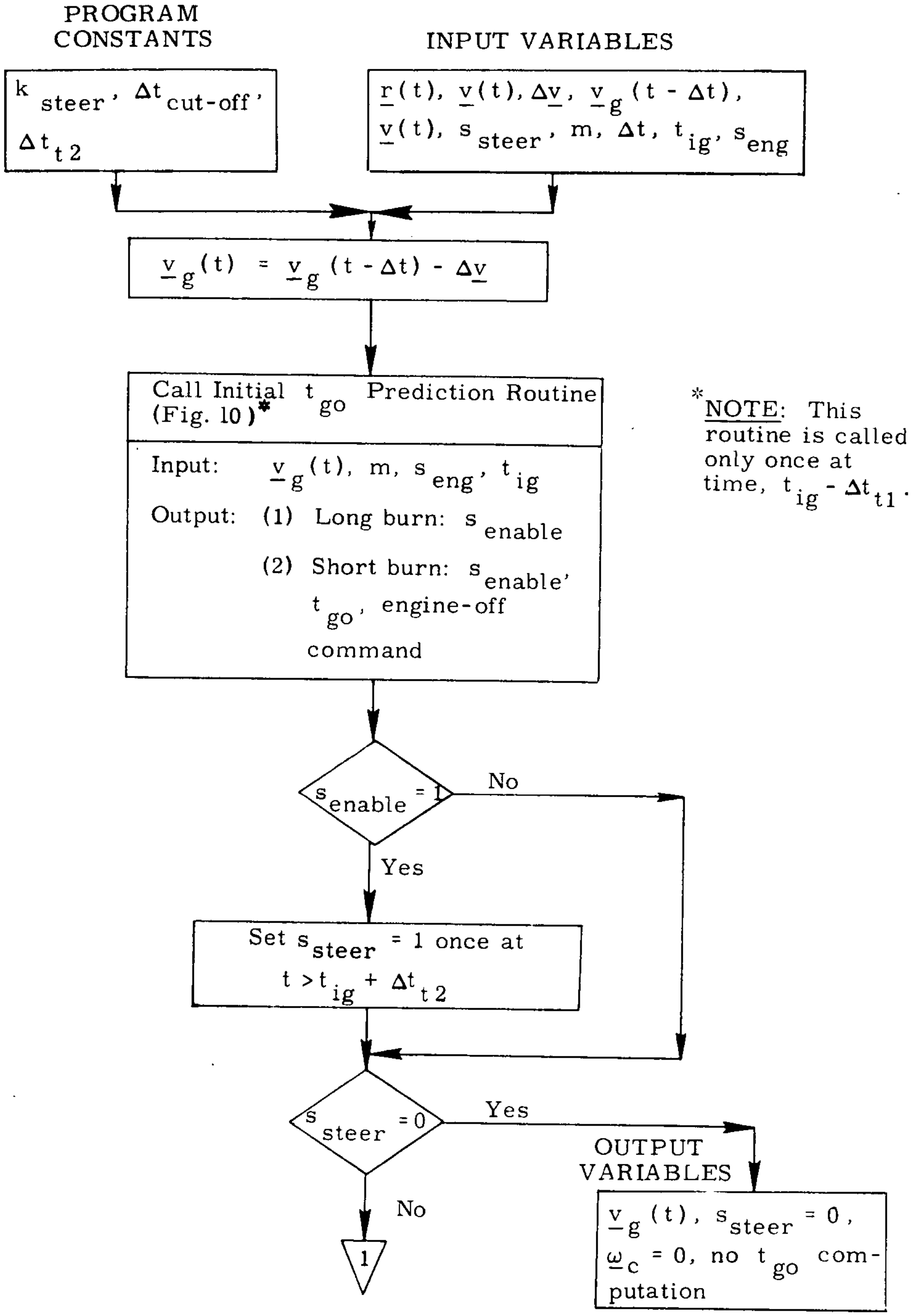

1
Call Cross-Product Steering Routine (Fig. 9)
Input: $\underline{v}_g(t - \Delta t)$, $\Delta\underline{v}$, s_{steer}, t_{ig}, m, Δt, $\underline{r}(t)$, $\underline{v}(t)$
Output: $\underline{v}_g(t)$, t_{go}, $\underline{\omega}_c$, s_{steer}
Required Velocity Updating Mode
No
Yes
$t \geq t_{ig} + \Delta t$
No
Yes
$\underline{r}_s(t) = \underline{r}(t)$
$\underline{v}_s(t) = \underline{v}(t)$
$\underline{g}_s(t) = -\mu \frac{\underline{r}_s(t)}{|\underline{r}_s(t)|^3}$
$s_{pert} = 0$
Call Powered Flight Navigation Routine (Ref. 1)
Input: $\underline{r}_s(t - \Delta t)$ $[\underline{r}(t)]$, $\underline{v}_s(t - \Delta t)$ $[\underline{v}(t)]$, Δt, $\Delta\underline{v}$, $[\Delta\underline{v}_{sensed}]$, $\underline{g}_s(t - \Delta t)$ $[\underline{g}(t)]$, s_{pert}
Output: $\underline{r}_s(t)$ $[\underline{r}(t + \Delta t)]$, $\underline{v}_s(t)$ $[\underline{v}(t + \Delta t)]$, $\underline{g}_s(t)$ $[\underline{g}(t + \Delta t)]$
Engine cut-off
No
Yes
EXIT
A
Return to A at start of next major guidance cycle

Figure 8b. Powered Flight Routines


PROGRAM CONSTANTS
k_{steer}, $\Delta t_{cut\text{-}off}$, Δt_{t2}
INPUT VARIABLES
$\underline{r}(t)$, $\underline{v}(t)$, $\Delta\underline{v}$, $\underline{v}_g(t - \Delta t)$, $\underline{v}(t)$, s_{steer}, m, Δt, t_{ig}, s_{eng}
$\underline{v}_g(t) = \underline{v}_g(t - \Delta t) - \Delta\underline{v}$
Call Initial t_{go} Prediction Routine (Fig. 10)*
Input: $\underline{v}_g(t)$, m, s_{eng}, t_{ig}
Output: (1) Long burn: s_{enable}
(2) Short burn: s_{enable}, t_{go}, engine-off command
$s_{enable} = 1$
No
Yes
Set $s_{steer} = 1$ once at $t > t_{ig} + \Delta t_{t2}$
$s_{steer} = 0$
Yes
No
1
OUTPUT VARIABLES
$\underline{v}_g(t)$, $s_{steer} = 0$, $\underline{\omega}_c = 0$, no t_{go} computation

*NOTE: This routine is called only once at time, $t_{ig} - \Delta t_{t1}$.

Figure 9a. Cross-Product Steering Routine

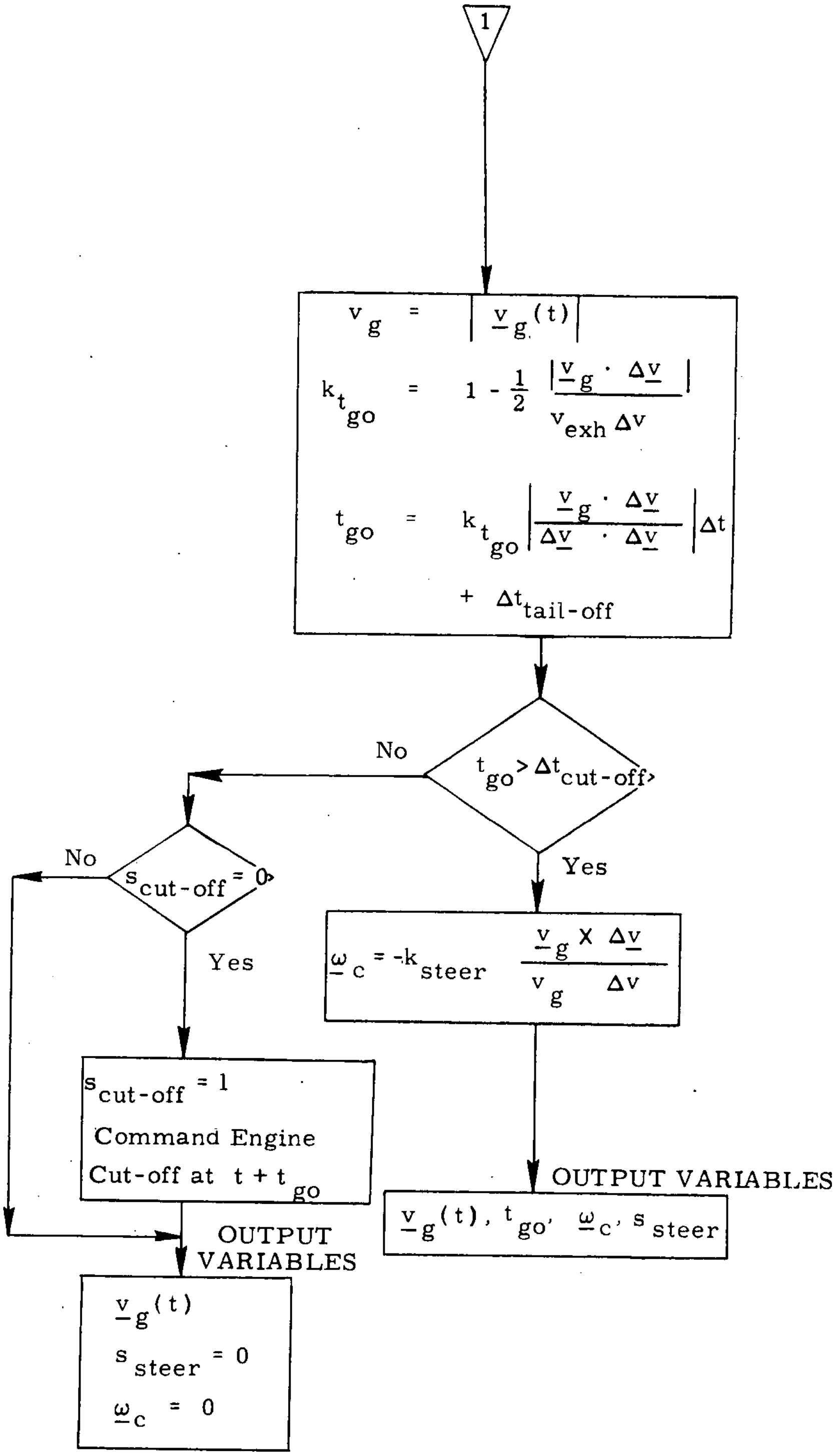

1
$v_g = |\underline{v}_g(t)|$
$k_{t_{go}} = 1 - \frac{1}{2} \frac{|\underline{v}_g \cdot \Delta\underline{v}|}{v_{exh} \Delta v}$
$t_{go} = k_{t_{go}} \left| \frac{\underline{v}_g \cdot \Delta\underline{v}}{\Delta\underline{v} \cdot \Delta\underline{v}} \right| \Delta t + \Delta t_{tail-off}$
$t_{go} > \Delta t_{cut-off}$
No
Yes
$s_{cut-off} = 0$
No
Yes
$\underline{\omega}_c = -k_{steer} \frac{\underline{v}_g \times \Delta\underline{v}}{v_g \quad \Delta v}$
$s_{cut-off} = 1$
Command Engine
Cut-off at $t + t_{go}$
OUTPUT VARIABLES
$\underline{v}_g(t), t_{go}, \underline{\omega}_c, s_{steer}$
OUTPUT VARIABLES
$\underline{v}_g(t)$
$s_{steer} = 0$
$\underline{\omega}_c = 0$

Figure 9b. Cross-Product Steering Routine

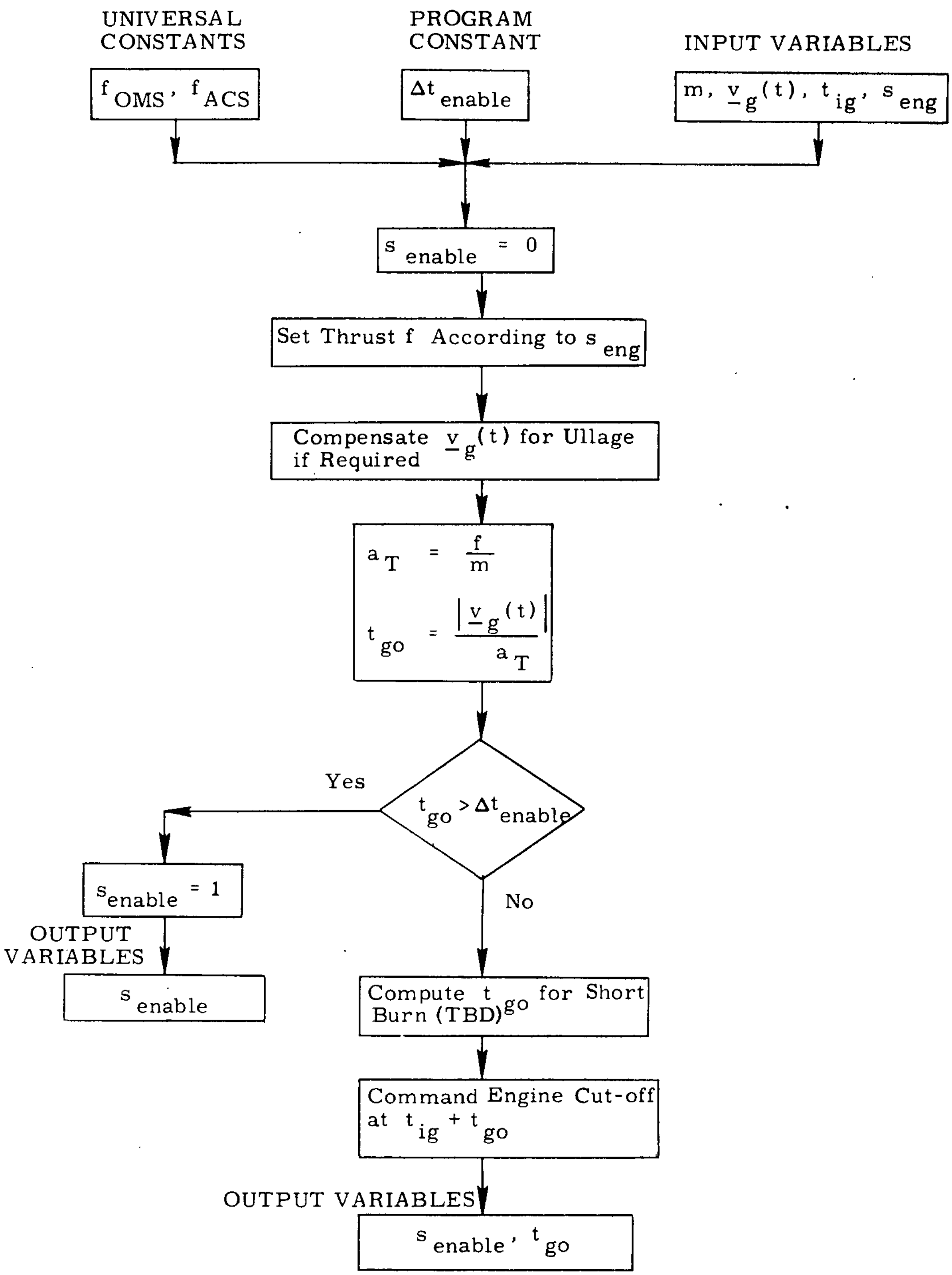

UNIVERSAL CONSTANTS
f_{OMS}, f_{ACS}
PROGRAM CONSTANT
Δt_{enable}
INPUT VARIABLES
m, $\underline{v}_g(t)$, t_{ig}, s_{eng}
$s_{enable} = 0$
Set Thrust f According to s_{eng}
Compensate $\underline{v}_g(t)$ for Ullage if Required
$a_T = \frac{f}{m}$
$t_{go} = \frac{|\underline{v}_g(t)|}{a_T}$
$t_{go} > \Delta t_{enable}$
Yes
No
$s_{enable} = 1$
OUTPUT VARIABLES
s_{enable}
Compute t_{go} for Short Burn (TBD)
Command Engine Cut-off at $t_{ig} + t_{go}$
OUTPUT VARIABLES
s_{enable}, t_{go}

Figure 10. Initial t_{go} Prediction Routine

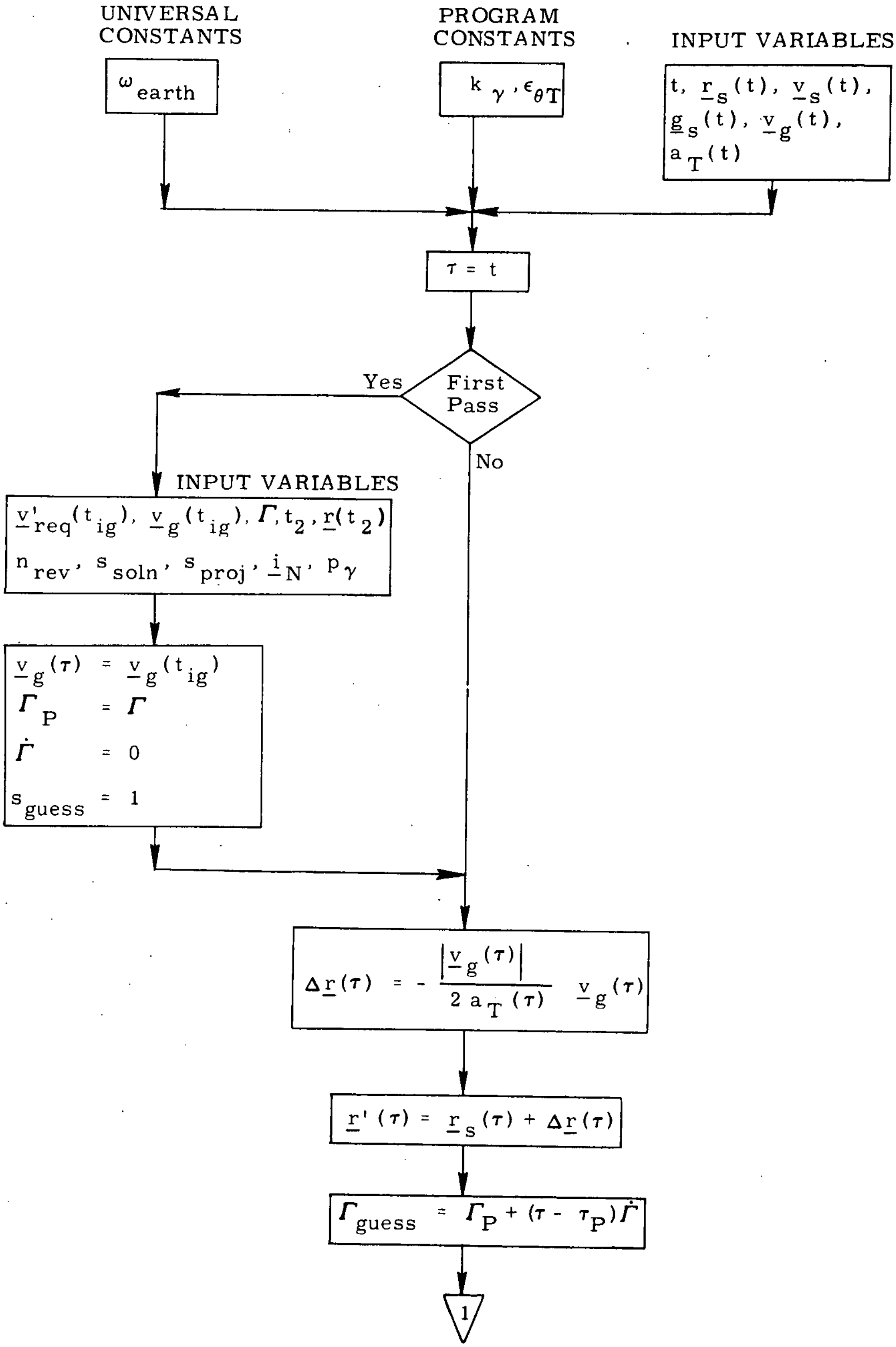

UNIVERSAL CONSTANTS
ω_{earth}
PROGRAM CONSTANTS
$k_{\gamma}, \epsilon_{\theta T}$
INPUT VARIABLES
$t, \underline{r}_s(t), \underline{v}_s(t), \underline{g}_s(t), \underline{v}_g(t), a_T(t)$
$\tau = t$
First Pass
Yes
No
INPUT VARIABLES
$\underline{v}'_{req}(t_{ig}), \underline{v}_g(t_{ig}), \Gamma, t_2, \underline{r}(t_2)$
$n_{rev}, s_{soln}, s_{proj}, \underline{i}_N, p_{\gamma}$
$\underline{v}_g(\tau) = \underline{v}_g(t_{ig})$
$\Gamma_P = \Gamma$
$\dot{\Gamma} = 0$
$s_{guess} = 1$
$\Delta\underline{r}(\tau) = -\frac{|\underline{v}_g(\tau)|}{2 a_T(\tau)} \underline{v}_g(\tau)$
$\underline{r}'(\tau) = \underline{r}_s(\tau) + \Delta\underline{r}(\tau)$
$\Gamma_{guess} = \Gamma_P + (\tau - \tau_P)\dot{\Gamma}$
1

Figure 11a. Velocity-to-be-Gained Routine

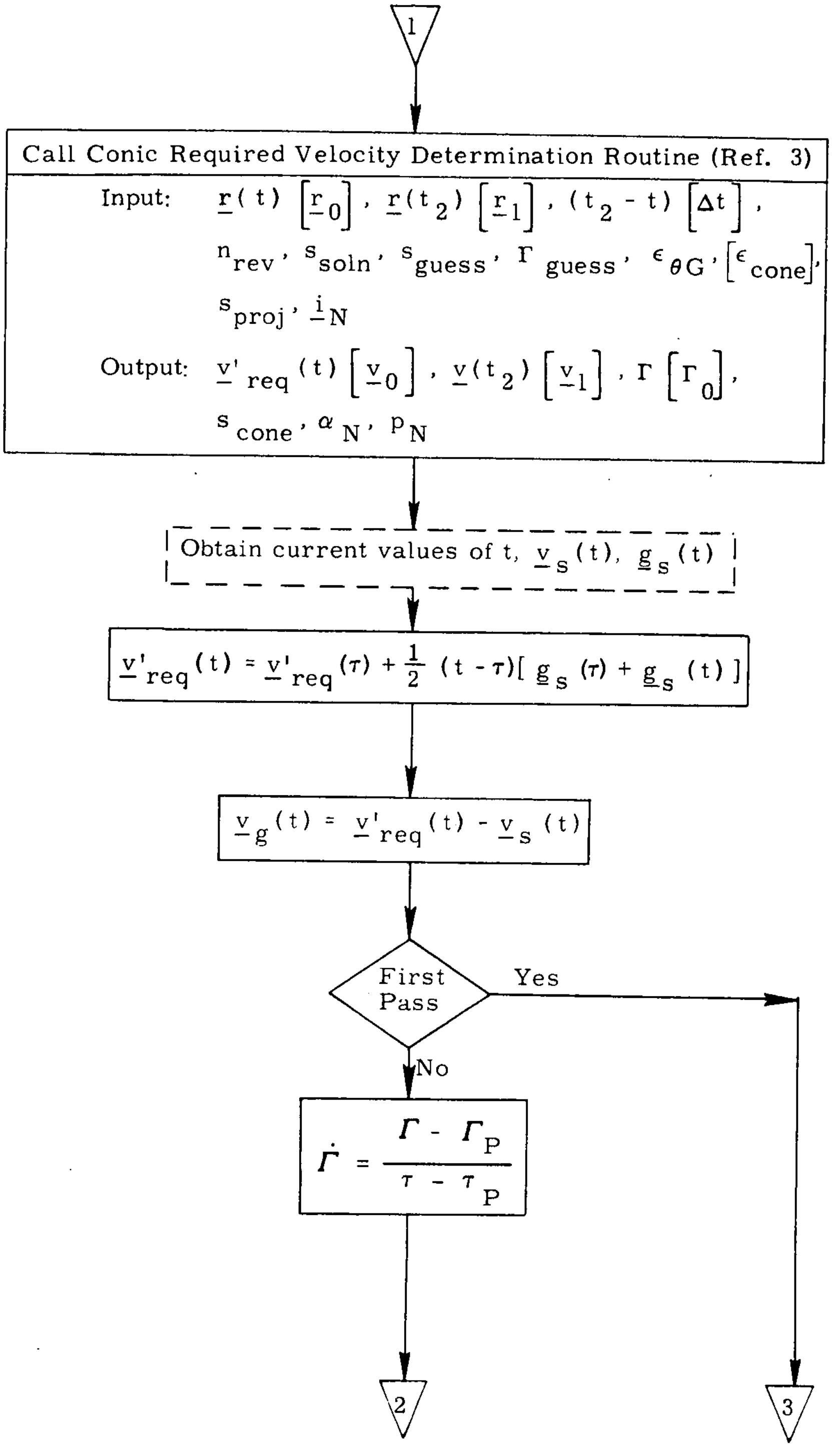
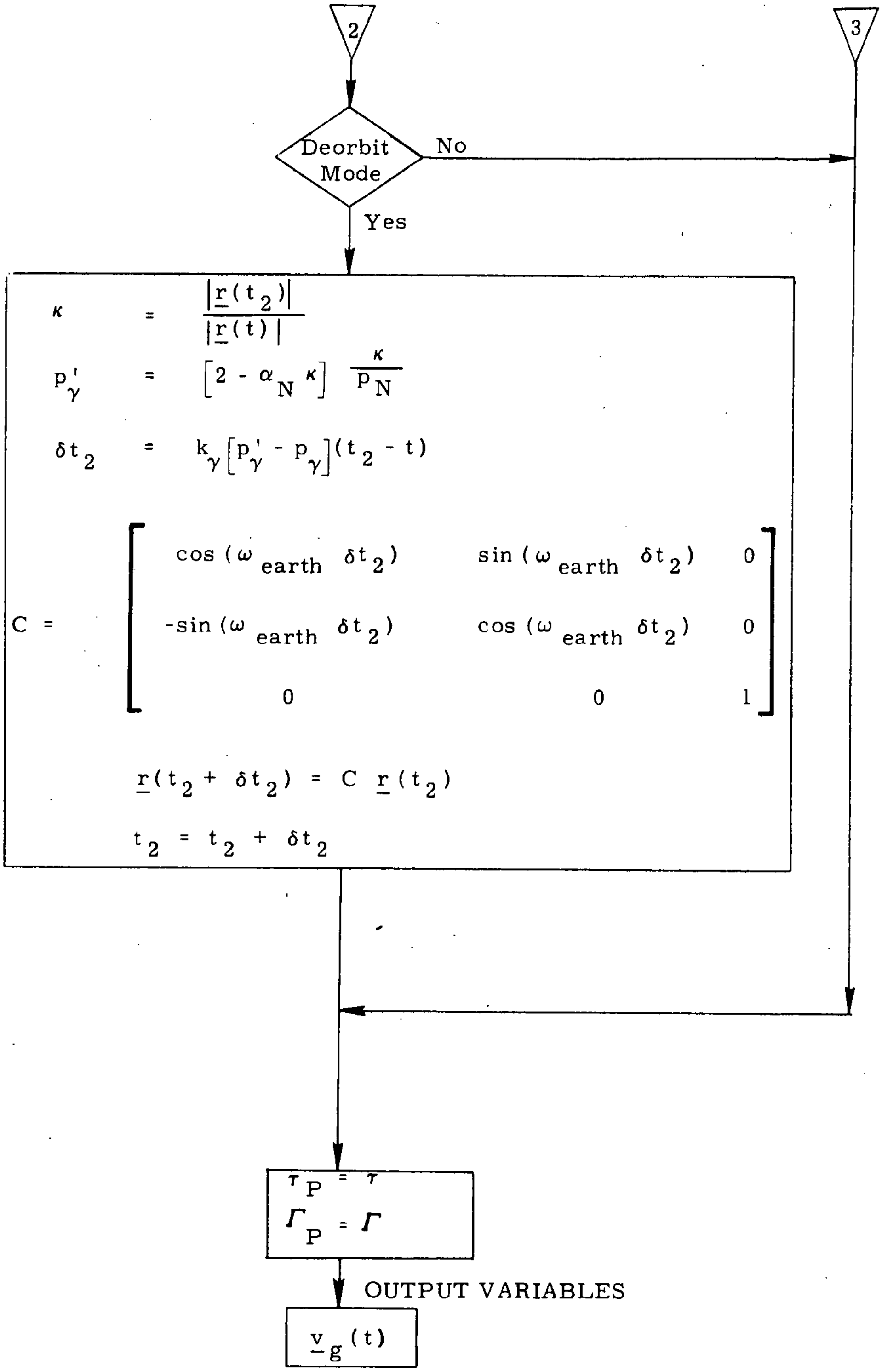

1
Call Conic Required Velocity Determination Routine (Ref. 3)
Input: $\underline{r}(t)\ [\underline{r}_0]$, $\underline{r}(t_2)\ [\underline{r}_1]$, $(t_2 - t)\ [\Delta t]$, n_{rev}, s_{soln}, s_{guess}, Γ_{guess}, $\epsilon_{\theta G}$, $[\epsilon_{cone}]$, s_{proj}, $\underline{i}_N$
Output: $\underline{v}'_{req}(t)\ [\underline{v}_0]$, $\underline{v}(t_2)\ [\underline{v}_1]$, $\Gamma\ [\Gamma_0]$, s_{cone}, α_N, p_N
Obtain current values of t, $\underline{v}_s(t)$, $\underline{g}_s(t)$
$\underline{v}'_{req}(t) = \underline{v}'_{req}(\tau) + \frac{1}{2}(t - \tau)[\underline{g}_s(\tau) + \underline{g}_s(t)]$
$\underline{v}_g(t) = \underline{v}'_{req}(t) - \underline{v}_s(t)$
First Pass
Yes
No
$\dot{\Gamma} = \dfrac{\Gamma - \Gamma_P}{\tau - \tau_P}$
2
3

Figure 11b. Velocity-to-be-Gained Routine



$$\kappa = \frac{|\underline{r}(t_2)|}{|\underline{r}(t)|}$$

$$p'_\gamma = \left[2 - \alpha_N \kappa\right] \frac{\kappa}{p_N}$$

$$\delta t_2 = k_\gamma \left[p'_\gamma - p_\gamma\right](t_2 - t)$$

$$C = \begin{bmatrix} \cos(\omega_{earth} \; \delta t_2) & \sin(\omega_{earth} \; \delta t_2) & 0 \\ -\sin(\omega_{earth} \; \delta t_2) & \cos(\omega_{earth} \; \delta t_2) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\underline{r}(t_2 + \delta t_2) = C \; \underline{r}(t_2)$$

$$t_2 = t_2 + \delta t_2$$

$$\tau_P = \tau$$

$$\Gamma_P = \Gamma$$

$$\underline{v}_g(t)$$

Figure 11c. Velocity-to-be-Gained Routine

6. SUPPLEMENTARY INFORMATION

The guidance equations presented for use in the Real-Time Required Velocity Updating Mode represent a departure from the APOLLO design. A complete description and derivation of the new concept can be found in Reference 5. The principle advantages of this scheme are an essentially constant attitude maneuver and an improved estimate of velocity-to-be-gained for more accurate cutoff. In addition, if the solution for required velocity cannot be completed every major guidance cycle, this scheme adapts particularly well to the problems of intermediate velocity-to-be-gained solutions and computational lag.

A comparison between the STS Lambert guidance equations, as formulated in this report, and the APOLLO Lambert guidance equations has been made for two typical STS orbital maneuvers and the results are contained in Reference 4. These results show that the STS Lambert guidance equations are significantly more accurate than the APOLLO equations. For the two maneuvers that were analyzed, the residual velocities-to-be-gained resulting from using the STS equations were less than 0.001 ft/sec.

REFERENCES

1. Robertson, William M., "Rapid Real-Time State Advancement During Specific Force Sensing", Space Shuttle GN & C Equation Document No. 5, Rev. 1, M.I.T./DL, To be published.

2. Robertson, William M., "Precision State and Filter Weighting Matrix Extrapolation", Space Shuttle GN & C Equation Document No. 4, Rev. 1, M.I.T./DL, To be published.

3. Robertson, William M., "Conic Required Velocity Determination", Space Shuttle GN & C Equation Document No. 10, Rev. 1, M.I.T./DL, September 1971.

4. Higgins, J. P., "Analysis of Proposed STS Lambert Guidance Equations", 23A STS Memo No. 31-71, M.I.T./DL,June 1971

5. Brand, T., "A New Approach to Lambert Guidance", R-694, M.I.T./DL, June 1971

6. Brand, T., "Precision Required Velocity Determination", Space Shuttle GN & C Equation Document No. 13, Rev. 1, M.I.T./DL, September 1971.

9.7.4 Thrust Vector Control

The decision has not yet been made whether the OMPS engines will be gimballed or fixed. Additionally, the possibility is being explored of using the gimbal actuators for trim but not for active TVC. Consequently, while the nature of the OMPS engines and their utilization is deliberated, Equations have been baselined for both fixed-engines and gimballed-engines TVC.

9.7.4.1 TVC, Fixed

SPACE SHUTTLE

GN&C SOFTWARE EQUATION SUBMITTAL

Software Equation Section Orbital Powered Flight Submittal No. 31

Function Orbital TVC Autopilot (Fixed OMPS)

Module No. OC3 Function No. 1 (mod.), 2 (MSC 03690)

Submitted by: J. Sunkel Co. GCD (EG-05152)

Date: 8/24/71

NASA Contract: W. H. Peters (name) Organization GCD

Approved by Panel III K. J. Cox (chairman) Date 8/24/71

Summary Description: The objective of the autopilot is to provide attitude control of the vehicle during on-orbit powered flight. Control implies following pitch commands from guidance.

Shuttle Configuration: (Vehicle, Aero Data, Sensor, Et Cetera)

On orbit vehicle, angular rates available, control torque from ACPS jets.

Comments:

(Design Status)

(Verification Status) Verification simulations performed.

Panel Comments:

9.7.4.1 A DIGITAL FLIGHT CONTROL LOGIC FOR ON-ORBIT THRUST VECTOR CONTROL WITH OMPS ENGINES FIXED

SUMMARY

This internal note documents the design of a DFCS (digital flight control system) for on-orbit TVC (thrust vector control) with OMPS engines fixed. The DFCS logic is comprised of three distinct operating modes: (1) Large error convergence logic, (2) limit cycle operation with disturbance torques present, and (3) minimum impulse limit cycle operation. Other unique features are (1) Simple disturbance torque estimation, (2) limit cycle logic that produces zero time average state erors while avoiding firings that are in phase with a disturbance torque, and (3) extremely simple logic for determining if control action is required. The design concepts have been tested and verified on the Univac 1108 computer.

INTRODUCTION

This analysis is concerned with the development of a digital autopilot able to control orbiter attitude and rate during on-orbit thrusting maneuvers. Basic to this design is the assumption that the OMPS engines are fixed. Thus all attitude hold and steering maneuvers are performed with the ACPS (attitude control propulsion system) under influence of substantial disturbance torques.

A summary of some of the previous work done in this area, principally by MIT, for the Apollo Program is contained in reference 1. This design effort has endeavored to simplify as much as possible the computations and logic required in these earlier DFCS designs.

The design effort and testing has been limited to a single axis which has been assumed to be the orbiter pitch axis. However, the same configuration is directly applicable to the roll and yaw axes. The complete 3-axis system will be verified in the six-degree-of-freedom MSC SSV functional simulator (reference 2).

Problem Definition

For rotation in the pitch plane, the moment equation is

$$I \ddot{\theta} = T + M \tag{1}$$

where T is the applied control torque and M is a disturbance torque.

By introducing attitude error θ and attitude error rate $\dot{\theta}$ as our state variables, we have

$$\bar{Y} = \begin{bmatrix} Y_1 \\ Y_2 \end{bmatrix} = \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix}$$

2

Therefore

$$\dot{Y}_1 = Y_2$$
$$\dot{Y}_2 = \frac{T}{I} + \frac{M}{I} \qquad (2)$$

We assume $\left| \frac{T}{I} \right| \leq V$ where V is maximum control acceleration

Define

$$X_1 \triangleq \frac{Y1}{V} \qquad \sec^2$$
$$X_2 \triangleq \frac{Y2}{V} \qquad \sec$$

Therefore

$$\dot{X}_1 = \frac{Y2}{V} = X2$$
$$\dot{X}_2 = \frac{T}{I\ V} + \frac{M}{IV} \qquad (3)$$

Define

$$\frac{T}{I\ V} \triangleq U \qquad \text{nondimensional}$$

$$\frac{M}{IV} \triangleq D \qquad \text{nondimensional}$$

Substituting into eqs (3) gives the normalized state equations

$$\dot{X}_1 = X_2$$
$$\dot{X}_2 = U + D \qquad (4)$$

where $|U| \leq 1$

3

The problem is to design a digital control logic for the system in equation (4), which will reduce large errors in attitude and rate to negligible values and hold them there in the presence of disturbance torques while performing in a fuel optimal manner. The autopilot logic will be broken into three parts: (1) Large error convergence logic, (2) limit cycle operation, and (3) minimum impulse limit cycle operation.

Large Error Convergence Logic

The system under consideration is described by eqn. (5), where X_1 and X_2 are the state variables to be controlled by U and D is a disturbance torque:

$$\begin{aligned} \dot{X}_1 &= X_2 \\ \dot{X}_2 &= U + D \end{aligned} \qquad (5)$$

The objective will be to define a control history U (t) over the time interval $t = 0$ to $t = T$, where T is unspecified, such that:

1. The system is driven from an arbitrary initial state

$$\begin{aligned} X_1(0) &= \xi_1 \\ X_2(0) &= \xi_2 \end{aligned} \qquad (6)$$

to a fixed, final state

$$\begin{aligned} X_1\,(T) &= 0 \\ X_2\,(T) &= 0 \end{aligned} \qquad (7)$$

2. The control signal is limited by the relation

$$|U(t)| \leq 1 \quad \text{for} \quad t \in [0, T] \qquad (8)$$

3. The control signal minimizes a weighted fuel-time functional of the form

$$I = \int_0^T (K + |U|)\, dt, \quad K \text{ is a positive weighting constant} \qquad (9)$$

The objective of the problem is to find the functional form of the control U which will minimize I in eqn. (9).

The solution of this problem is documented in reference 3 for the second order system:

$$\dot{X}_1 = X_2 \qquad \dot{X}_2 = U \tag{10}$$

The results may be summarized by use of phase plane partitioning and the association of a control signal (U = -1, 0, 1) with each partitioned region. Figure 1 shows the phase plane partitioning into four regions G+, G-, H+, and H-. The partitioning curves $\gamma+$, $\gamma-$, $\Gamma+$, and $\Gamma-$ are associated with the regions as follows:

$$\gamma + \subset G + \qquad \gamma - \subset G - \qquad \Gamma + \subset H + \qquad \Gamma - \subset H - \tag{11}$$

where the partitioning curves are defined as follows:

$$\gamma - \ : \ X_2 \geqslant 0, \quad X_1 = -\tfrac{1}{2} X_2^2 \tag{12a}$$

$$\gamma + \ : \ X_2 \leqslant 0, \quad X_1 = \tfrac{1}{2} X_2^2 \tag{12b}$$

$$\Gamma - \ : \ X_2 \leqslant 0, \quad X_1 = \tfrac{1}{2} \left(\frac{K + 4}{K}\right) X_2^2 \tag{12c}$$

$$\Gamma + \ : \ X_2 \geqslant 0, \quad X_1 = -\tfrac{1}{2} \left(\frac{K + 4}{K}\right) X_2^2 \tag{12d}$$

The control laws derived may be summarized as a phase plane switch logic:

$$\begin{aligned} U &= -1 \quad \text{for } (X_1, X_2) \in G- \\ U &= 0 \quad \text{for } (X_1, X_2) \in H- \cup H+ \\ U &= +1 \quad \text{for } (X_1, X_2) \in G+ \end{aligned} \tag{13}$$

Figure 1 also shows a typical optimal trajectory originating in G-. Note the characteristic "bang-coast-bang" nature of the control, due to the weighted fuel-time cost function.

Predicting Control Logic

Derivation of a predictive, or open loop, control history U (t) is a straightforward application of the previous results and is documented in reference 3 for the undisturbed second order system given by equation (10). What is required is a determination of a switch time (T_i) for each trajectory crossover in the phase plane, as a function of the systems initial conditions (ξ_1, ξ_2). Figure 2 shows a trajectory originating in G- and switching at $t = T_1$ and $t = T_2$. The switching time equations are summarized as follows:

For $(\xi_1, \xi_2) \in$ G-

$$T_1 = \xi_2 + \left(\frac{\xi_2^2 + 2\xi_1}{1 + P_K}\right)^{\frac{1}{2}} \tag{14a}$$

$$T_2 = \xi_2 + \frac{1}{2}\ (1 + P_K)\left(\frac{\xi_2^2 + 2\xi_1}{1 + P_K}\right)^{\frac{1}{2}} \tag{14b}$$

where $P_K \triangleq \frac{K + 4}{K}$

For $(\xi_1, \xi_2) \in$ G+

$$T_1 = -\xi_2 + \left(\frac{\xi_2^2 - 2\xi_1}{1 + P_K}\right)^{\frac{1}{2}} \tag{14c}$$

$$T_2 = -\xi_2 + \frac{1}{2}\ (1 + P_K)\left(\frac{\xi_2^2 - 2\xi_1}{1 + P_K}\right)^{\frac{1}{2}} \tag{14d}$$

If $(\xi_1, \xi_2) \in$ H - , it is clear that there is only one switching time of interest. Let T_3 be the time when U switches from U = 0 to U = +1.

Therefore

$$T_3 = \frac{1}{2}\ \xi_2 - \xi_1 \Big/ \xi_2 \tag{15a}$$

For $(\xi_1, \xi_2) \in$ H +

$$T_3 = -\frac{1}{2}\ \xi_2 - \xi_1 \Big/ \xi_2 \tag{15b}$$

Once U switches from U = 0 to U = 1, the system follows the γ +curve to the origin. Let T_4 be the time predicted to travel to the origin. Then if the system switches at $(B_1, -B_2)$, T_4 is given by:

$$T_4 = -B_2 \tag{16a}$$

Similarly for the $\gamma -$ curve

$$T_4 = B_2 \tag{16b}$$

The equations above have been derived for the system given by equation (10). The question which now arises is what effect a disturbance torque has on the above results. If we assumed that the "bang-coast-bang" nature of the solution will remain the same, and still require that the final state condition be the origin of the phase plane, as specified by equation (7), then the $\gamma +$ and $\gamma -$ curves must be replaced by the natural trajectories of the system (5) which intersect the origin. Thus the disturbance torque is factored into equations (12a) and (12b) which become:

$$\gamma - : \quad X_2 \geqslant 0 , \quad X_1 = \frac{1}{2(-1 + D)} X_2^2 \tag{17a}$$

$$\gamma + : \quad X_2 \leqslant 0 , \quad X_1 = \frac{1}{2(1 + D)} X_2^2 \tag{17b}$$

The time to travel to the origin (T_4) along the $\gamma +$ becomes:

$$T_4 = - \frac{B_2}{1 + D} \tag{18a}$$

Similarly for the $\gamma -$ curve

$$T_4 = - \frac{B_2}{D - 1} \tag{18b}$$

In the case of the Γ curves, however, the equations become extremely complicated when a disturbance torque is considered. The same is true of the expressions for T_1, T_2, and T_3. The very small errors which result from ignoring the disturbance torque dictate that these equations remain unchanged.

The equations summarized above constitute the predictive controller to be used in the large error convergence logic.

Limit Cycle Operation

Once the large error convergence logic has reduced large errors in attitude and rate to relatively small values, we need an attitude hold logic which will allow us to operate within a small region about the phase plane origin with minimum fuel usage. This dictates the need for a "fuel efficient" limit cycle logic. In addition to the fuel economy requirement there is a requirement that the time average attitude deviation from the phase plane origin (attitude error) be near zero to avoid inserting a bias in the guidance loop.

Basically, the "limit cycle" requirements are met by the following design features which are illustrated in figure 3:

a. No control action is taken as long as the two element state vector magnitude is less than a specified constant R1.

b. Limit cycle conditions are differentiated from large error convergence conditions by another check on the state vector magnitude for exceeding a specified value R2.

c. When significant disturbance torques exist, control torque application times are computed based on the time required for the intersection of two parabolas. The first is the trajectory parabola defined by: (1) Current state, (2) control acceleration, and (3) disturbance acceleration. The second parabola is the unique symmetrical limit cycle parabola defined by the two requirements: (1) Time integral of attitude error is zero, and (2) control torque applied upon state vector magnitude exceeding R1, with the simplifying assumption that the time of control torque application is small compared to the remainder of the limit cycle period.

d. When both disturbance torque and rate error are less than specified quantities,then a minimum impulse is fired.

These features are derived and discussed in following sections of this report.

Derivation of Disturbance Torque Limit Cycle Logic

Derivation of a predictive control history requires that we determine the proper sense of U and control application time TJET for each trajectory crossover of the R1 circle, as a function of the crossover point.

The sense of the control U is determined by the quadrant in the phase plane in which the trajectory crosses the switch curve.

The proper jet on time depends, however, on whether or not disturbance torques are present. As a trajectory crosses the R1 circle switch curve, the autopilot makes a disturbance torque estimate by simply dividing the change in the state X_2 by the time lapse. That is

$$D = \frac{\xi_2 - X_{2F}}{\sum_{i=1}^{N} T_i}$$

where T = control sample period

N = number of control logic passes since terminating the last firing.

Figure 4 shows a typical limit cycle trajectory with a positive disturbance torque present. The trajectory crosses the switch curve at (ξ_1, ξ_2). At this point, the autopilot estimates the disturbance torque and in this case, finds it to have a positive sense. Next, the autopilot determines the sense of the control U.

A positive attitude error and positive rate error requie that a negative control be applied, thus U = -1.

The problem now, with the disturbance torque and control sense known, is to find the jet on time.

The jet on time will be calculated so that the resulting limit cycle trajectory is expected to satisfy the following condition:

$$\int_{o}^{T} X_1 \; dt = 0 \tag{19}$$

where T is the time lapse from application of control torque to the anticipated time for application of the next control torque pulse.

Satisfying equation (19) results in a symmetrical limit cycle about the phase plane origin with zero average attitude error.

Figure 5 illustrates the desired symmetrical limit cycle. The problem is to calculate X_{2F} from which the required jet on time is readily available. To do this, we assume $\xi_1 \simeq X_{1F}$. With ξ_1, ξ_2, the disturbance torque, and X_{1F} known, X_{2F} can be derived as shown below. From equation (19), we have:

$$\int_{o}^{T} X_1 \; dt = 0$$

The equation of the symmetrical parabola is given by:

$$X_1 = \frac{X_2^2}{2\ D} - A \tag{20}$$

where: D is the disturbance torque

A is unknown

Substituting (20) into (19) results in

$$\int_{o}^{T} \left(\frac{X_2^2}{2\ D} - A \right) dt = o \tag{21}$$

We neglect the jet on portion of the limit cycle since

$$\left[\int_{o}^{T} X_1 dt\right]_{\text{JET ON}} << \left[\int_{o}^{T} X_1 dt\right]_{\text{JET OFF}}$$

Substituting from equation (5) into equation (21) gives

$$\int_{-X_{2F}}^{X_{2F}} \left(\frac{X_2^2}{2\ D} - A \right) \frac{d\ X_2}{D} = 0$$

Integrating:

$$\frac{X_{2F}^3}{6\ D^2} - \frac{A}{D} X_{2F} = 0 \tag{22}$$

Substituting equation (2) into the constraint $X_{1F}^2 + X_{2F}^2 = R1^2$ results in

$$\left(\frac{X_{2F}^2}{2\ D} - A\right)^2 + X_{2\ F}^2 = R1^2 \tag{23}$$

But $A = \frac{X_{2F}^2}{6\ D}$ from (22). Substituting into equation (23) results in:

$$\frac{X_{2F}^4}{9\ D2} + X_{2F}^2 - R\ 1^2 = 0 \tag{24}$$

Solving the quartic gives:

$$X_{2F} = \pm \frac{3D}{\sqrt{2}} \left\{ -1 + \sqrt{1 + \frac{4\,R}{9\,D^2}} \right\}^{\frac{1}{2}} \tag{25}$$

The ambiguity of the $\pm$ sign is resolved by the fact that in the case where the disturbance torque is positive X_{2F} must be negative, implying that the negative sign is used. Thus

$$X_{2F} = - \frac{3\,D}{\sqrt{2}} \left\{ -1 + \sqrt{1 + \frac{4R1^2}{9\,D^2}} \right]^{\frac{1}{2}} \tag{26}$$

Crossing over the switch curve in quadrant III with a negative disturbance torque results in the same equation. Crossing the switch curve in quadrant III with a positive disturbance, however, requires a change in logic. In this case all that is required is sufficient control to achieve a small positive rate. The disturbance torque alone will then drive the system into quadrant I. This logic lets the disturbance torque do the work and saves on fuel. In this case, we arbitrarily set

$$X_{2F} = -\tfrac{1}{2}\,\xi_2 \tag{27}$$

Crossing the switch curve in quadrant I with a negative disturbance torque requires the same logic.

Once X_{2F} has been determined, the predicted jet on time TJET is calculated from the required rate change divided by the anticipated acceleration:

$$\text{TJET} = \frac{X_{2F} - \xi_2}{U + D} \tag{28}$$

Minimum Impulse Limit Cycle Logic

Figure 6 shows a typical minimum impulse limit cycle trajectory. The trajectory crosses the switch curve at ($-\xi_1$, ξ_2). At this point, the autopilot estimates the disturbance torque and in this case, finds it to be zero or less than a small predetermined value. In addition, the trajectory must crossover the switch curve within the minimum impulse threshold. With these two conditions satisfied, the autopilot determines the sense of the control U. In this case, we have a negative attitude error and negative rate error which requires that a positive control be applied, thus $U = +1$.

The jet on time in the minimum impulse limit cycle mode is a fixed impulse of 50 milliseconds. Therefore,

$$\text{TJET} = .05 \text{ sec} \tag{29}$$

Summary
Discussion of Control Logic

Figure 7 illustrates the complete OMPS TVC digital flight control system logic in the phase plane. In summary, the area enclosed by the circle of radius R1 is the coast region and no jets are fired while the state is within this region except during convergence from large error. The perimeter of the R1 circle is the switch curve for limit cycle operation. Once the state crosses the R1 switch curve, the limit cycle logic determines the proper sense of the control U and the jet on time TJET required to limit cycle about the phase plane origin. The perimeter of the R2 circule is the limit cycle large error convergence interface. Any rate or attitude change which causes the state to exceed the R2 perimeter switches the autopilot into large error convergence logic. Note that when the autopilot is in the limit cycle mode, the large error convergence logic switch curves are not used.

The simplicity of the design is readily apparent. Given a point (ξ_1, ξ_2) the radius vector is calculated as follows:

$$R = \sqrt{\xi_1^2 + \xi_2^2}$$

Proper control action is determined by comparing R with R1 and R2. Further computational economy is obtained by use of flags (logic switches), which effectively provide a memory of the operating mode. A flowchart of the logic is shown in figure 8.

Simulation Results

The autopilot logic has been programmed in Fortran IV and a number of runs have been made on the Univac 1108 to verify the design concepts. Table I lists the values of constants used in the simulation. The results of several simulation runs appear as figures 9 through 13, and are discussed below.

Figure 9 illustrates the large error convergence logic. The initial conditions for the run were (2, 0). The inclination of the trajectory in the coast region is due to the presence of a disturbance torque. In this case, a constant disturbance torque of +.1 was present. Note that the trajectory slightly overshoots the γ + switching curve. This is due to the autopilot sampling period. The control switched from U = +1 to U = 0 at (-.007, .06).

Figure 10 illustrates the disturbance torque limit cycle logic and therefore only the limit cycle switch curves are shown. This run is a continuation of run 1 after control switch off. The disturbance torque was again constant at +.1. The main feature to be noted in figure 12 is the operation of the symmetrical limit cycle logic. Note that we have converged to a symmetrical limit cycle after one cycle. The limit cycle trajectories overshoot the R1 switching curve due to sampling period of the autopilot.

Figure 11 illustrates the disturbance torque limit cycle logic for a constant disturbance torque of -.01. Here again, we enter a symmetrical limit cycle during the first cycle.

Figure 12 illustrates the minimum impulse limit cycle logic for a constant disturbance torque of +.001. In this mode, the limit cycle switches twice during each cycle. Note that during the second cycle, the trajectory exceeds the minimum impulse threshold as it crosses the switch curve. This is due to the effect of the positive disturbance torque. In this case, the system switches into the disturbance torque limit cycle logic and calculates a jet on time required to enter a symmetrical limit cycle with a disturbance torque of +.001. The system will switch back to total minimum impulse limit cycle operation in one or two cycles and then the sequence will repeat. Had the system not switched into the disturbance torque limit cycle logic, the minimum impulse limit cycles would have gradually shifted upward along X_2 and eventually triggered the large error convergence logic.

Figure 13 illustrates the operation of the disturbance torque limit cycle logic when a change occurs in the disturbance torque. In this case, the disturbance torque changes from +.1 to -.01 at the end of a firing sequence. Notice that the limit cycle trajectory immediately begins to diverge and crosses the R1 switching circle in quadrant III. The symmetrical limit cycle logic now targets for a point on the D = -.01 symmetrical limit cycle. The trajectory overshoots the target point which results in an additional jet firing but then the system does enter the D = -.01 symmetrical limit cycle.

Concluding Remarks

This study has established and verified design concepts for an on-orbit TVC DFCS with OMPS engines fixed. Additional simulation will be required to establish (1) the suitability of the design for coupled 3-axis operation, (2) numerical values for the various constants and deadbands based on the actual performance of the orbiter (i.e., nonnormalized system) and (3) projected ACPS propellant utilization for this type of attitude control for specific SSV orbiter configurations to generate trade information regarding weight effectiveness of this control scheme versus hydraulically gimballed engines.

The fixed engine TVC logic presented herein has the unique features of (1) simple logic for disturbance torque estimation, (2) extremely simple logic if control action is not required,(3) limit cycle logic that provides "integral error compensation" for external disturbances, and (4) simple logic for selecting one of three possible control modes.

TABLE I.- SIMULATION CONSTANTS

K	=	1
R1	=	.25
R2	=	.3
DMIN	=	.0033
THRESHOLD	=	.045

REFERENCES

1. MIT Report Number R-567 (Rev. 3), "Guidance System Operations Plan for Manned LM Earth Orbital and Lunar Missions Using Program LUMINARY 1C (Rev. 130)," October 1969.

2. LEC Technical Report, "Space Shuttle Guidance, Navigation, and Control Functional Simulator," (Three volumes), January 1971.

3. NASA MSC Internal Note EG-71-7," Fuel-Time Optimal Control of a Double Integrating Plant," March 22, 1971.

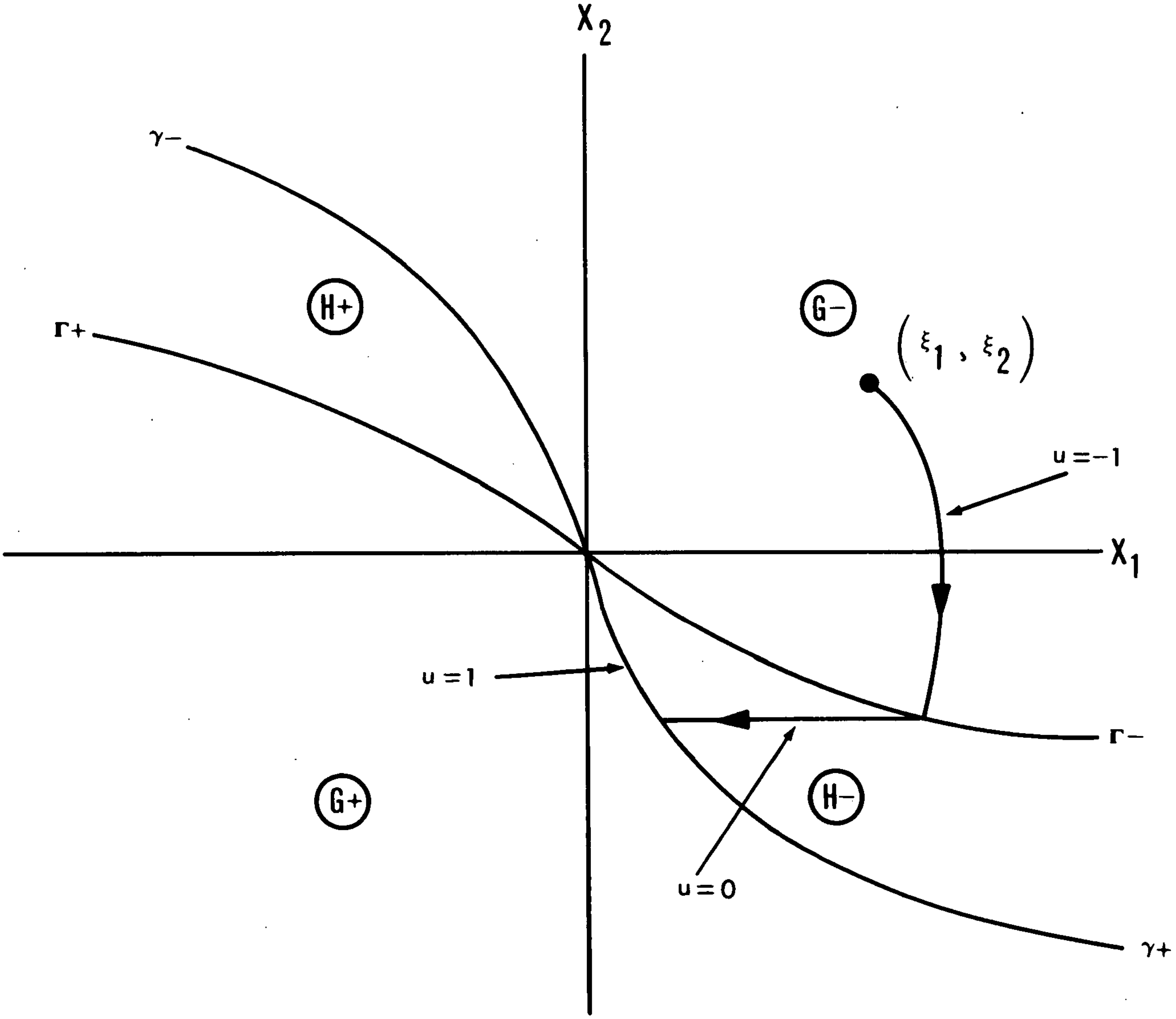


Figure 1.- Phase plane switch curve.

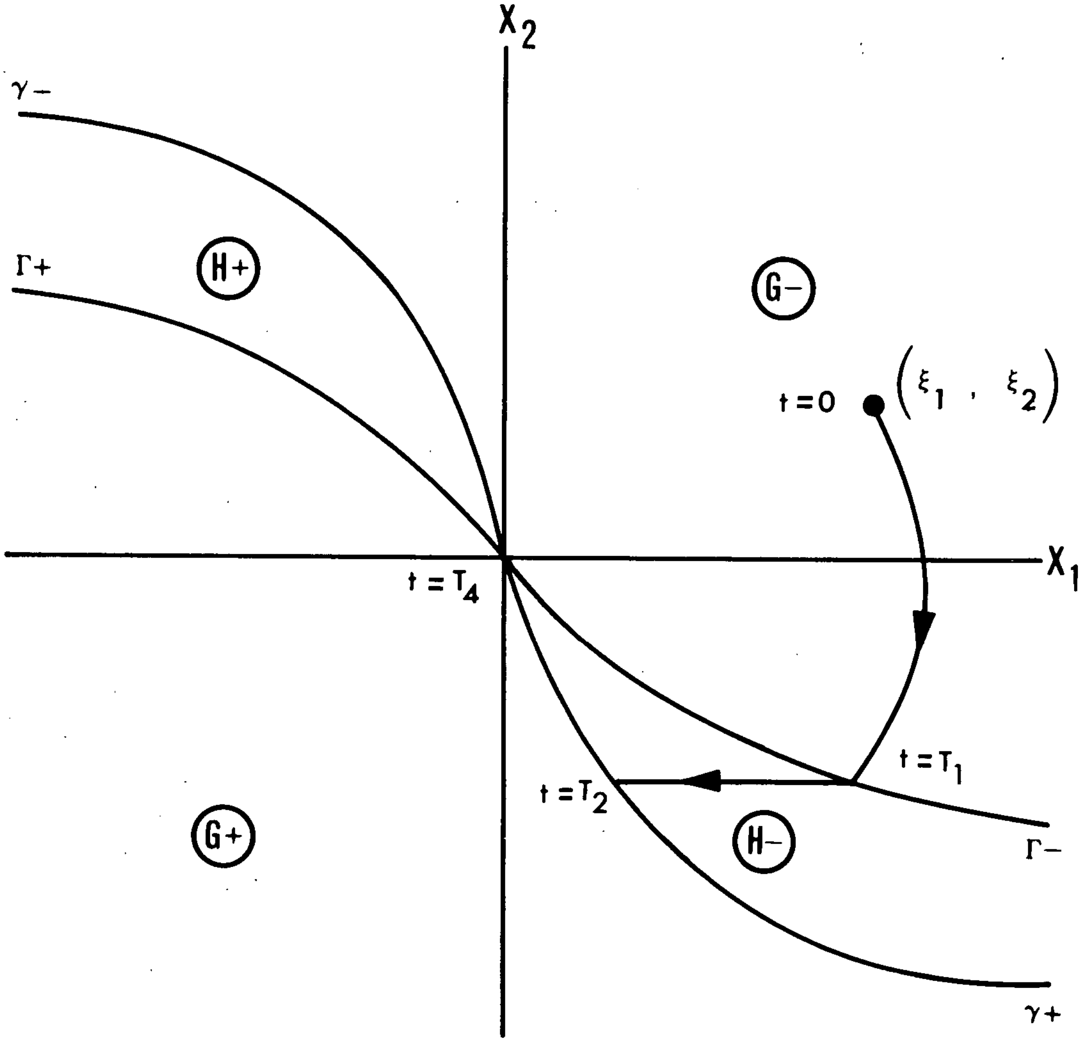


Figure 2.- Phase plane switch times.

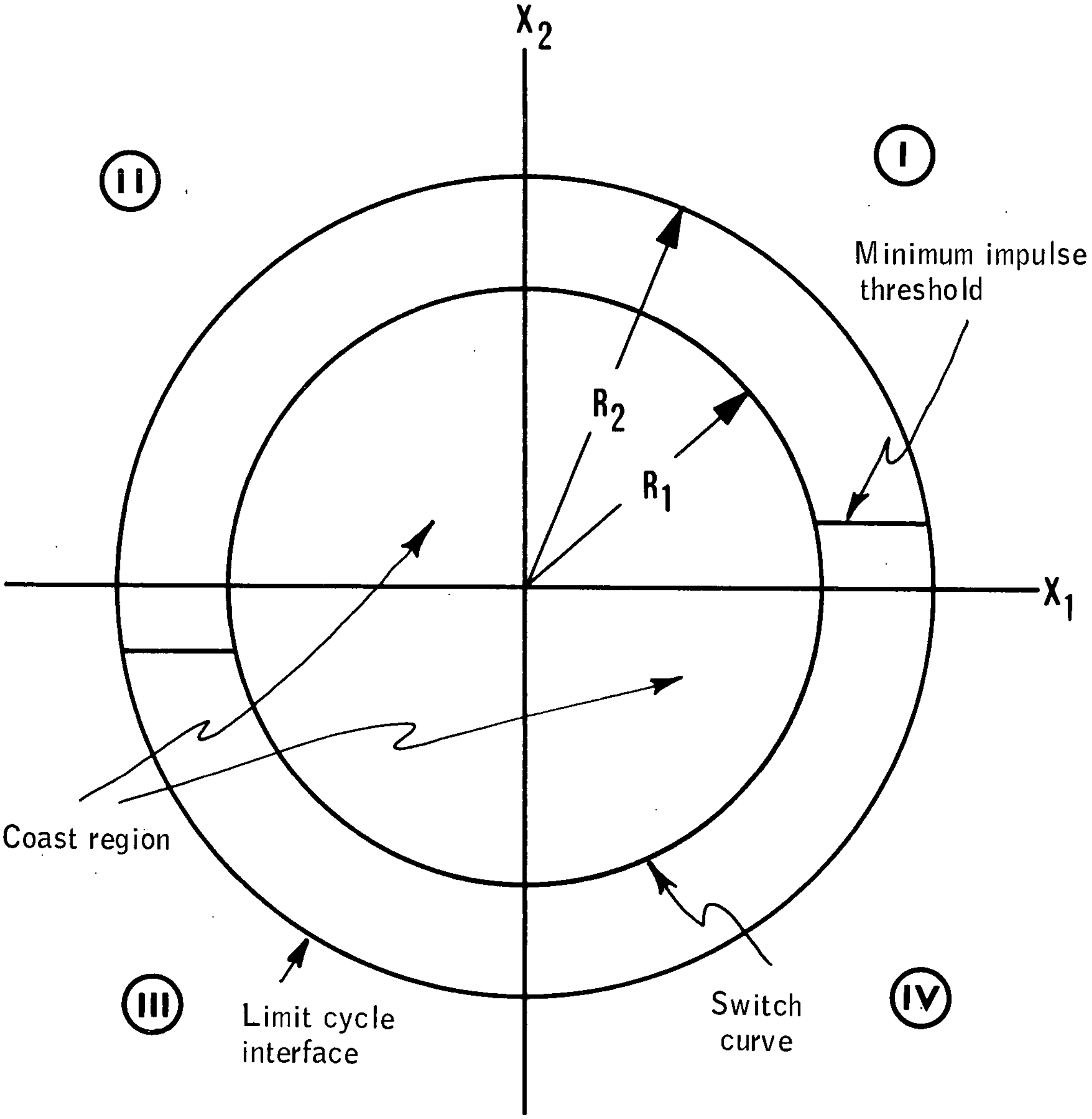


Figure 3.- Limit cycle design.

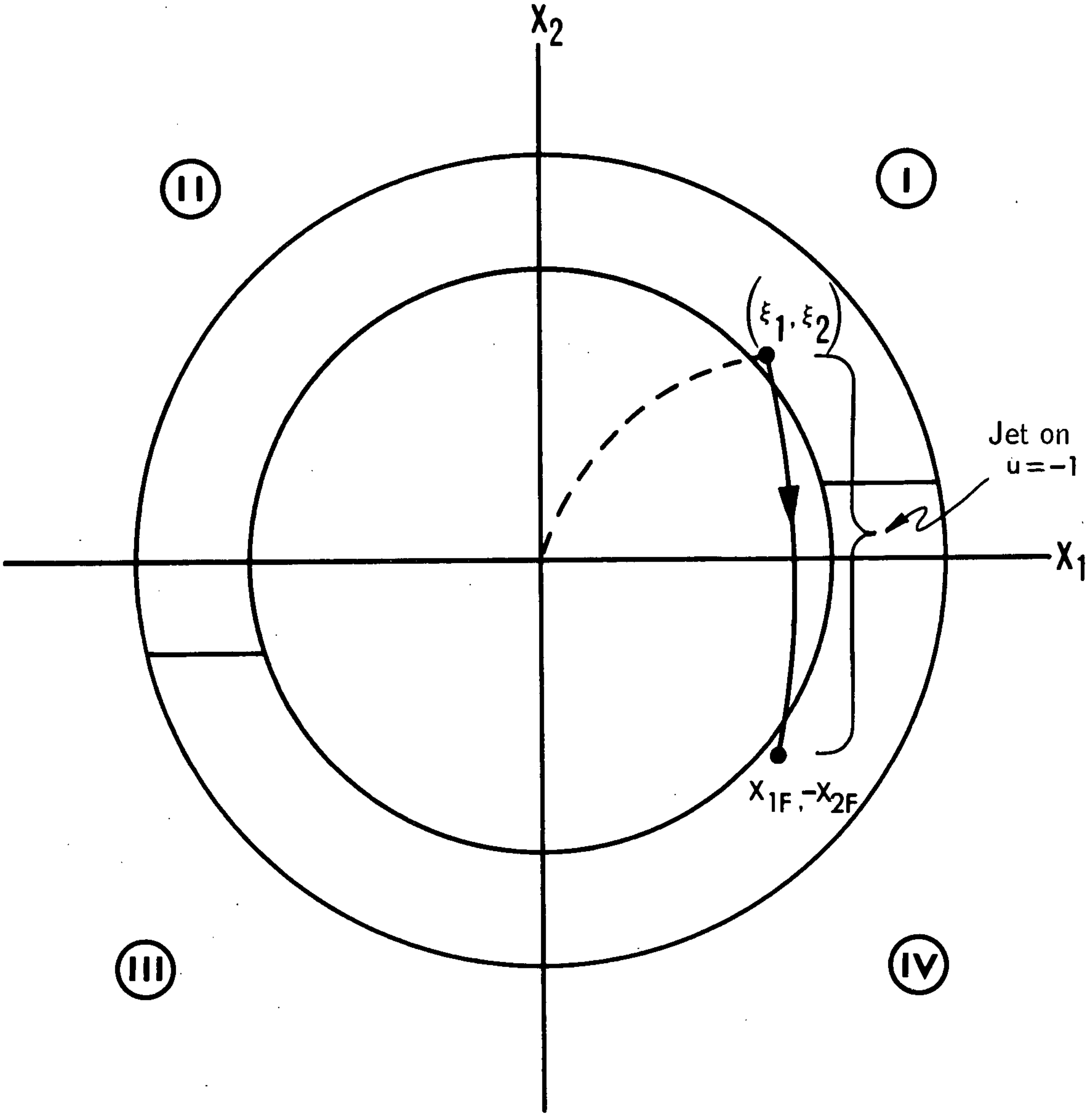


Figure 4.- Disturbance torque limit cycle.

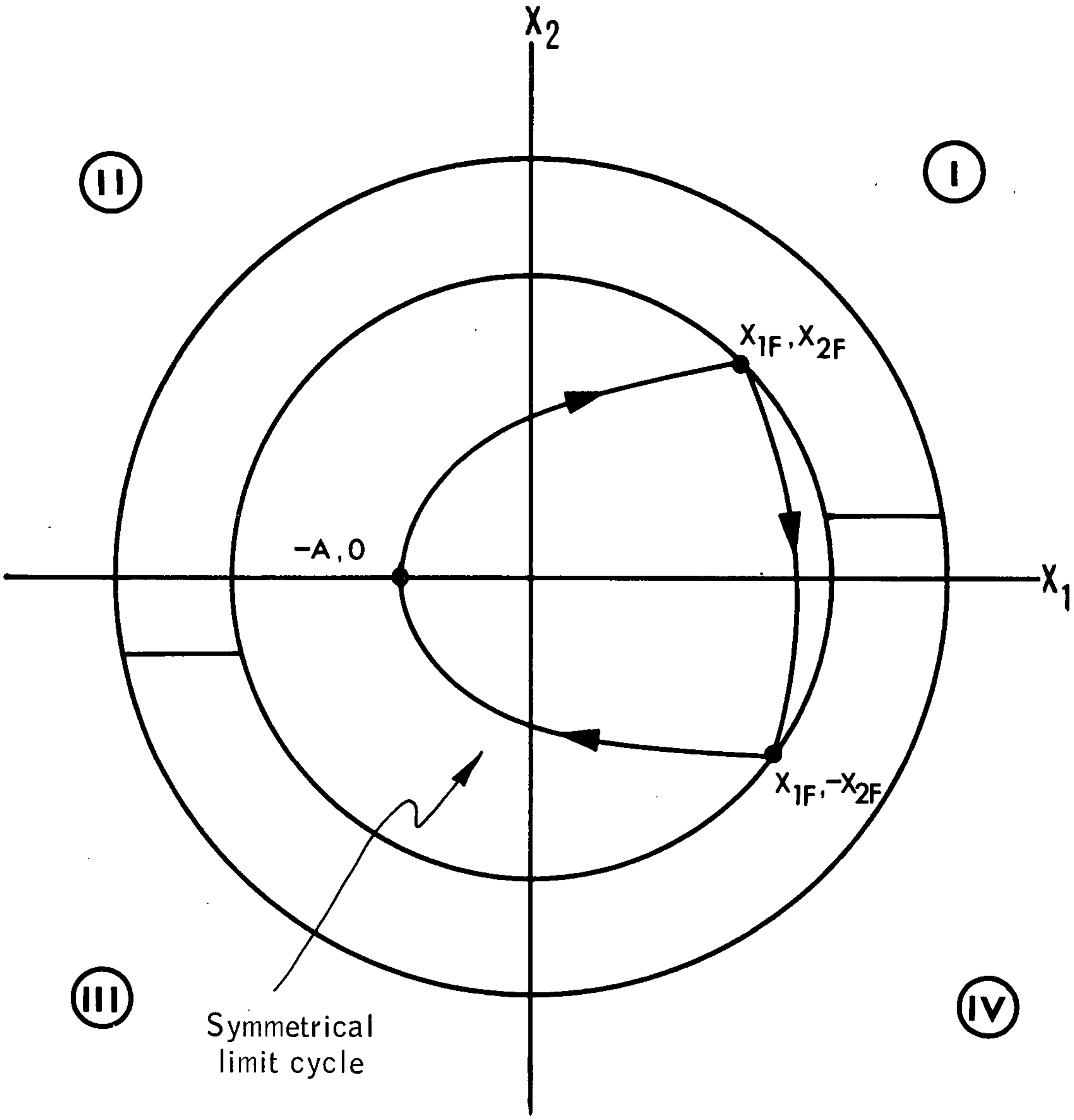


Figure 5.- Symmetrical limit cycle.

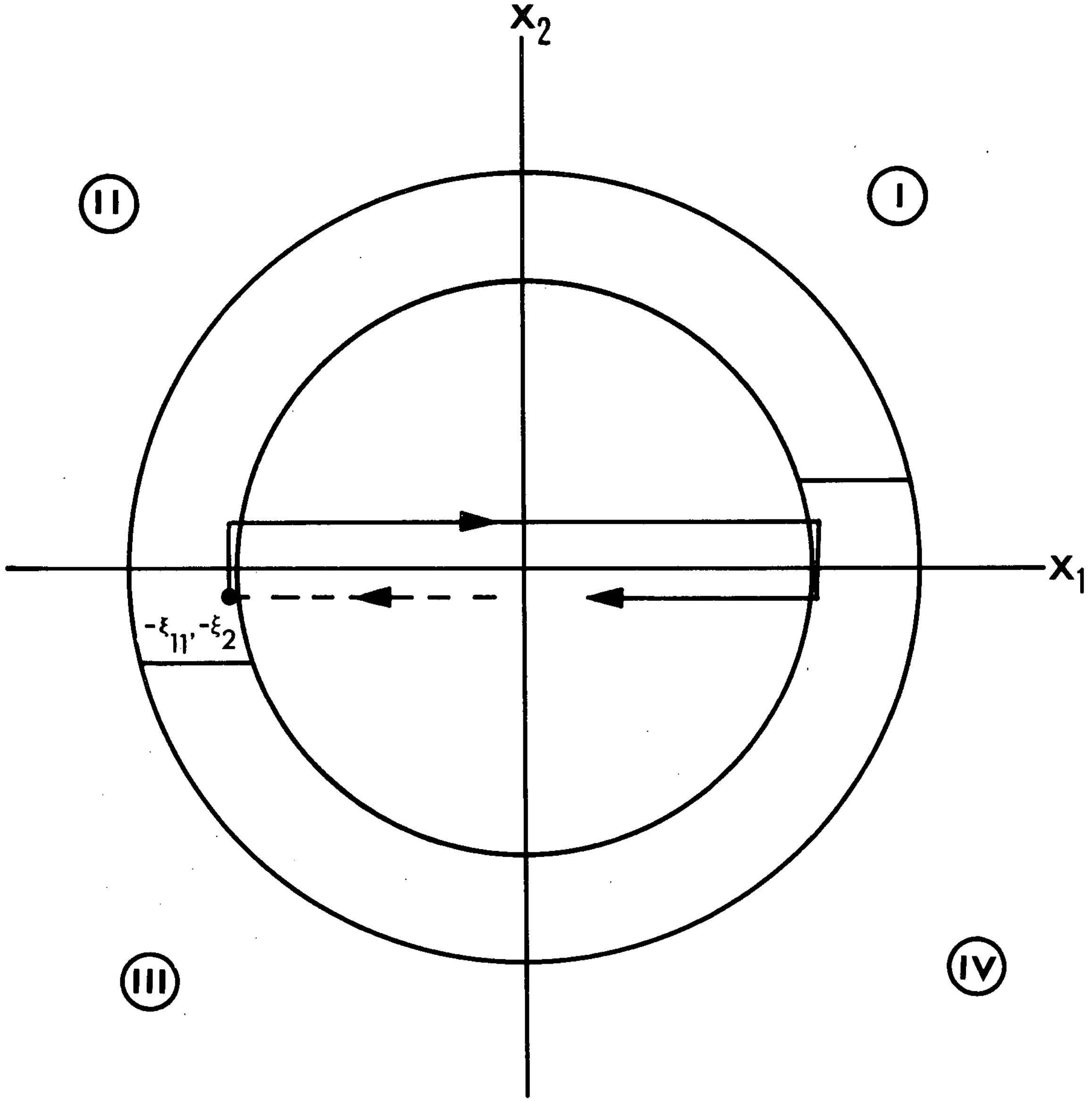


Figure 6.- Minimum impulse limit cycle.

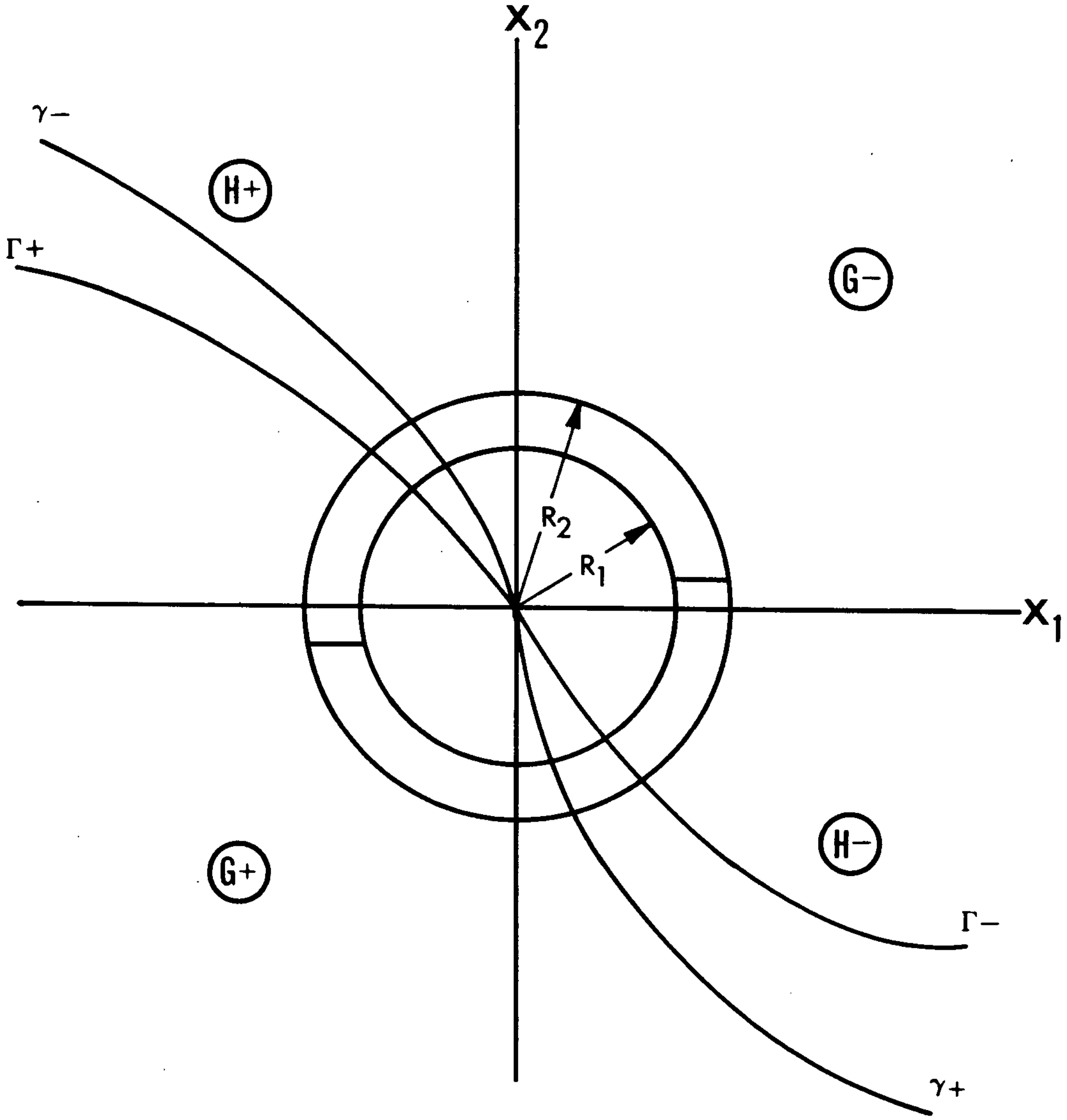


Figure 7.- TVC DFCS.

9.7.4.1 TVC, Fixed (cont'd)

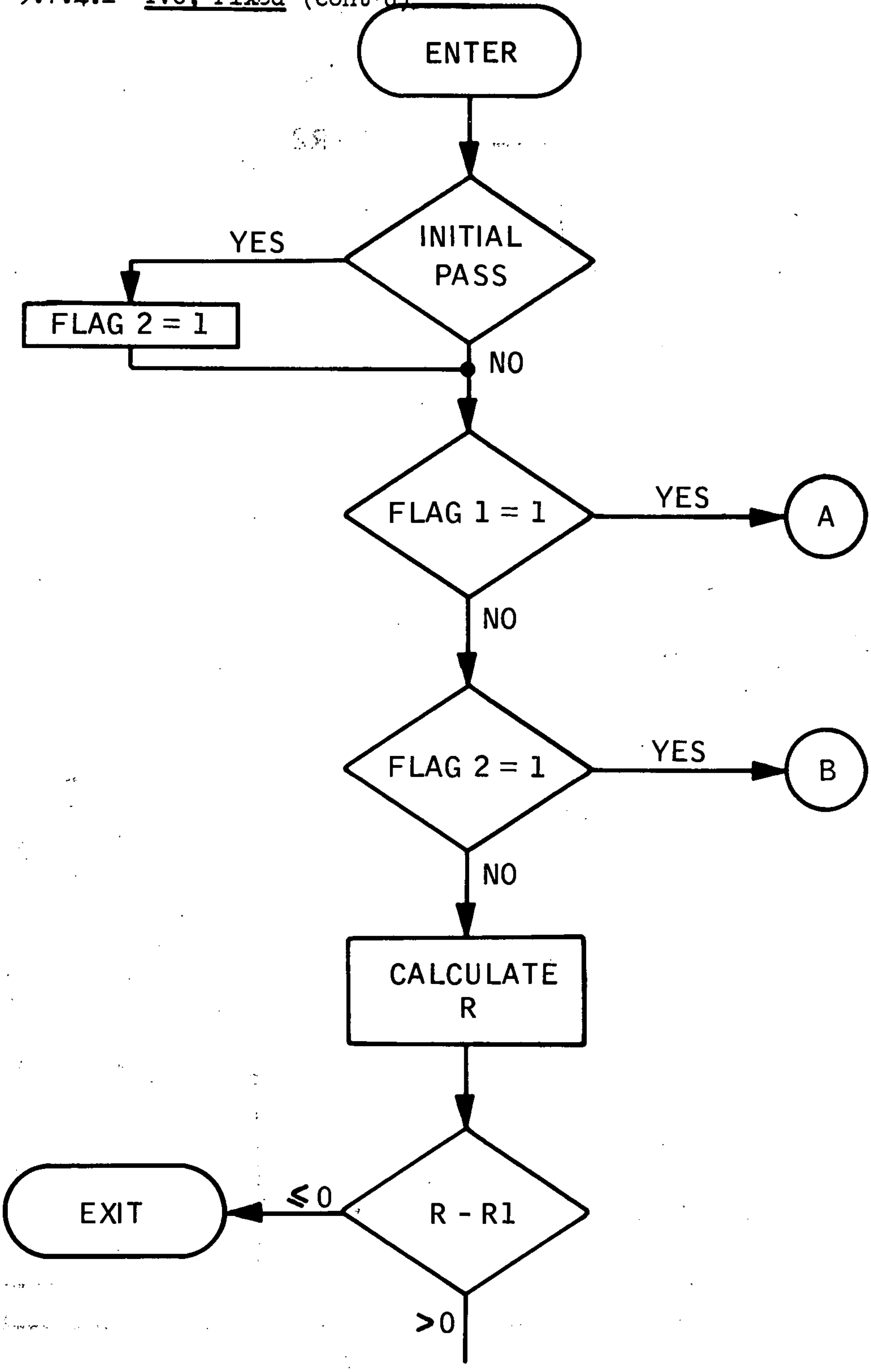


Figure 8a.- Functional flow chart.

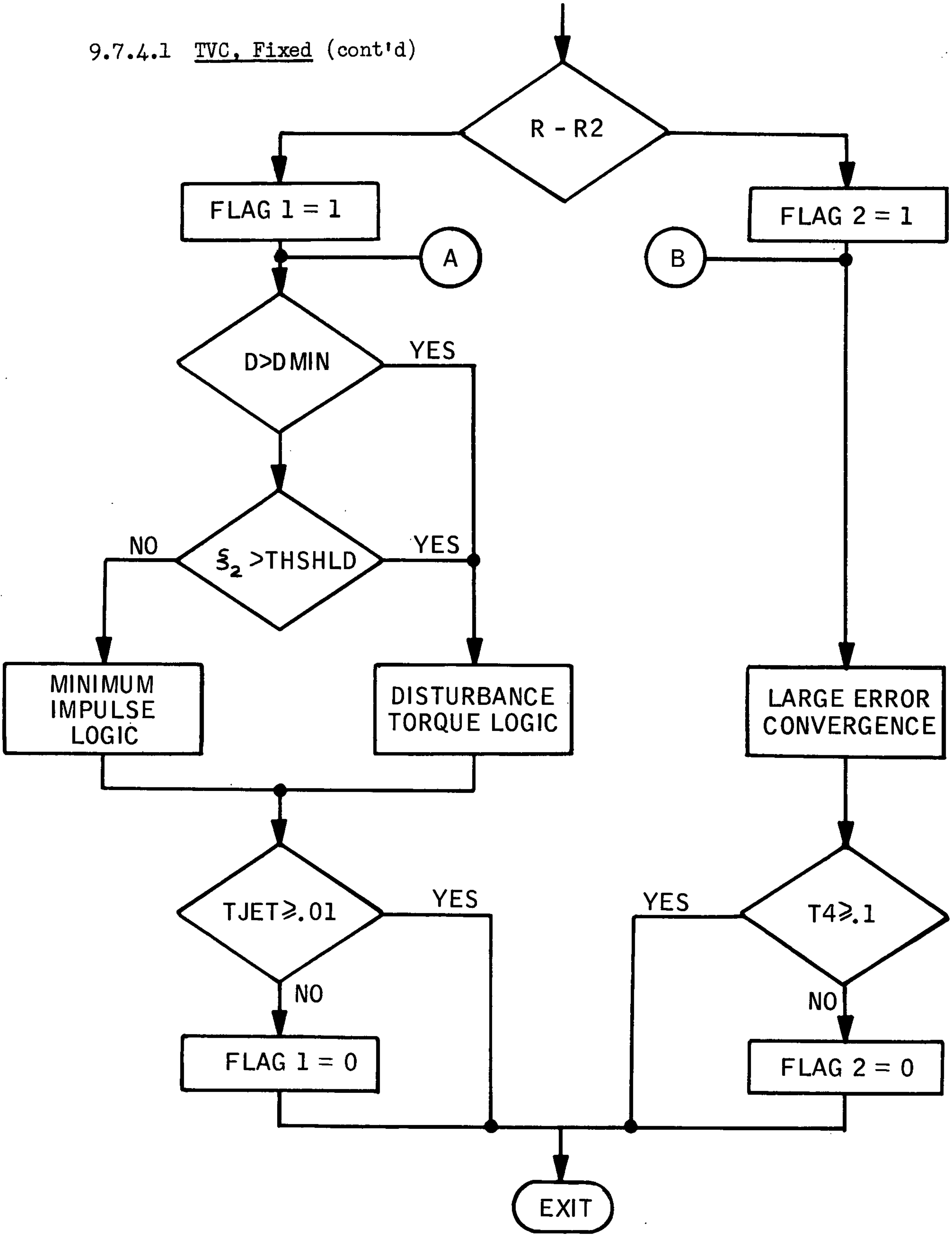


Figure 8b.- Functional flow chart.

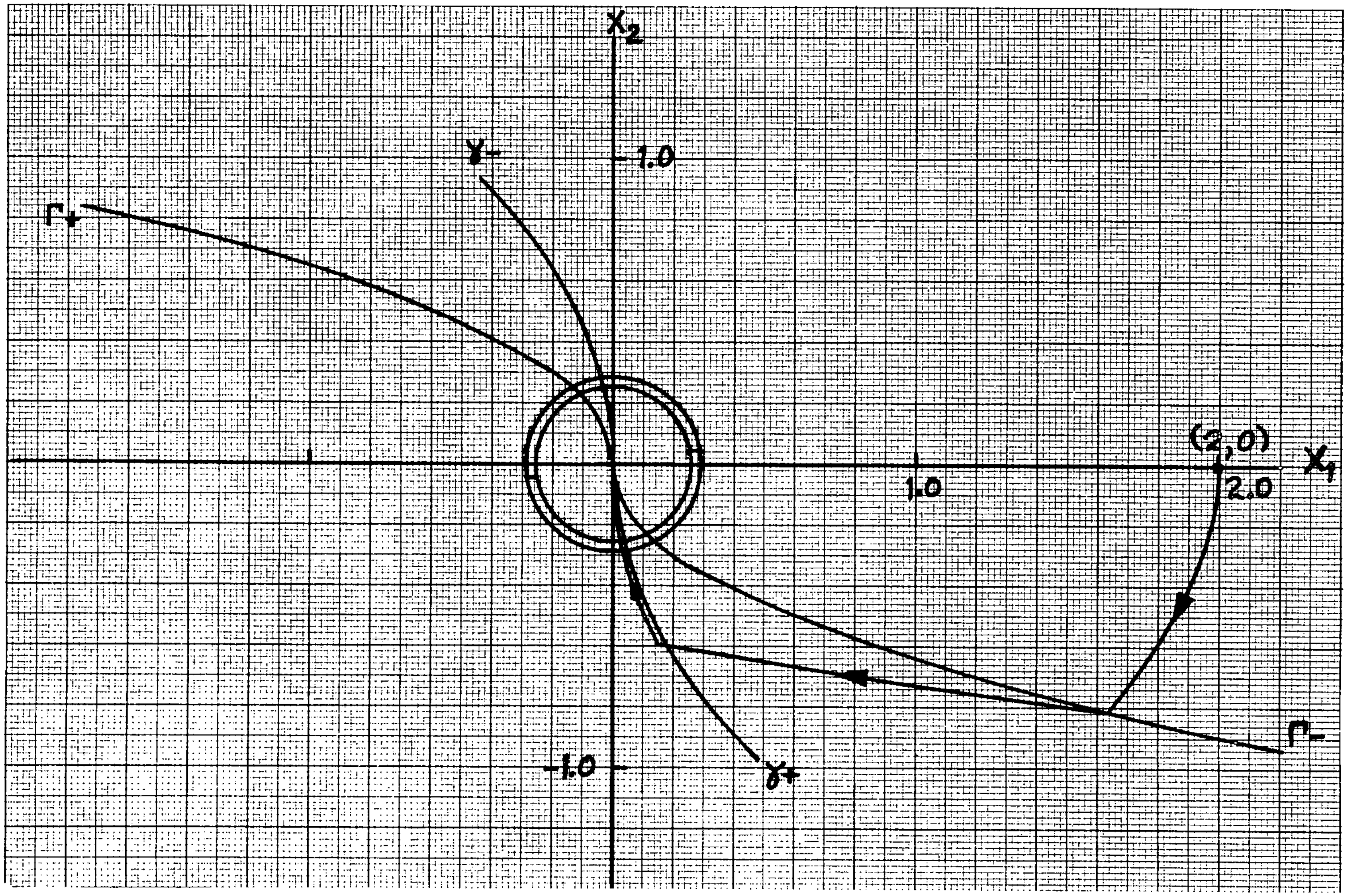


Figure 9.- Large error convergence -D = +.1

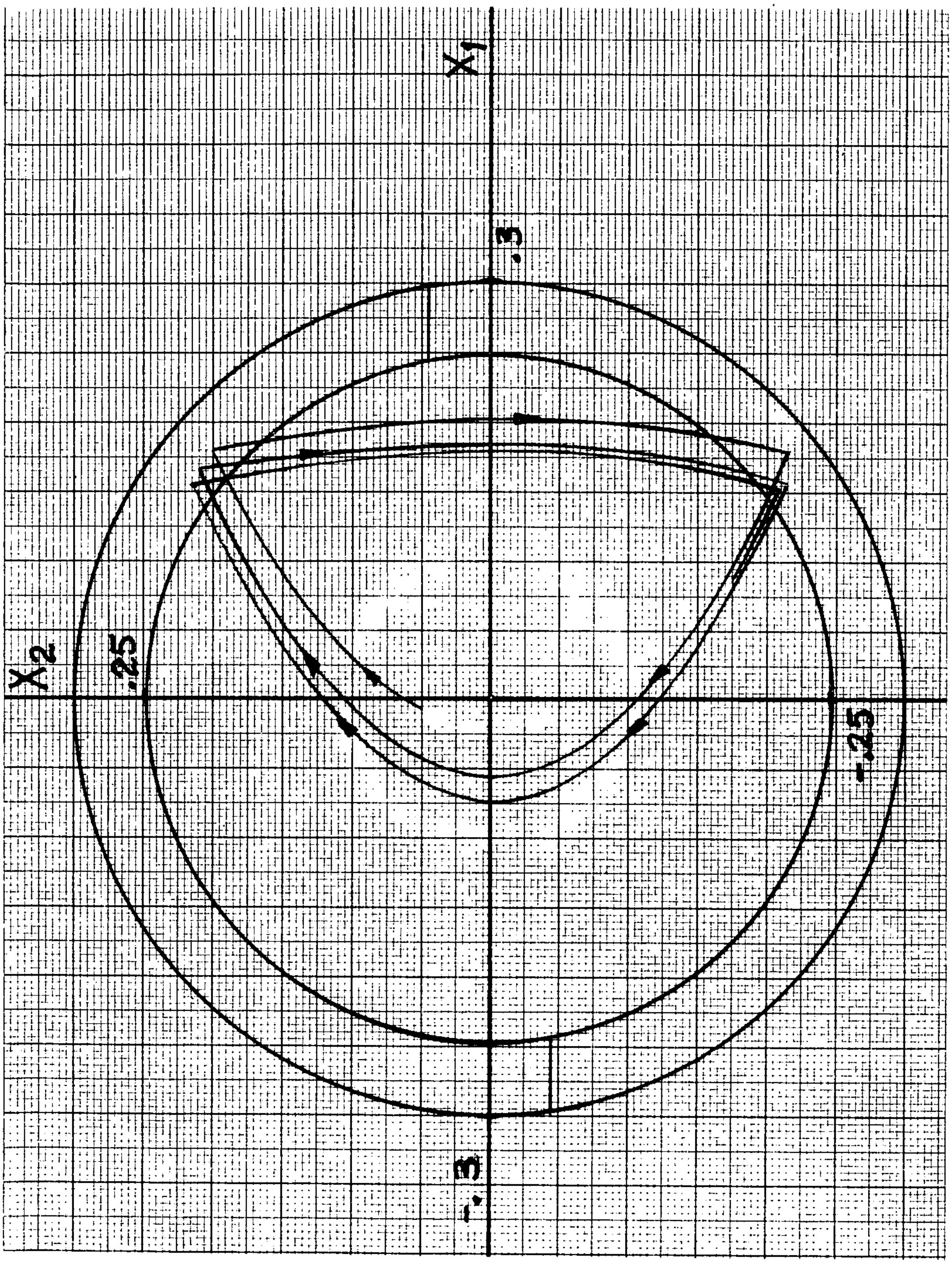


Figure 10.- Disturbance torque limit cycle - D=+.1

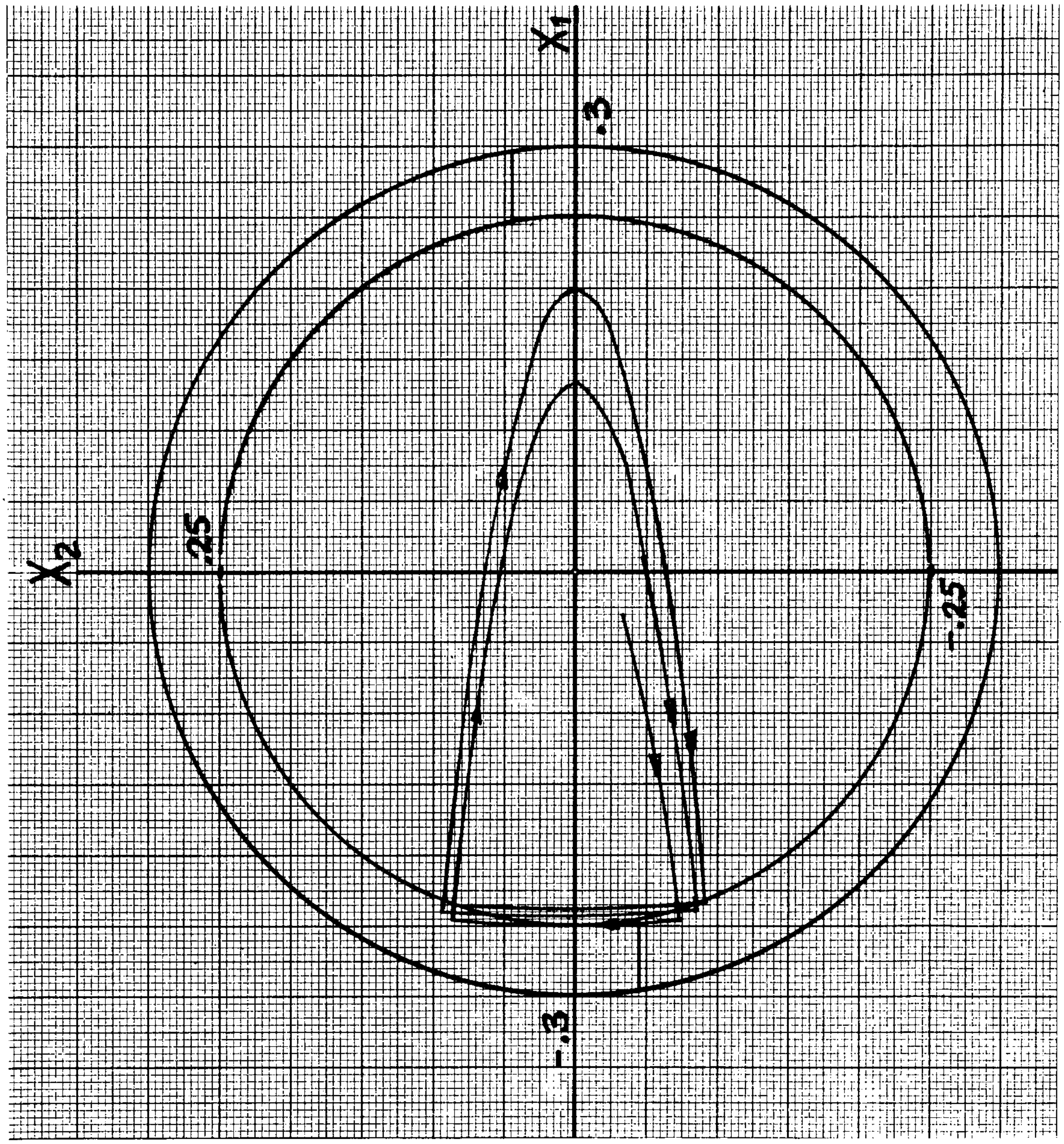


Figure 11.- Disturbance torque limit cycle - D=-.01

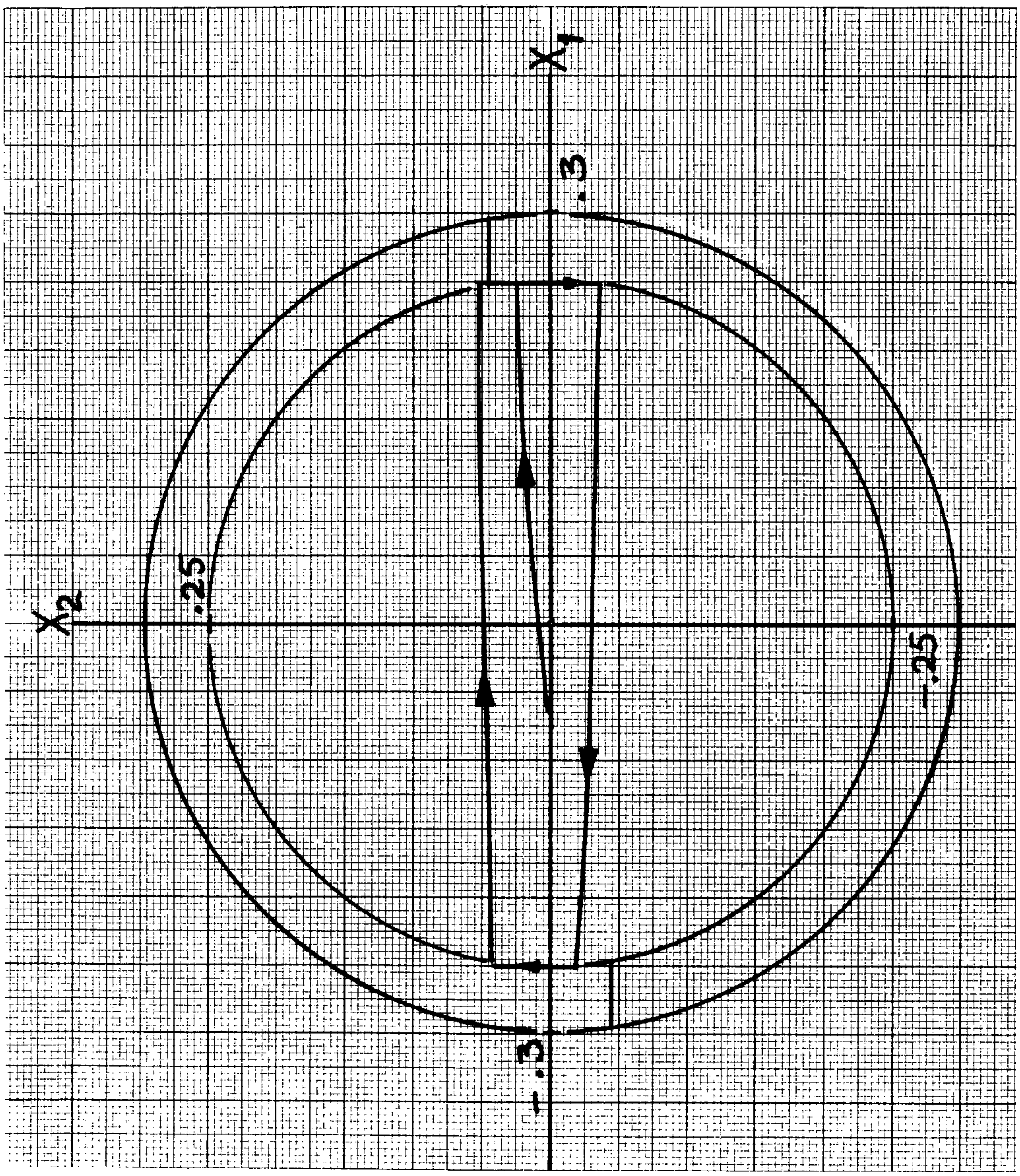


Figure 12.- Minimum impulse limit cycle - D=±.001

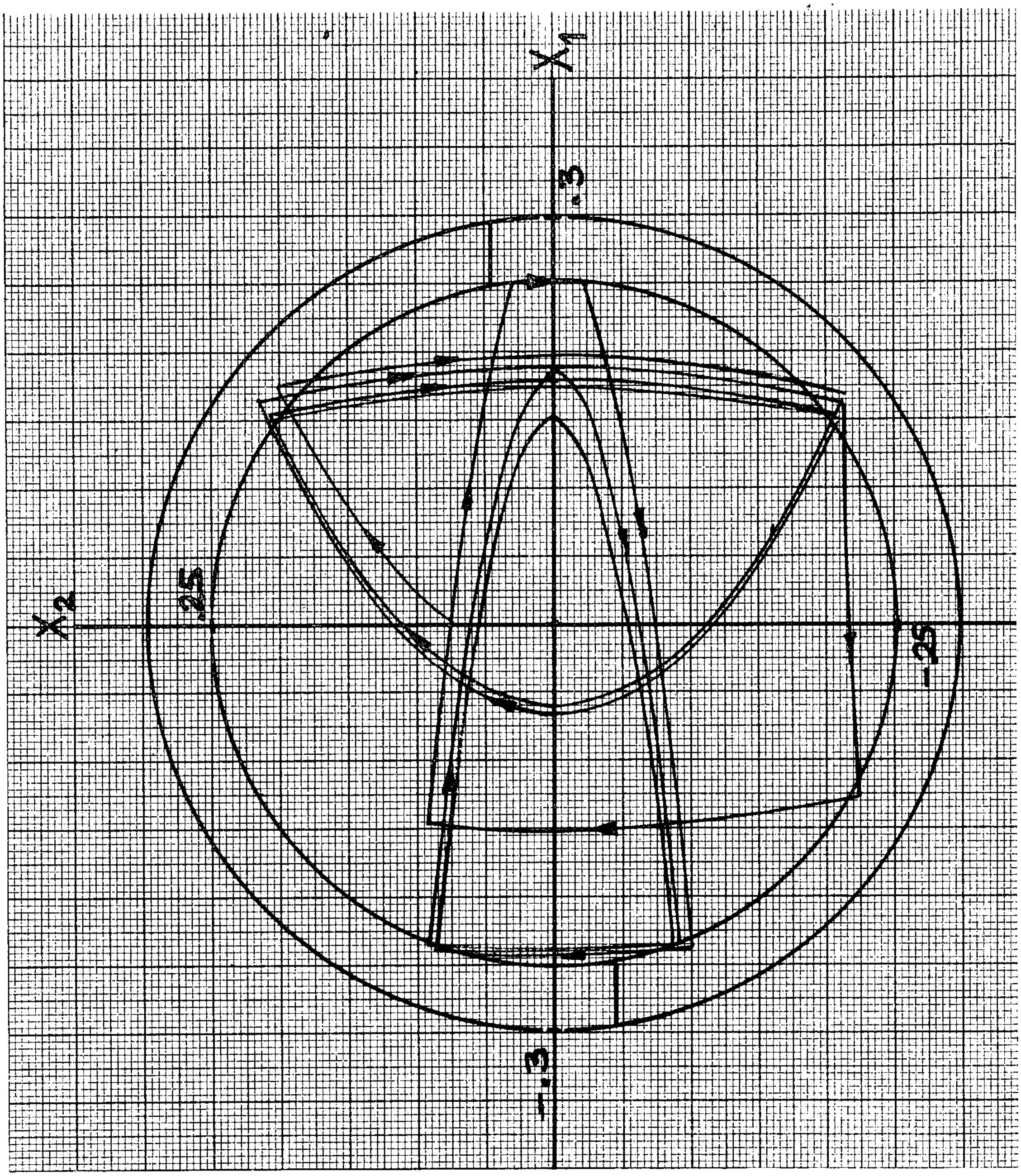


Figure 13.- Disturbance torque limit cycle -D = +.1 to D = -.01

9.7.4.2 TVC, Gimballed

PRECEDING PAGE BLANK NOT FILMED

SPACE SHUTTLE

GN&C SOFTWARE EQUATION SUBMITTAL

Software Equation Section Orbital Powered Flight Submittal No. 32

Function Orbital TVC Autopilot (Gimballed OMPS)

Module No. OC-3 Function No. 1,2 & 5 (MSC 03690)

Submitted by: T. Lee Co. TRW (MSC-05151)

Date: 8/24/71

NASA Contract W. H. Peters Organization GCD

Approved by Pannel III K.J. Cox (chairman) Date 8/24/71

Summary Description: Thrust control of Orbiter using gimballed engines alone. Task complicated by extremely low performance actuator. Control scheme includes (non-baselined) guidance technique, which is presented for information.

Shuttle configuration: (Vehicle, Aero Data, Sensor, Et Cetera)

The design is constrained by NR 161-C vehicle and associated actuator.

Comments:

(Design Status)

(Verification Status)

Panel Comments:

1. INTRODUCTION

The subject of this report is a preliminary design for control of the Shuttle Orbital Maneuvering Propulsion System (OMPS) gimballed engines during orbital burns. For the preliminary design, it was assumed that the gimballed engines will provide all the required control. Other types of thrust control will be considered in later designs; these will include reaction-jet control and a combination of reaction-jet and gimballed-engine thrust vectoring.

Reference 1 supplied the data on vehicle configuration and OMPS actuator performance boundaries used in this study. One of the objectives of the preliminary design is to evaluate the configuration and determine if any serious control problems result from the vehicle design. The subject configuration (NR 161-C) creates several such problems which will be discussed below.

One of the groundrules assumed for this study is that the Thrust Vector Control (TVC) system be compatible with External Delta-V guidance as defined for Apollo. This type of guidance computes the velocity increment required based on the orbital parameters desired at the completion of the burn and the current parameters determined by navigation. The velocity gain (V_g) desired is computed once only for the entire burn and entered into the computer as an external command prior to the scheduled ignition time. The vehicle attitude is aligned by the reaction jets to the direction determined by the computer from the direction of the V_g vector and the estimate of trim thrust direction relative to the vehicle. During the burn, the V_g vector is updated periodically by sensed acceleration for the purpose of determining engine cut-off time and for commanding the TVC to remove cross-axis (lateral) velocity errors that have accumulated. Elaborate preburn alignment and frequent velocity updates yield greater burn accuracy (or lower TVC requirements) when coupled with External Delta-V guidance because the target velocity is not changed during the burn. The TVC is designed for sensitivity to velocity errors and cannot cope with large transients or abrupt sign reversals in the V_g commands which result from other types of guidance.

The basic task of the TVC system is to null the accumulated cross-axis velocity errors during the burn. An opposing cross-axis acceleration is required to null the velocity errors. The thrust vector is controlled by rotating the vehicle until the required acceleration is obtained. The vehicle steering in this Shuttle design differs from the Apollo SPS TVC in two significant respects; steering commands are proportional to the cross-axis velocity and no direct attitude control is employed. These changes from Apollo will improve the end-of-burn steering loop stability and reduce residual velocity errors due to pointing errors.

Attitude stabilization and steering rates are achieved by a rate control loop which issues deflection commands to the actuator position control. Included in the actuator commands are estimates of trim deflections which are continually updated during the burn so that rate biases are eliminated.

In the remaining sections of this report, the results associated with the preliminary design will be reported. The first topic will be the basic concept of the control system and the reasons for selecting it. The second section will present the vehicle mass properties and actuator performance data used as inputs to the study. The modeling of the actuator dynamics will then be treated in detail because of the effects of actuator nonlinearities on the total design effort. The software design will be described in two sections; the first of these sections will be devoted to the rate loop, and the second to the accelerometer loop. The last section reporting results will estimate the sensor requirements that would be consistent with the system developed. The final sections of this report will consist of the software equations for the preliminary TVC design and a review of the study progress to date.

9.7.4.2 TVC, Gimballed (continued)

LIST OF SYMBOLS

Subscripts:

c	commanded (desired)
m	measured -
Z	cross axis (pitch)
$(\dot{\ })$	derivative with respect to time
$(\underline{\ })$	vector

A	Acceleration (see Subscripts)
ACPS	Attitude Control Propulsion System
DAP	Apollo control system
E, E_P	ACPS control error
E_A	Acceleration control error
I	Moment of inertia of vehicle
K	Actuator forward loop gain
K'	Trim bias integrator gain
K_A	Acceleration loop gain
K_F	Steady-state gain of rate filter
K_S	Velocity error gain
K_R	Actuator rate feedback gain
$K_{\dot{\theta}}$	Frequency-dependent gain of rate filter
ℓ	Thrust control moment arm
ℓ_A	Acclerometer moment arm
LVDT	Linear Variable Differential Transducer; converts actuator displacement to deflection angle
m	Vehicle mass
OMPS	Orbital Maneuvering Propulsion System
s	Laplace operator

LIST OF SYMBOLS (continued)

SPS	Apollo Service Module Propulsion System
t	Elapsed time
T	Thrust
TACH	Tachometer; measures actuator displacement rate
V_g, V_G	Velocity-to-be-gained (see Subscripts)
$(V_G)_{ZERO}$	Preburn Delta-V command
ΔV	Velocity gained during burn (see Subscripts)
α	Thrust vector angle
γ	Velocity error vector angle
δ	Engine delfection angle (see Subscripts)
δ_{cg}	Actual cg-trim engine deflection angle
δ_{EST}	Preburn trim estimate
δ_{PK}	Peak engine deflection angle
δ_{TRIM}	Updated trim estimate
$\Delta\delta$	Control deflection command
ϵ_A	Acceleration error
$\epsilon_{\dot{\theta}}$	Vehicle rate error
θ,$\Delta\theta$	Vehicle pointing angle (see Subscripts)
ξ	Thrust vector angular error
τ	Time constant of rate filter
ϕ	Phase shift or phase angle
ω	Frequency of sinusoidal input

2. BASIC CONCEPT

The basic concept for the preliminary TVC design is shown in the simplified pitch plane block diagram of Figure 1. The software is divided into two sections, steering and rate stabilization. The actuator electronics is depicted as an analog computer and torque generator. The actuator dynamics complete the feedback loops to the two actuator sensors, and the vehicle dynamics model closes the rate gyro and accelerometer loops to the software interface. Three other inputs to the software are shown in the block diagram; they are the preburn guidance commands and trim estimates and the manual steering commands. Gains and filter coefficents can be treated as preburn input constants.

The guidance commands in the cross-axis direction are always zero for External Delta-V. V_z is the cross-axis velocity error as shown in Figure 2. Cross-axis velocity results from errors in pointing the thrust vector along the initial V_g vector. As the cross-axis velocity accumulates, the V_g vector rotates thru an angle γ in Figure 2. In order to reduce the cross-axis velocity (V_z), the thrust vector must be rotated to an angle above the initial V_g vector so that an acceleration component, A_z, is obtained in the same direction as V_z. However, the time remaining for taking out V_z is approximately V_g/A (time-to-go). The time required to null V_z is potentially V_z/A_z; in terms of the angles γ and α, the thrust vector angular error is very nearly (γ-α). Apollo steering is based on the error angle (γ-α) as a result of the steering law:

$$\underline{\omega}_C = K_S\,(\underline{1}_A \times \underline{1}_{VG})$$

This cross product of unit vectors along the $\underline{A}$ and $\underline{V}_g$ vectors yields a pitch component of steering commands as follows:

$$\dot{\theta}_C = K_S \text{ Sin } (\gamma - \alpha)$$

FIGURE 1

BLOCK DIAGRAM OF BASIC CONCEPT

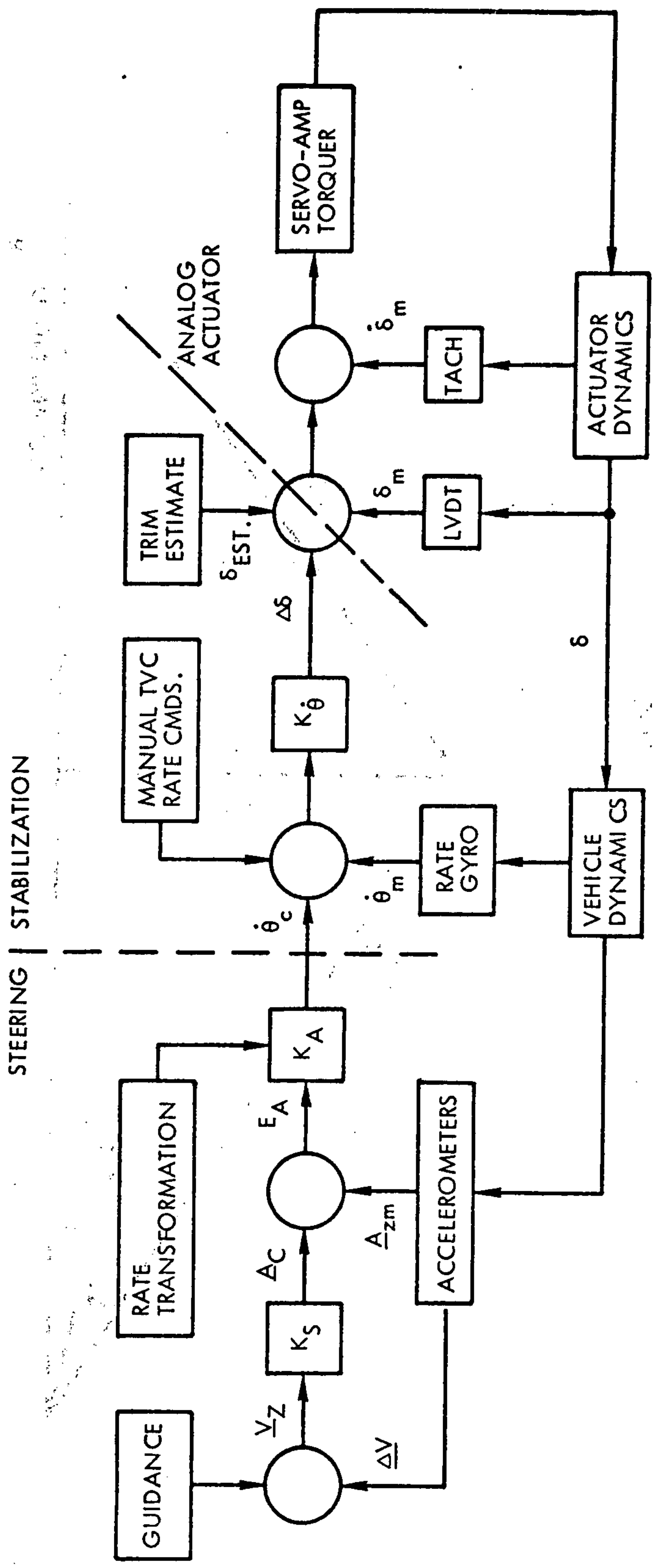
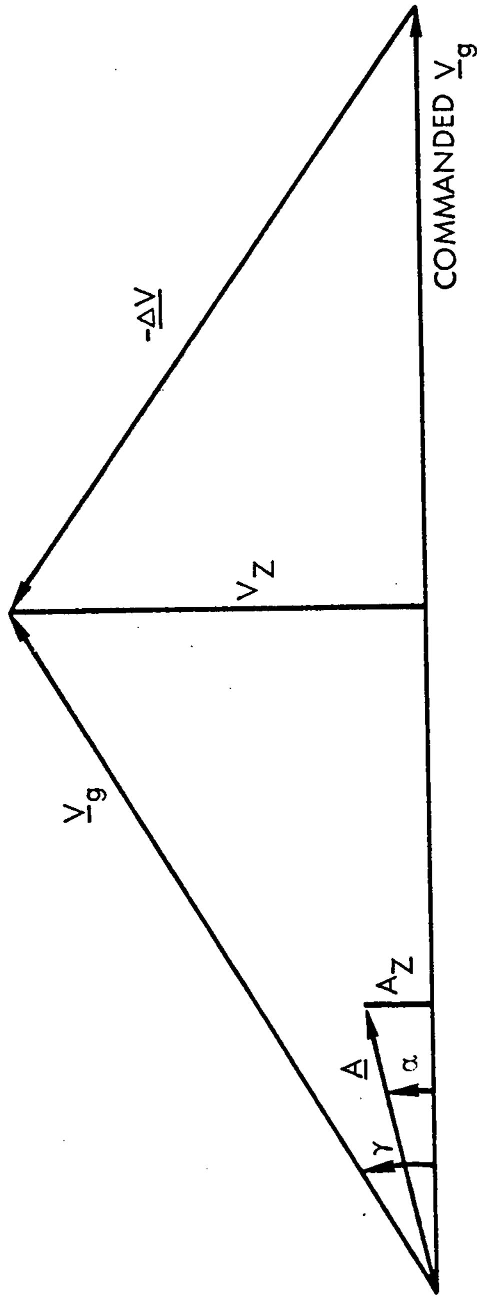


FIGURE 2

STEERING CONCEPT



The parameter we would like to null is the cross-axis velocity, and the steering response to V_z can be obtained by substitution for γ and α using Figure 2; that is:

$$\dot{\Theta}_C = K_S (V_z/V_g - A_z/A)$$

$$(A/K_S) \ \dot{\Theta}_C = (A/V_g) \ V_z - A_z \quad \text{where } A = \text{Thrust/mass}$$

From the last equation we determine that the Apollo steering gain is inversly proportional to the time-to-go before engine shut-down.

The variable gain resulting from the Apollo steering design creates several problems which can be avoided with the "constant-gain" steering in Figure 1. From Figure 1, the equation

$$(1/K_A) \ \dot{\Theta}_C = (K_S) \ V_z - A_z$$

is comparable to the Apollo equation derived above, which can be written

$$(1/K_S)(T/m) \ \dot{\Theta}_C = (T/m)(1/V_g) \ V_z - A_z$$

By adjusting the gains in the proposed shuttle steering equation, the two equations could be identical except for the variations in V_g. At the start of a long burn, the Apollo steering is very slow because of the low gain; most of the steering rate is due to vehicle pointing errors and engine gimbal transients. Four seconds before engine cut-off, the steering gain is too high, and better end conditions are achieved by commanding zero rates. The Apollo steering gain at 6 seconds from cut-off is 1/(6 seconds) or 0.167/seconds.

The basic idea behind the shuttle TVC concept is to provide the highest steering gain consistent with stability throughout the burn, including the last 4 seconds. The shuttle steering will be dominated by attitude and gimbal transients during the first few seconds of the burn; this early response is the same as in Apollo burns and results from the accelerometer feedback, A_z. During the mid-burn period the shuttle steering will produce an exponential decay in V_z and the control response will decay proportionally. At the end of the burn, the velocity residuals should be small, and the control system should respond slightly to these errors.

Referring again to Figure 1, the shuttle concept has no redundant attitude control loop. The Apollo DAP contains an attitude-hold feature which opposes the change in attitude required for steering. The steering error (E_A in Figure 1) is proportional to the thrust vector pointing error, which is the only "attitude error" needed to command the correct vehicle rates. The pointing of the vehicle is otherwise irrelevant to thrust vector control.

The gain K_A in Figure 1 controls the attitude response of the vehicle; the vehicle pointing error is the angular equivalent of the acceleration error (E_A) defined by the steering equation. The attitude error (E_Θ) can be computed from the equation:

$$E_\Theta = E_A/(T/m)$$

The equivalent "attitude loop gain" is:

$$\dot{\Theta}_C/E_\Theta = (T/m)K_A$$

The "control law" can be obtained by incorporating the rate loop gain as follows:

$$\Delta\delta_C = (T/m)(K_A)(K_{\dot{\Theta}})\ E_\Theta + (K_{\dot{\Theta}})\ \dot{\Theta}$$

Too much emphasis should not be placed on the "control law" as a representation of any control system. Aerospace vehicle control systems require D.C. bias compensation, signal shaping for stability, and filtering for noise; that is, $K_{\dot{\Theta}}$ in the above equation is a frequency-dependent gain or "filter".

The output of the software is a vehicle angular acceleration command converted to a total engine deflection angle. Because the angular acceleration is not necessarily zero for a null deflection angle, a trim deflection bias, δ_{trim}, is added to the control deflection. Before the burn, the trim bias is estimated; then, the estimate is updated during the burn by integrating the acceleration commands and adding small increments to the trim bias.

In the roll axis, this study assumes no requirement for steering or attitude control. A simple roll rate damping control is provided using the pitch actuators. Low roll rates will prevent any coupling between pitch and yaw steering, and the small roll angle which may develop should not violate any antenna or window pointing constraints. The outputs of the software are pitch/roll and yaw deflection commands to the actuator electronics of the two OMPS engines.

In review of the preliminary Shuttle TVC design concept, the following features are listed:

1) Constant-gain steering is used to eliminate cross-axis velocity errors early in the burn.

2) Attitude control is based on thrust vector errors only.

3) A trim integrator is provided to eliminate command biases.

4) Simple rate damping is employed in the roll axis control.

In the following paragraphs, the development of this concept into a preliminary design will be discussed.

3. CONFIGURATION DATA

The mass properties and OMPS engine data required for completing the preliminary TVC design were selected from that given in Reference 1. This data is presented in Table I for orbital maneuvers. Table II gives the same data after it has been manipulated into the required form. A block diagram of the vehicle dynamics appears in Figure 3.

Because of the small variance in vehicle dynamics, it was not necessary to vary gains or filter coefficients. The nominal dynamics gains in Table II were used to derive the following transfer functions required for the sensor feedbacks:

$$\dot{\Theta}/\delta = -0.06/s \text{ per sec}$$

$$A_{zm}/\dot{\Theta} = \frac{-0.1207\ (s^2 - 0.314)}{s} \quad \frac{\text{ft/sec}^2}{\text{deg/sec}}$$

The actuator limits given by Reference 1 are as follows:

$$\delta \text{ limit} = \pm 4°$$

$$\dot{\delta} \text{ limit} = \pm 7 \text{ deg/sec}$$

$$\ddot{\delta} \text{ limit} = \pm 35 \text{ deg/sec}^2$$

The impact of these actuator limits on the control design will be discussed in the following section.

The bending frequency range is 1.25 to 1.5 H_z according to Reference 1. No amplitude data was provided, so it was assumed that the bending frequencies would require an attenuation of 30 db. Slosh was not treated in the preliminary design; however, Apollo experience has shown slosh can be a difficult problem. The topic of slosh stabilization will be a major consideration during the final design phase of this study.

9.7.4.2 TVC, Gimballed (continued)

TABLE I - NR161-C MASS PROPERTIES

Payload	INSERTION			ENTRY		
	65K	40K	25K	65K	40K	25K
Weight (lbs.)	327,359	302,793	316,940	293,706	268,706	272,230
cg x (in.)	1710.8	1711.1	1721.8	1677.6	1674.2	1669.8
cg y (in.)	0.2	0.2	-1.5	0.2	0.2	-1.7
cg z (in.)	376.7	374.8	379.5	376.1	373.8	378.4
I_{xx} (slug-ft^2)	2.60×10^6	2.57×10^6	2.73×10^6	2.43×10^6	2.40×10^6	2.50×10^6
I_{yy} (slug-ft^2)	19.80×10^6	19.51×10^6	19.88×10^6	18.36×10^6	18.05×10^6	17.95×10^6
I_{zz} (slug-ft^2)	21.30×10^6	21.02×10^6	21.50×10^6	19.72×10^6	19.41×10^6	19.35×10^6

Engine locations (x, y, z) = (2399, ±53, 502) inches

Thrust per engine = 10,000 pounds

C.6

TABLE II - VEHICLE DYNAMICS GAINS

	INSERTION			ENTRY		
Payload	65K	40K	25K	65K	40K	25K
m (slugs)	10,180	9,410	9,850	9,120	8,360	8,480
ℓ (ft.)	58.3	58.3	57.4	61.1	61.4	61.8
T (lbs.)	19,896 (2 outboard engines canted slightly inboard)					
Tℓ/I ($1/sec^2$)	0.0569	0.0564	0.0553	0.0624	0.0611	0.0615
T/m (ft/sec^2)	1.96	2.12	2.03	2.19	2.38	2.36

Nominal gains used for transfer functions:

$T\ell/I = 0.06/sec^2$

$T/m = 2.17\ ft/sec^2$ (or $0.0379\ ft/sec^2$ per degree)

$\ell_A = 29.2$ ft (assumed; no data available)

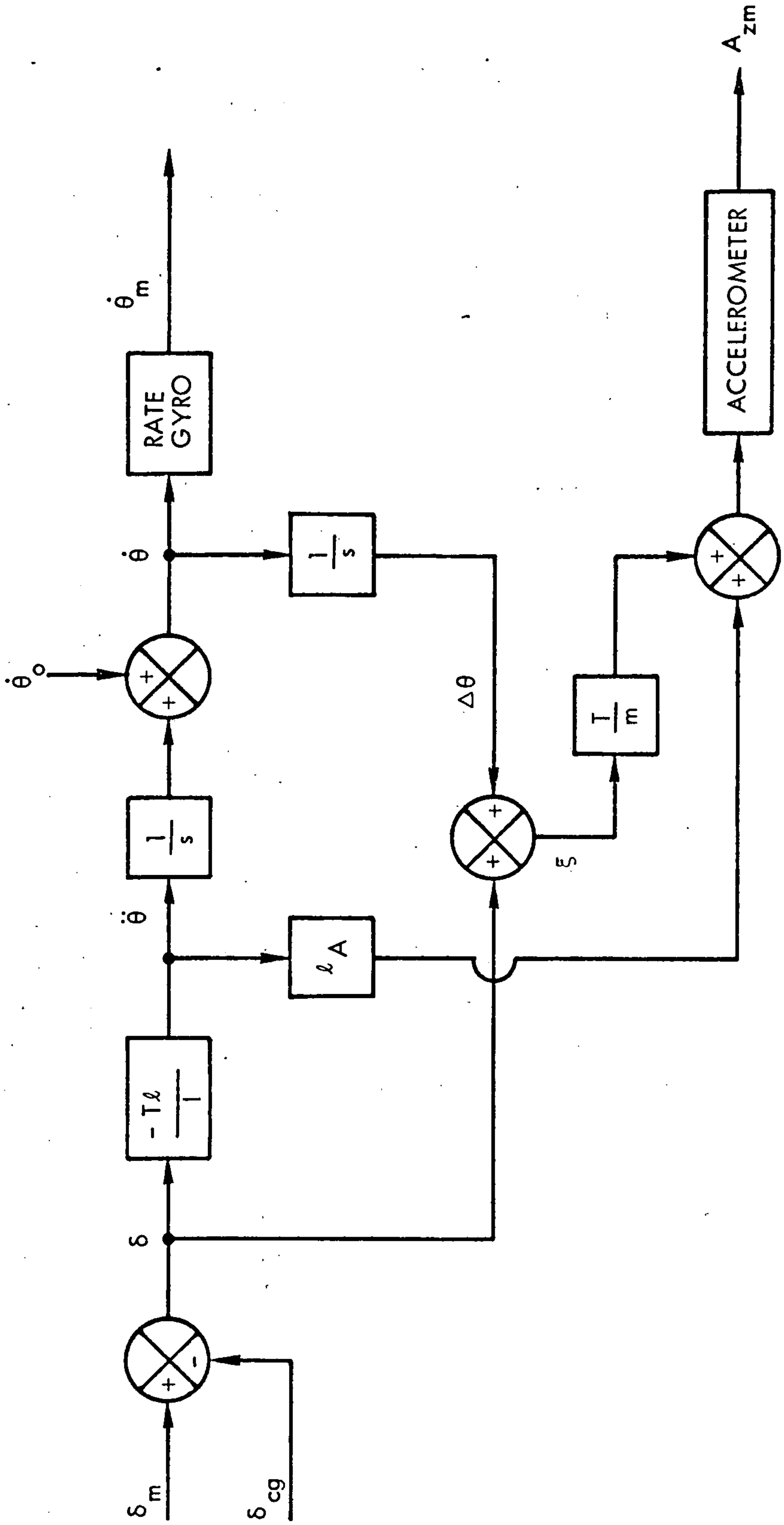


FIGURE 3

VEHICLE DYNAMICS

4. ACTUATOR DYNAMICS

The nonlinearities of the actuator dynamics must be studied carefully before proceeding with the software design. A block diagram of the actuator, based on the data given in the preceeding section of this report, is shown in Figure 4. Although the actuator can be represented simply at low frequencies, it actually represents an aggregate of electronic, hydraulic and mechanical devices which cause a large number of lags and nonlinearities at higher frequencies. The limits will be assumed to be representative of all such lags and nonlinearities at the lower frequencies.

The limits given in Reference 1 for the Shuttle were compared with those for the Apollo SPS. The deflection and rate limits are comparable, but the acceleration limit is only a small fraction of the SPS equivalent. The specified minimum output torque of the SPS actuator was sufficient to guarantee an angular acceleraticn of 3.5 radians/sec^2, which is about 6 times that indicated for the Shuttle.

Reference 1 gives the length of the engine as only 6 ft (75") which would indicate a moment of inertia of only about 90 slug-ft^2 based on SPS data. A moment arm of 0.6 ft should be attainable. If these assumptions are reasonable, the lateral force required to achieve the indicated acceleration limit is in the 100-pound neighborhood. Such a small force would appear to be in the vibration noise level of a 10,000 lb. engine mounted on gimbals. The tail-wags-dog effect is not serious.

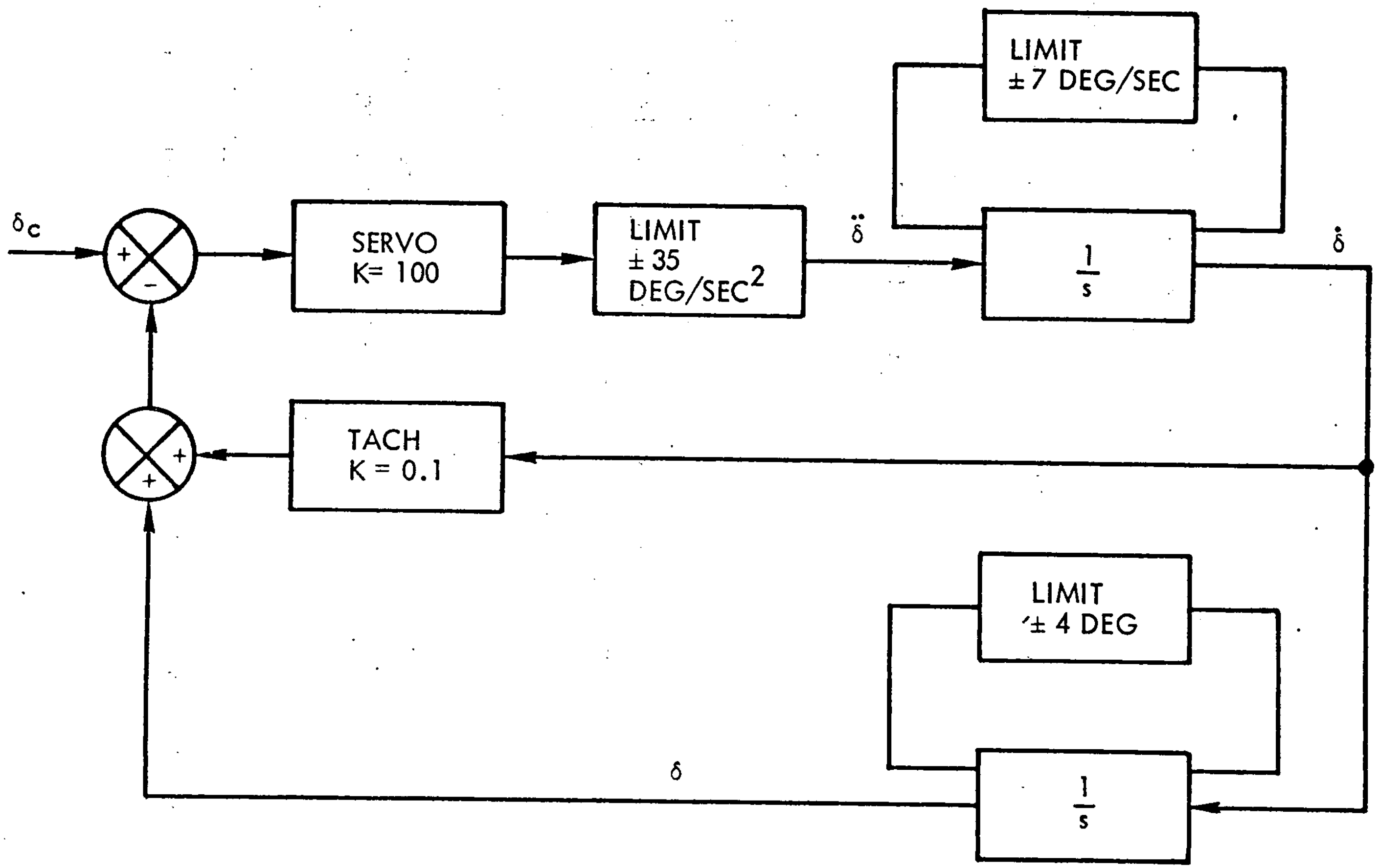
The servo-amp/torquer gain, K, and the tachometer feedback gain, K_R, are not specified. Without limiting, the linear transfer function for the actuator in Figure 4 is as follows:

$$\frac{\delta}{\delta_C} = \frac{K}{s^2 + K_R K s + K}$$

9.7.4.2 TVC, Gimballed (continued)

FIGURE 4

ACTUATOR MODEL



The linear natural frequency is

$$\omega_N = \sqrt{K}$$

but this frequency is the bandpass for input amplitudes of

$$A < 35/K$$

For larger amplitudes, limiting will reduce the bandpass of the actuator.

The gain proposed by NR can be taken as an example; for K = 625:

$$\omega_N = 25 \text{ rad/sec}$$

$$A = 0.056 \text{ deg.}$$

If we assume K is infinite, or that the input amplitude is infinite, the maximum output amplitude achievable is only 0.070 deg at 25 rad/sec. These amplitudes are so small that the actuator will be operating as a bang-bang system at nearly all input amplitudes. The signal-to-noise ratio of the tach and LVDT feedbacks will probably be too low at these amplitudes and frequencies to justify the wide bandpass and high gain.

In order to evaluate the actuator performance for all frequencies and amplitudes, it is necessary to determine the frequency response characteristics of all three limits. The first step is to determine the maximum output as a function of frequency for each of the limits separately. The maximum output due to the deflection limit is, of course, given as

$$\delta_{MAX} = 4° \qquad (\delta_{LIMIT})$$

The maximum amplitude of output deflection assuming a bang-bang rate limit of 7 deg/sec is

$$\delta_{MAX} = 7t$$

where t is one-fourth the period, or

$$t = \frac{\pi}{2\omega}$$

$$\delta_{MAX} = 11°/\omega \qquad (\dot{\delta}_{LIMIT})$$

Finally, the maximum output for a bang-bang acceleration limit of 35 deg/sec^2 is

$$\delta_{MAX} = \frac{1}{2}(35)\, t^2$$

$$\delta_{MAX} = 43.2°/\omega^2 \qquad (\ddot{\delta}_{LIMIT})$$

By this process, we are able to convert all three limits to deflection limits as a function of frequency. These output amplitude limits are plotted in Figure 5.

Figure 5 shows that an input of 4 degrees will yield an output of 4 degrees out to about 2.5 rad/sec. Additional computations show that some rate limiting will occur between 2.39 and 7.85 rad/sec; however, the acceleration limit begins to dominate beyond 4 rad/sec. At 10 rad/sec, unity gain cannot be achieved for amplitudes greater than 0.41 degree; this is unfortunate because large quantities of energy can be dissipated in the servo/torquer as a result of bending frequency (7.8 to 9.4 rad/sec) inputs from the control system if limiting occurs. In addition, phase shift will be added to the rate loop at the bending frequencies if this type of limit is encountered.

The natural frequency of the actuator was chosen as 10 rad/sec for two reasons; first, the rate loop must be designed to cope with actuator nonlinearities starting at 2.5 rad/sec, and second, the output amplitude is only 10 percent of maximum deflection at that frequency. The damping ratio was arbitrarily chosen as 0.5, which yields a linear transfer function and gains as follows:

$$\frac{\delta}{\delta_C} = \frac{100}{s^2 + 10s + 100}$$

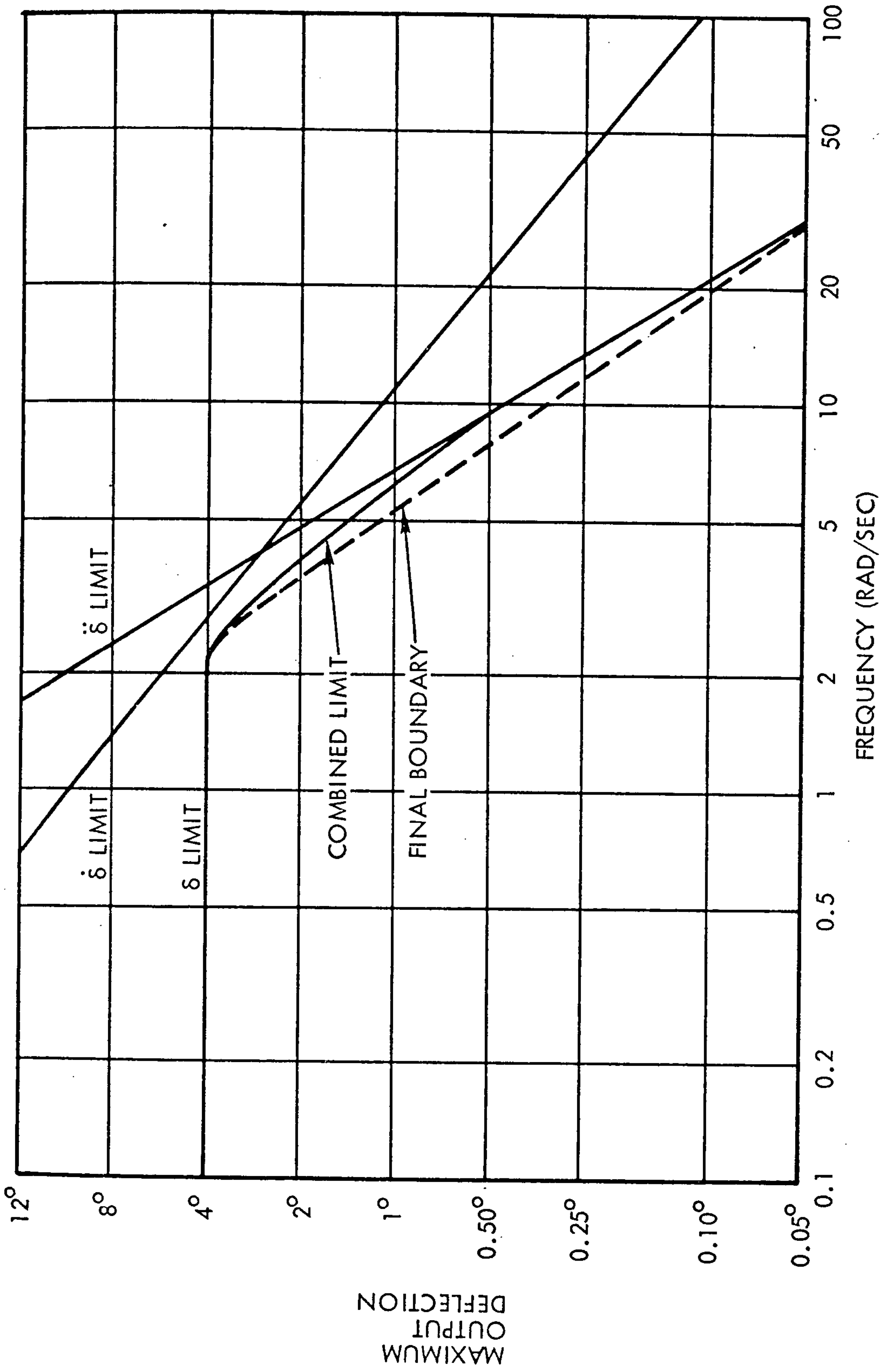
$$K = 100/\text{sec}^2$$

$$K_R = 0.1 \text{ sec}$$

The effects of the finite servo/torquer gain and multiple limits were added to the rigid limit boundaries in Figure 5 to obtain the curve labeled "Final Boundary".

FIGURE 5

ACTUATOR LIMITS VS. FREQUENCY



For a given input amplitude, the gain will be unity (zero db) as long as the output level remains below the "Final Boundary" curve in Figure 5. By drawing zero db gain lines for selected input amplitudes (horizontal lines in Figure 5), a family of gain curves is obtained as shown in Figure 6.

The phase curves in Figure 6 were obtained by studying the input and output wave forms during the limiting process. Standard curves can be used for phase shift during linear operation. When severe acceleration limiting occurs at high frequencies (above 7.85 rad/sec), and large amplitudes (one-half deg/sec), the phase shift is near 180 degrees due to the bang-bang effect. If rate limiting occurs, an equation for phase shift is required as a function of the amplitude ratio and the input frequency. This equation will be derived with the aid of Figure 7.

In Figure 7, the output deflection is plotted against output phase, and the input is defined by a variable phase. The feedback and output differ by a fixed bias during rate limiting, and this curve is also plotted. The rate limiting will end when the input and feedback are equal as shown by the circled point in Figure 7; the acceleration changes sign and nulls the rate in 0.2 sec. (Bang-bang acceleration is assumed; no error is introduced by this assumption for reasons given later).

The equations applying at the acceleration switch point circled in Figure 7 are as follows:

$$\delta = \delta_{PK} - (35/2)(0.2)^2$$

$$\dot{\delta} = 7 \text{ deg/sec}$$

$$\delta + 0.1\dot{\delta} = \delta_{PK}$$

$$A \text{ Sin } (\phi_{in}) = \delta_{PK}$$

$$\phi_{in} = \pi - \arcsin (\delta_{PK}/A)$$

$$\phi_{out} = \frac{\pi}{2} - 0.2\omega$$

$$\phi = \phi_{out} - \phi_{in}$$

$$\phi = \arcsin (\delta_{PK}/A) - \frac{\pi}{2} - \frac{\omega}{5}$$

$$(2.4 < \omega < 7.8 \text{ rad/sec})$$

FIGURE 6
ACTUATOR FREQUENCY RESPONSE

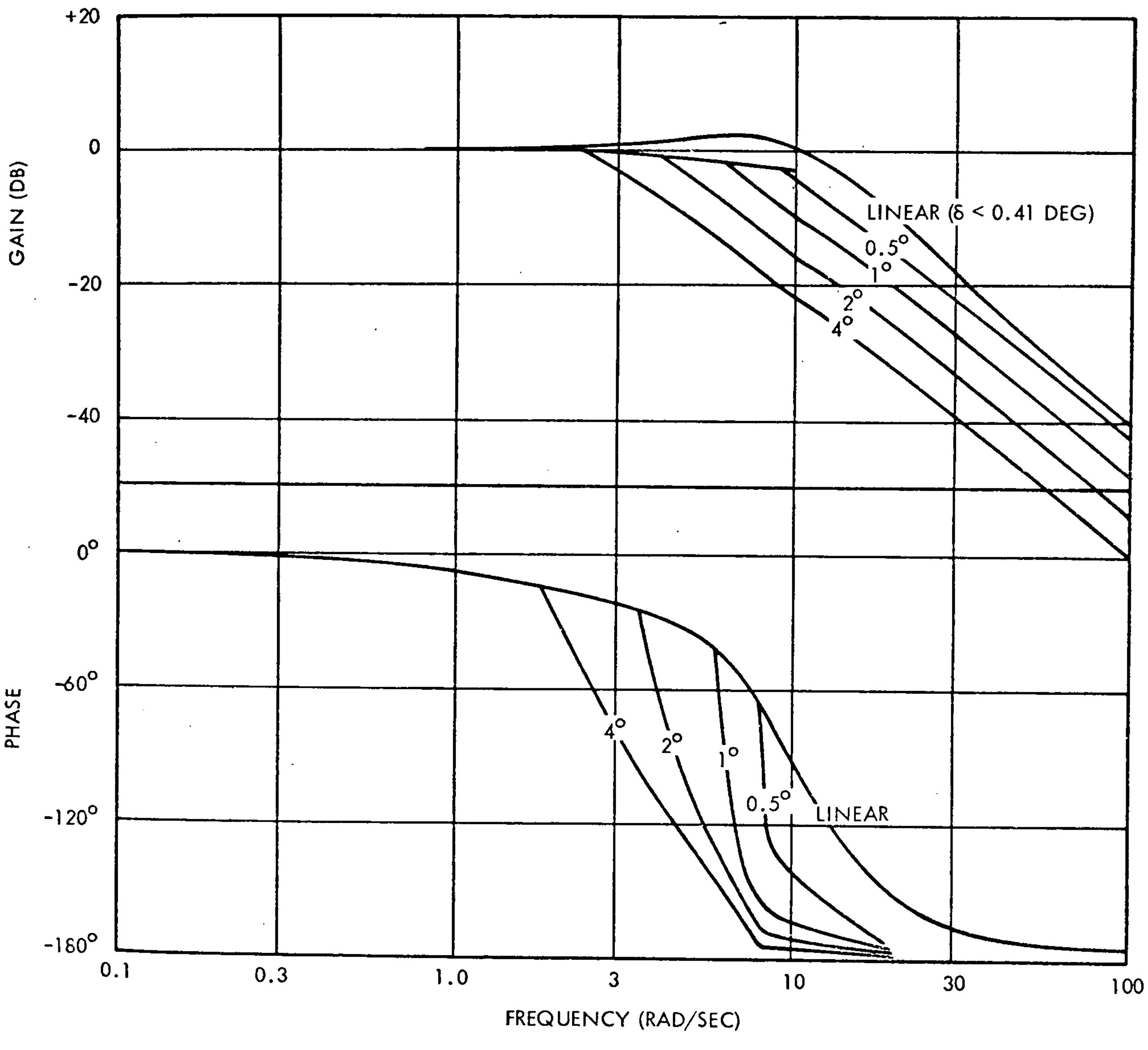
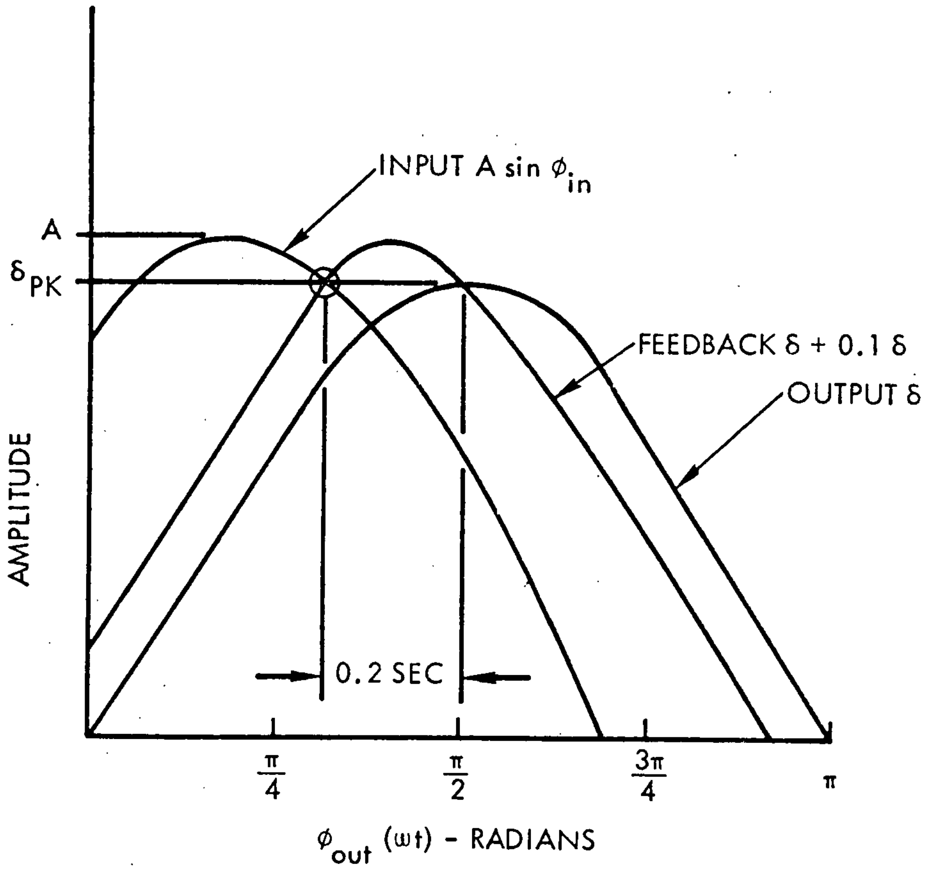


FIGURE 7

PHASE LAG IN ACTUATOR



The last equation yields the phase shift plotted in Figure 6 based on the gain ratios of the same figure. (Note that the arcsin term contains the gain ratio, δ_{PK}/A, which has been modified to reflect the attenuation and lag due to employing a finite gain; hence, the bang-bang acceleration switching assumption is correct).

In this design study, the selected actuator was not well suited to vehicle stabilization by engine gimballing alone. A minimum actuator acceleration limit of 1.5 radians/sec^2 would alleviate many of the design problems encountered in this study. The phase-shift discontinuities at the bending frequencies that result from the baseline actuator design may prevent phase stabilization of the control system if a bending instability problem develops in the later stages of the program development. Notch filters may be required in the control system to prevent large bending amplitudes from reaching the actuator input.

5. RATE LOOP DESIGN

The controllability of the vehicle for rate loop design is bounded by the actuator limits multiplied by the vehicle dynamics gain $T\ell/I$. The deflection limit must be reduced by ±0.23 degree because of variations in the location of the vehicle cg; other errors are known to exist between the intended null alignment and the physical trim deflection, but no data is available on these biases. The total null bias is estimated at 0.4 degree, which leaves 3.6 degrees for control deflection; multiplying by $T\ell/I$, we obtain the vehicle rate control capability as follows:

$$\ddot{\theta} = 3.6(0.06) = 0.216 \text{ deg/sec}^2$$

This controllability function defines the step time response and ramp-input response of the optimum control system designed for this vehicle configuration. For example, the time response to a step input of one degree/second might be specified as 5 seconds; the rate errors resulting from ramp inputs of up to 0.2 deg/sec^2 might be specified as 0.2 deg/sec (assuming a maximum actuator step-response of one second).

The frequency response of the rate loop is probably more important than the controllability available. The primary reason for having a rate loop is to damp thrust vector oscillations; hence, performance boundaries must be determined for the rate loop as a function of frequency. This analysis can be accomplished by the method used for determining the performance boundaries for the actuator in the preceding section.

The limits on vehicle body rates can be obtained by converting each of the actuator limits to limits on the first three derivatives of body rate and then integrating one additional time to obtain body rates. That is, apply the transfer function

$$\dot{\theta}/\delta = -0.06/s \quad \text{per sec}$$

to the maximum actuator response. When the actuator limits are multiplied by the gain, the limits become:

$$\ddot{\Theta}_L = \pm 0.24 \quad \text{deg/sec}^2$$

$$\dddot{\Theta}_L = \pm 0.42 \quad \text{deg/sec}^3$$

$$\ddddot{\Theta}_L = \pm 2.10 \quad \text{deg/sec}^4$$

The maximum output vehicle rates are obtained by integration over the quarter-cycle as follows:

$$\dot{\Theta}_{MAX} = (0.24)(\pi/2\omega) \qquad [\delta \text{ limit}]$$

$$\dot{\Theta}_{MAX} = (1/2)(0.42)(\pi/2\omega)^2 \qquad [\dot{\delta} \text{ limit}]$$

$$\dot{\Theta}_{MAX} = (1/6)(2.1)(\pi/2\omega)^3 \qquad [\ddot{\delta} \text{ limit}]$$

These individual rate boundaries are plotted as solid lines in Figure 8. The boundary representing the combination of limits is labeled "MAX. VEHICLE RATES". The curve labeled "0.32 DEG/SEC" was selected as the linear design goal because of the rapid loss of bandpass beyond one radian/sec.

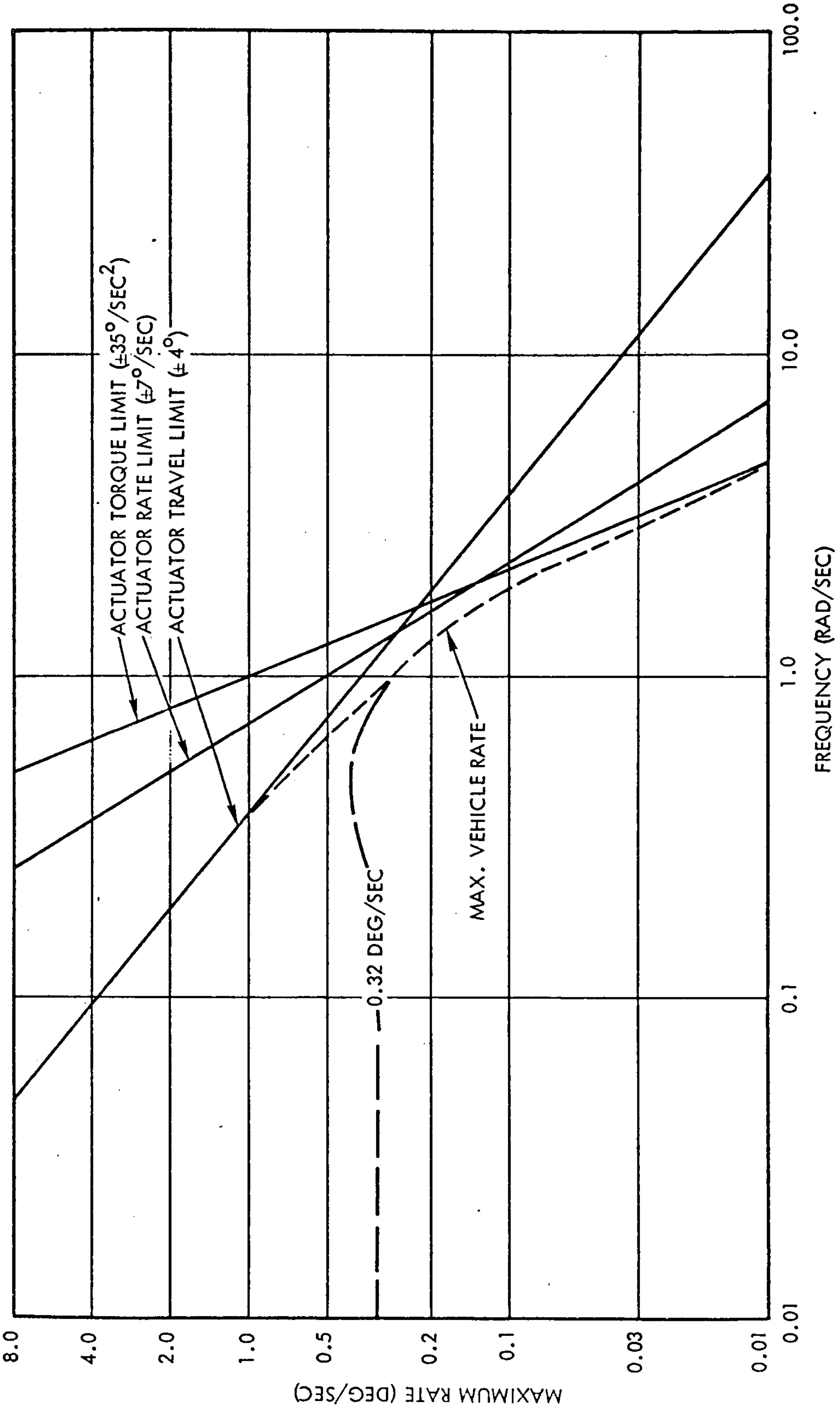
The maximum control response to the bending frequency of 8 radians/sec is about 0.0025 deg/sec peak rate. Assuming second-order attenuation above one rad/sec, the input from the rate gyro would have to be 0.16 deg/sec to produce saturation at the bending frequency. The damping provided by the structure would have to be about 0.015 to prevent the possibility of control instability; this requirement assumes that the control feedback is shifted 180°. The additional lag provided by the linear actuator at 10 rad/sec should eliminate most of the effects of linear control. Regardless of whether the control system stabilizes or destabilizes, the effect of control on bending will not be spectacular.

One more requirement must be recognized before completing the rate loop design. The rate error should ideally be used to generate a vehicle angular acceleration command proportional to the force required; for engine gimballing we must convert the angular acceleration command to an engine deflection command as follows:

$$\delta_C = \delta_{TRIM} - K\ddot{\Theta}_C$$

FIGURE 8

MAXIMUM VEHICLE RATES VS. FREQUENCY



The bias term δ_{TRIM} can only be estimated before the burn; however, if $\ddot{\Theta}_C$ is biased in one direction for a long period of time, it is safe to transfer the bias to δ_{TRIM} regardless of the true source of the bias. The transfer is obtained as follows:

$$\delta_{TRIM} = \delta_{EST} - K' \int K\ddot{\Theta}_C \, dt$$

$$\delta_C = \delta_{EST} - \left(\frac{K'}{s} + 1 \right) K\ddot{\Theta}_C$$

The input δ_{EST} is important because errors in the estimated engine trim will result in control transients, the only source of velocity errors computed during the burn. The transfer function sought for trim estimation is:

$$\delta_C/\ddot{\Theta}_C = -K \left(\frac{s + K'}{s} \right)$$

The closed loop linear bandpass of the rate loop was determined from the limit plots of Figure 8. In addition, several open loop requirements and characteristics have been determined; they are:

1) The vehicle dynamics
2) The linear actuator dynamics
3) The desired phase lag at the bending frequency
4) The form of the trim estimation transfer function
5) The attenuation of the bending frequency.

The only characteristics of the open loop that can be varied are the sample frequency, the lead term, K', in the trim estimation, the loop gain, and the form of the filter.

The sampling frequency was arbitrarily set at 10 samples per second. Lower frequencies may be possible, but the increased D/A filtering that would be required at the actuator interface is not an attractive prospect. This frequency is also compatible with reaction-jet control, which will be considered in conjunction with engine gimballing in the Final Design Phase of this study.

The form of the filter was determined by assuming a second-order lag with a first-order lead term; that is

$$\frac{s + A}{s^2 + Bs + C}$$

This form provides the maximum flexibility in controlling the phase/gain relationships by changing the damping and break frequencies. This process is mostly "trial and error". The result was an overdamped filter that could easily be replaced with a first-order lag; underdamped filters produced a closed-loop that did not fit the limit curve in Figure 8.

The selection of a first-order lag filter left only three remaining parameters to be determined; the gain, the filter time constant, and the trim integrator break frequency, K'. The signal compensation in the rate loop takes the form:

$$\frac{\delta_C}{\varepsilon_{\dot{\Theta}}} = \left(\frac{s + K'}{s}\right)\left(\frac{-K_F}{\tau s + 1}\right)$$

The vehicle and actuator dynamics feedback is:

$$\frac{\dot{\Theta}}{\delta_C} = \left(\frac{-.06}{s}\right)\left(\frac{100}{s^2 + 10s + 100}\right)$$

The product of these two transfer functions constitutes the complete open rate loop transfer function; however, the sampling delay must be added to the phase lag ($\phi = 57.3\ \omega/20$). The three remaining parameters can be determined in various ways depending on the effect desired.

The selection of these last three parameters for the basic preliminary design was based on the following objectives:

1) Maximize the high-frequency gain to the structural filtering requirement of 30 db at 8 rad/sec.

2) Maintain a phase lag of 270 degrees at the bending frequency.

3) Separate the trim bias frequency (K') from the filter frequency ($1/\tau$) by a factor of 10.

Only 14 degrees phase lead was needed to bring the total phase lag down to the minimum of 270 degrees at the bending frequency; this determined that the lead-lag should be:

$$\frac{s + 0.2}{s + 2.0}$$

The gain K_F was then determined by plotting the amplitude-ratio curve and computing the gain required for the 30 db attenuation. The result is $K_F = 21.5$ sec.

The frequency response of the rate loop is shown in Figure 9 for both the open and closed loops. The rate loop compensation developed above appears at the right in the block diagram in Figure 10. The closed loop bandpass in Figure 9 is higher than the curve sketched in Figure 8; the faster rate loop is desired for the initial response to ignition transients but will increase the range of nonlinearity in the rate loop. The curve in Figure 8 can be approximated by reducing the gain K_F by a factor of two, if bandwidth is less important than linearity.

The limit of ±1 deg/sec on the steering rate commands was selected based on the time required to remove the command rate (about 5 seconds). This limit could probably be reduced by a factor of 10 in order to eliminate the possibility of actuator saturation due to steering. The maximum steering command under normal (non-fail) operation is expected to be about 0.25 deg/sec due to the low thrust acceleration. The next section will discuss the steering system further.

The roll-axis control is a simple rate-damper. Because no attitude control is required in roll, the rate loop bandpass can be reduced by lowering the gain to 5 sec compared to 21.5 sec for pitch/yaw. No rate filter is required with the lower gain. The requirement for roll damping stems from the probability that the two pitch actuators will have different trim biases. The roll torque is the difference between the pitch torques supplied by the two engines. The pitch control responds only to the sum of the pitch torques; hence, the roll rate loop is required to compute a differential command to the pitch actuators. A trim integrator is included in roll to allow the roll rates to decrease as the burn progresses.

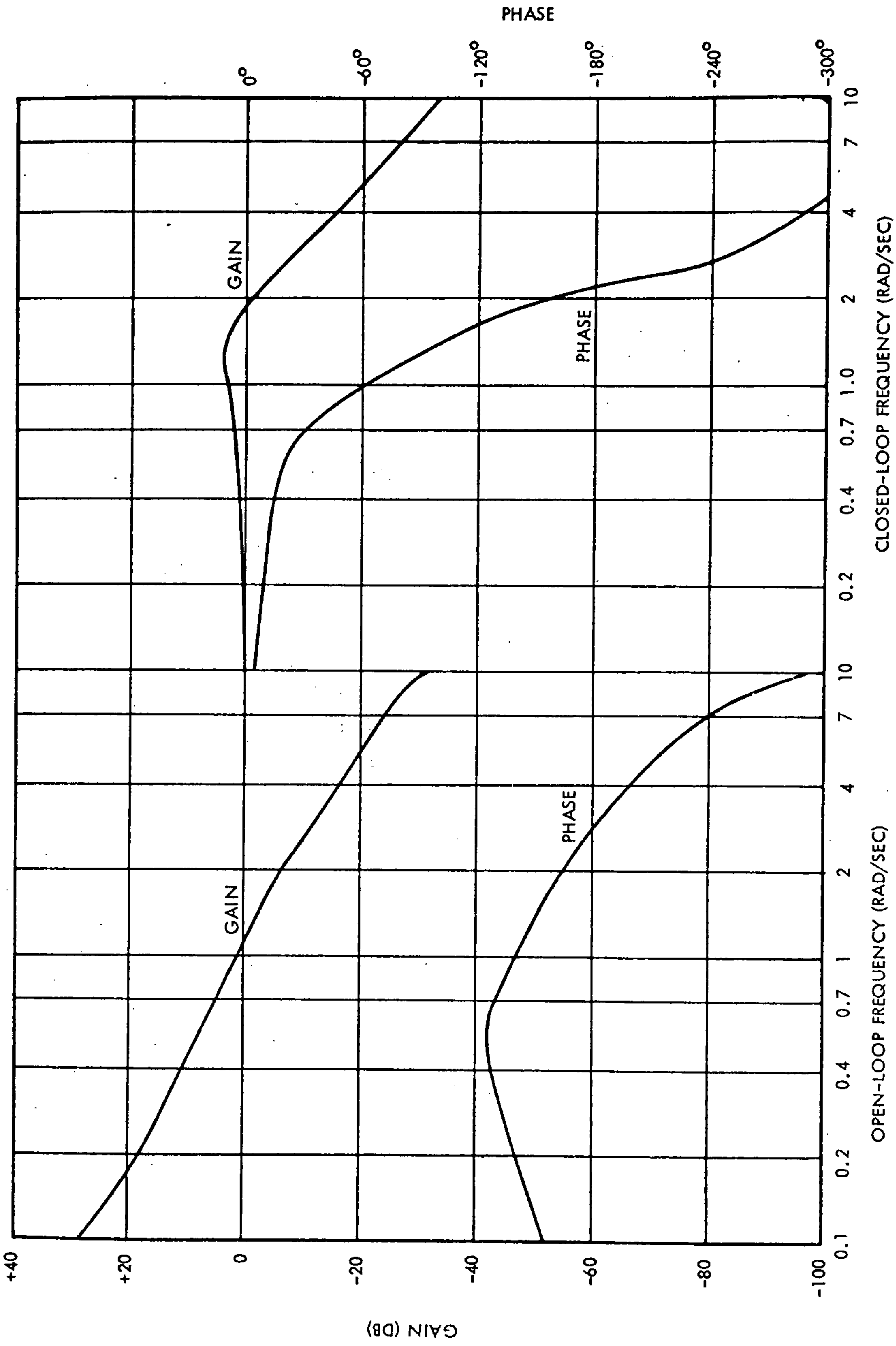


FIGURE 9
RATE LOOP FREQUENCY RESPONSE

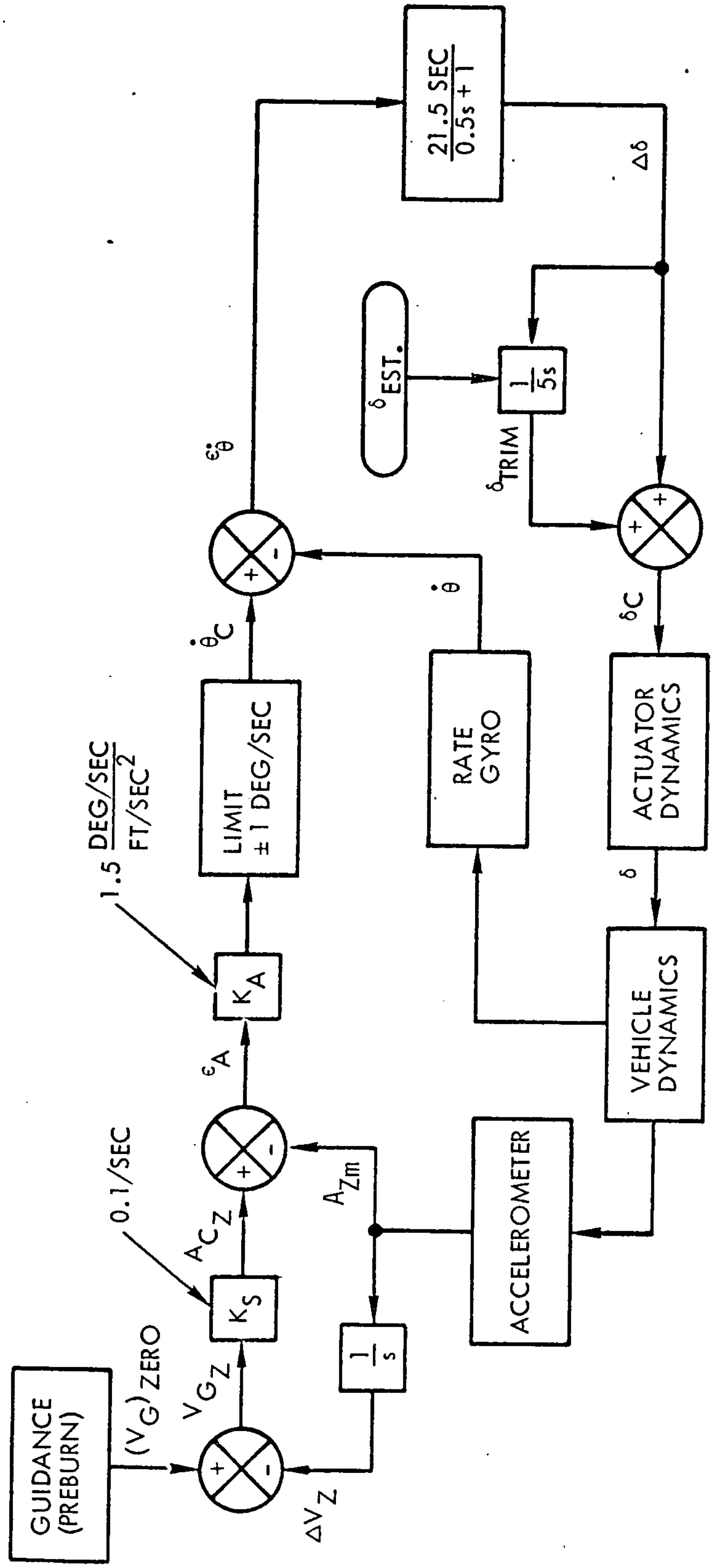


FIGURE 10
BLOCK DIAGRAM OF SOFTWARE DESIGN

6. STEERING LOOP DESIGN

The configuration of steering loop is shown on the left in Figure 10. For the purpose of this preliminary design, it was assumed that the accelerometers are mounted on a stable platform. The steering variables shown in Figure 10 are merely symbolic of the pitch (Z-axis) components of the steering variables that would be contained in the flight computer. The guidance input is a constant Delta-V command used as a directional reference for thrust vector control; $(V_g)_{ZERO}$ has no Z component, by definition, and is shown in Figure 10 only to indicate the summation point for the comparison of velocity vectors. Vector steering commands are resolved into body rates before limiting.

The steering loop can be represented as three transfer functions and a delay. The transfer function for the steering computation is:

$$\dot{\Theta}_C/A_{zm} = -K_A\left(\frac{s + K_S}{s}\right)$$

The transfer function for the vehicle dynamics was given previously as:

$$A_{zm}/\dot{\Theta} = \frac{-0.1207(s^2 - 0.314)}{s}$$

The closed rate loop is represented by Figure 9 and could be equated to the following transfer function:

$$\dot{\Theta}/\dot{\Theta}_C = 2.18/(s^2 + 1.48s + 2.18)$$

A sample period of two seconds was selected; the maximum period for the selected vehicle is about 5 seconds. The sampling lag is one second ($\phi = 57.3\omega$).

The most troublesome feature of the steering loop design is the location of the accelerometers relative to the cg. The vehicle dynamics shown in Figure 3 yield the general transfer function for acceleration measurement (in ft/sec per radian), as follows:

$$A_{zm}/\delta = T/m - (T\ell/I)(\ell_A + T/m\ s^2) .$$

Using the nominal values from Table 2 for T/m and $T\ell/I$, we obtain a function of the accelerometer moment arm:

$$A_{zm}/\delta = (-.06/s^2)\ [(\ell_A - 36.2)s^2 + 2.17]$$

The coefficient $(\ell_A - 36.2)$ arises from having both the sensor and center of thrust application at stations different from the cg.

If the accelerometers are located more than 36.2 feet forward of the cg, the steering will respond too much to vehicle angular accelerations. If the accelerometers are located too near the cg location, the steering will respond too much to gimbal angle transients which are opposite in sign to the vehicle angular acceleration. Either way, these signals change so rapidly that the next two-second steering command should not be affected by them. The ideal location of the accelerometers is 36.2 feet from the cg. The location assumed is seven feet aft of the ideal station to allow for cg changes and practical limitation on placement of the sensor in the vehicle. Less favorable locations will require compensation in the steering loop.

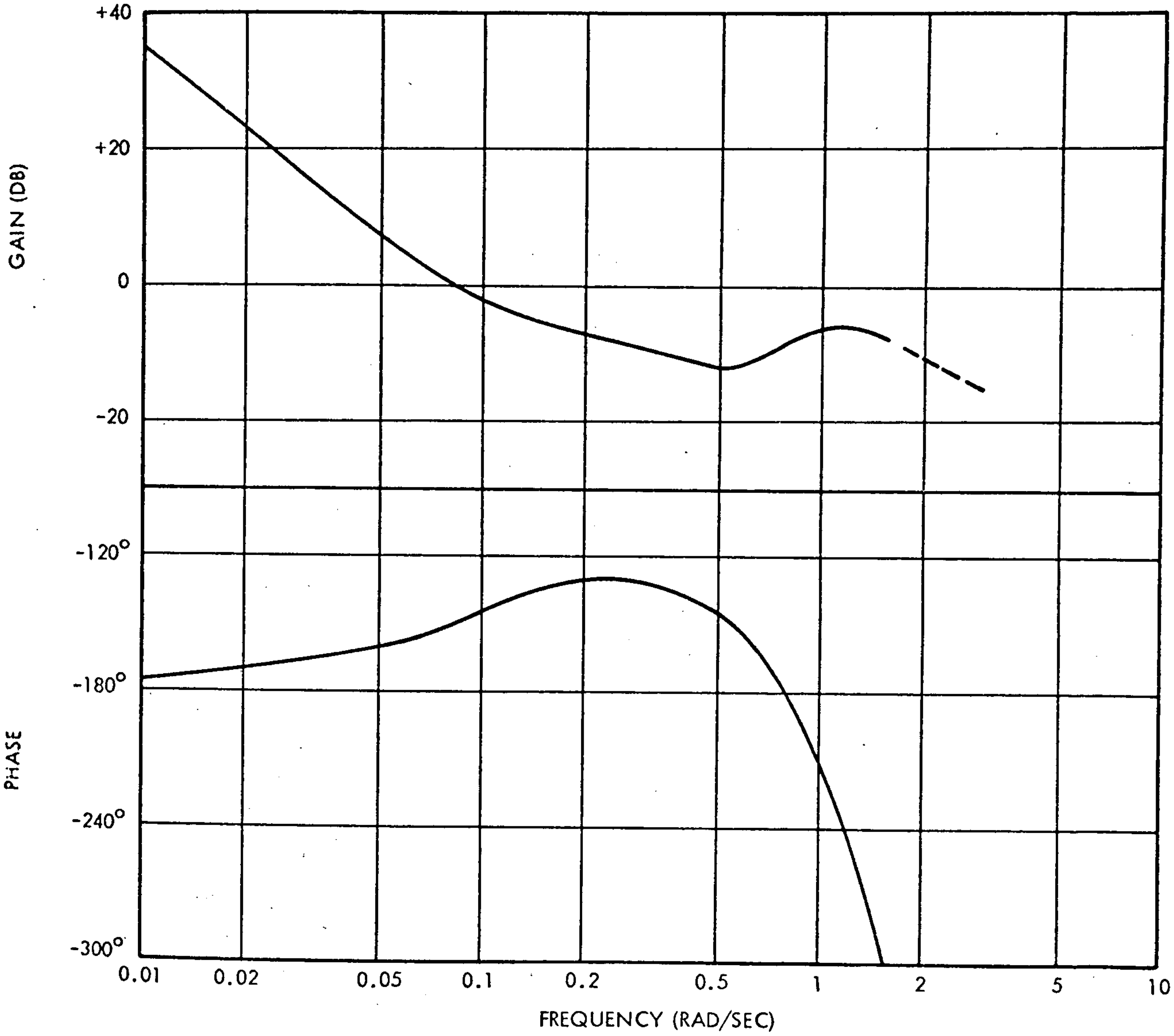
The acceleration gain, K_A, is determined by the rate loop resonant peak at 1.2 rad/sec, and the velocity error gain, K_S, determines the bandpass of the closed steering loop. The open steering loop frequency response is shown in Figure 11. Because the 180° phase occurs near the rate loop peak, the open loop gain should be less than -6 db in that region; otherwise, the closed steering loop will contain a resonance at the higher frequency. The peaking of the gain curve is chiefly due to the measurement zero at 0.56 rad/sec, which result from the non-optimum location of the accelerometers. The gain required is 1.5 deg/sec per ft/sec^2 (0.84 rad/sec per g).

The selection of the steering gain, K_S, was made on the basis that the closed steering loop requires a 6 db resonance in order to achieve the faster response without causing overshoot in the velocity error. The

FIGURE 11

OPEN STEERING LOOP FREQUENCY RESPONSE

(Open at accelerometer output or at limiter in Figure 10)



value selected for K_S was 1/(10 seconds) which produces the closed-loop frequency response shown in Figure 12.

The large resonant peak at the low frequency in Figure 12 would be interpreted as a massive overshoot in response to step guidance inputs. This is true. As mentioned previously, this design assumes External Delta-V guidance which provides for no cross-axis velocity inputs. The steering loop is designed to respond to ramp velocity errors which will develop during attitude stabilization. The amplitude peak at one rad/sec is a disturbance resulting from the intrusion of angular accelerations into the steering measurement.

The chief input is expected to result from residual alignment errors from the ACPS at the start of the burn. The ideal relationship between the ACPS phase plane errors would be:

$$E = -(2 \text{ seconds}) \dot{E}$$

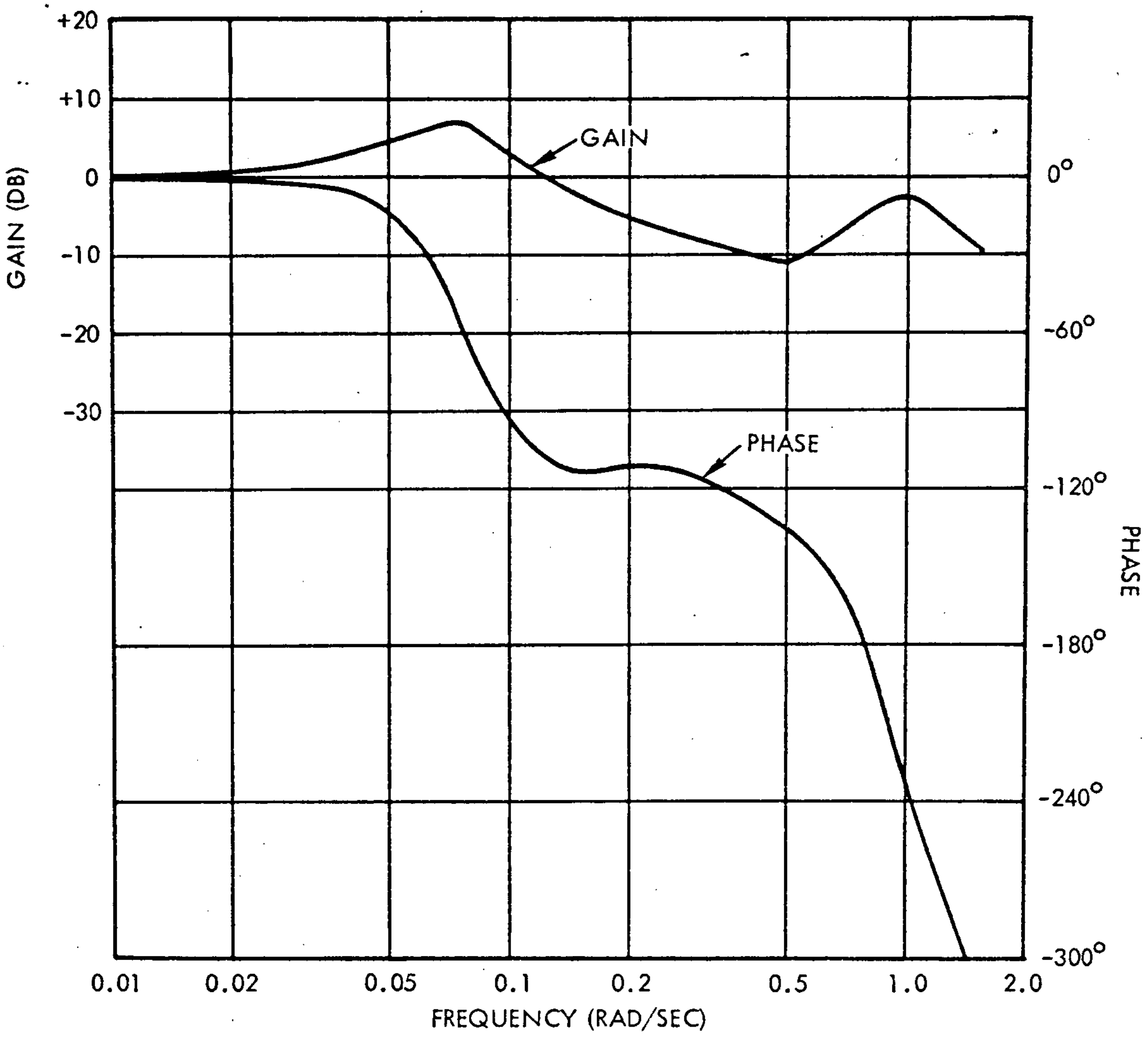
This relationship would align the vehicle properly during the first two-second steering cycle. The next steering command should contain a small pointing error, but the remaining commands will be mostly a result of velocity errors during the first three seconds. The first steering command is input at ignition as:

$$\dot{\Theta}_C = E_p/2 \qquad (\text{Limit} = 1 \text{ deg/sec})$$

ACPS attitude errors greater than 0.2 degree will produce actuator limiting, but such large errors are not anccticipated.

Engine trim estimation errors will also result in pointing errors, but these errors will be smaller because of the low angular acceleration resulting from the low thrust-to-inertia ratio. Thrust vector errors due to a one degree initial trim error would be about 0.02 degree. Peak velocity errors from both sources is expected to be about 0.04 ft/sec. This level of measurement will require more accurate accelerometers than used on Apollo because of the low thrust acceleration. The effect of thrust on sensor requirements is the topic of the following section.

BLOCK 12
CLOSED STEERING LOOP FREQUENCY RESPONSE
(Unity Feedback)



7. SENSOR REQUIREMENTS

Sensor characteristics were not considered as inputs to the preliminary design. This design assumed that the accelerometer and rate gyro thresholds would be as low as required. The burn accuracy obtainable with the preliminary design for the vehicle described in Section 3 of this report is about 0.01 ft/sec. The accelerometer threshold required to obtain this burn accuracy is about 10^{-4} g (0.003 ft/sec^2) or a two-second velocity increment of 0.006 ft/sec. The Apollo LM accelerometers generate torque pulses at the 10^{-4} g level, but the output velocity increments are too large (0.0328 ft/sec). Future thrust vector control studies should consider the effects of larger accelerometer thresholds than required by the preliminary design.

Guidance studies should be performed to determine the portion of the velocity error budget that is allocated to thrust vector control. This study reveals that the precision of the accelerometer measurements will probably be more significant than the control scheme error for the low-thrust vehicle. As a result, preliminary estimates of thrust vector control errors can be directly related to the accelerometer characteristics.

The rate gyro requirements are also a function of required burn accuracy as well as control factors. The preliminary design would require a rate gyro threshold of about 0.005 deg/sec and a linear range of ±1.5 deg/sec. If bending stability can be ignored due to low thrust or the slow actuator response, the rate gyro threshold can be raised.

8. SOFTWARE EQUATIONS

The equations appearing in this section are intended for implementation on the Space Shuttle Functional Simulator. These equations are equivalent to the functions indicated in Figure 10 and assume that the vehicle, actuator and sensor dynamics are those described in previous sections of this report. A top-level flow diagram is presented in Fig. 13.

INPUTS: The inputs are defined as follows:

$\underline{V}_{GO}$	Desired velocity increment in stable member coordinates.
$\underline{\Delta V}$	Accelerometer interface unit outputs in stable member coordinates accumulated over a two-second steering cycle.
$\underline{A}$	Measured acceleration.
[*SMB*]	Rate transformation matrix from stable-member to body-axis coordinates.
$\dot{\Theta}_m, \dot{\Psi}_m, \dot{\phi}_m$	Rate gyro interface unit outputs in body-axis coordinates (pitch, yaw, roll).
$\delta_{PEST}, \delta_{YEST}, \delta_{REST}$	Engine trim deflection estimates (pitch, yaw, roll); input about 5 seconds prior to ignition.
X	Vector cross-product multiplication.
Θ, Ψ	Attitude error computed by ACPS.
K	Constants as given below.

CONSTANTS: The constants below are equation values assuming accelerometer outputs in ft/sec, rate gyro outputs in deg/sec, and gimbal commands in degrees.

$K_S = 0.1$/sec

$K_A = 1.5$ deg-sec/ft

$K_{BP} = K_{BY} = 4.73$ sec

9.7.4.2 TVC, Gimballed (continued)

$K_{CP} = K_{CY} = 0.78$

$K_D = 0.02$

$K_E = 5.$ sec

$K_{RL} = 1.$ deg/sec

INITIALIZATION:

$\Delta\delta_P = \Delta\delta_y = 0$

$\delta_{PTRIM} = \delta_{PEST}$

$\delta_{YTRIM} = \delta_{YEST}$

$\delta_{RTRIM} = \delta_{REST}$

$\underline{U}_{VGO} = \underline{V}_{GO}/|V_{GO}|$

$\underline{V}_G = \underline{V}_{GO}$

ENGINE TRIM: (Entered 5 seconds before ignition)

$\delta_{PL} = \delta_{PTRIM} - \delta_{RTRIM}$

$\delta_{PR} = \delta_{PTRIM} + \delta_{RTRIM}$

$\delta_{YL} = \delta_{YTRIM}$

$\delta_{YR} = \delta_{YTRIM}$

(P = pitch, Y = yaw, L = left, R = right engine; these commands are outputs to the engine actuators - positive command produces negative pitch, yaw on vehicle).

IGNITION: (Entered after engine is on)

TSTEER = 2.0 seconds

$\dot{\Theta}_C = \Theta/2$

$\dot{\Psi}_C = \Psi/2$

(Limit $\dot{\Theta}_C$, $\dot{\Psi}_C$ to $\pm K_{RL}$)

Go to "RATE LOOP"

RATE LOOP: (Entered every 0.1 second)

$$
\begin{aligned}
E_P &= \dot{\Theta}_m - \dot{\Theta}_C \\
\Delta\delta_P &= K_{BP}E_P + K_{CP}\Delta\delta_P \\
\delta_{PTRIM} &= \delta_{PTRIM} + K_D\Delta\delta_P \\
\delta_{PC} &= \delta_{PTRIM} + \Delta\delta_P
\end{aligned}
$$

$$
\begin{aligned}
E_Y &= \dot{\Psi}_m - \dot{\Psi}_C \\
\Delta\delta_Y &= K_{BY}E_Y + K_{CY}\Delta\delta_Y \\
\delta_{YTRIM} &= \delta_{YTRIM} + K_D\Delta\delta_Y \\
\delta_{YC} &= \delta_{YTRIM} + \Delta\delta_R
\end{aligned}
$$

$$
\begin{aligned}
\Delta\delta_R &= K_E\dot{\phi}_m \\
\delta_{RTRIM} &= \delta_{RTRIM} + K_D\Delta\delta_R \\
\delta_{RC} &= \delta_{RTRIM} + \Delta\delta_R
\end{aligned}
$$

$$
\begin{aligned}
\delta_{PL} &= \delta_{PC} - \delta_{RC} \\
\delta_{PR} &= \delta_{PC} + \delta_{RC} \\
\delta_{YL} &= \delta_{YC} \\
\delta_{YR} &= \delta_{YC}
\end{aligned}
$$

TSTEER = TSTEER - 0.1 second

If TSTEER $\leq$ 0, perform "STEERING"

STEERING: (Entered every 2 seconds during burn)

TSTEER = 2.0 seconds

$$
\begin{aligned}
\underline{A}_S &= \underline{A} \times \underline{U}_{VGO} \\
\underline{A}_C &= K_S\,(\underline{U}_{VGO} \times \underline{V}_G) \\
\underline{E}_A &= \underline{A}_C - \underline{A}_S \\
\underline{RATE} &= K_A\,[*SMB*]\underline{E}_A
\end{aligned}
$$

STEERING: (Continued)

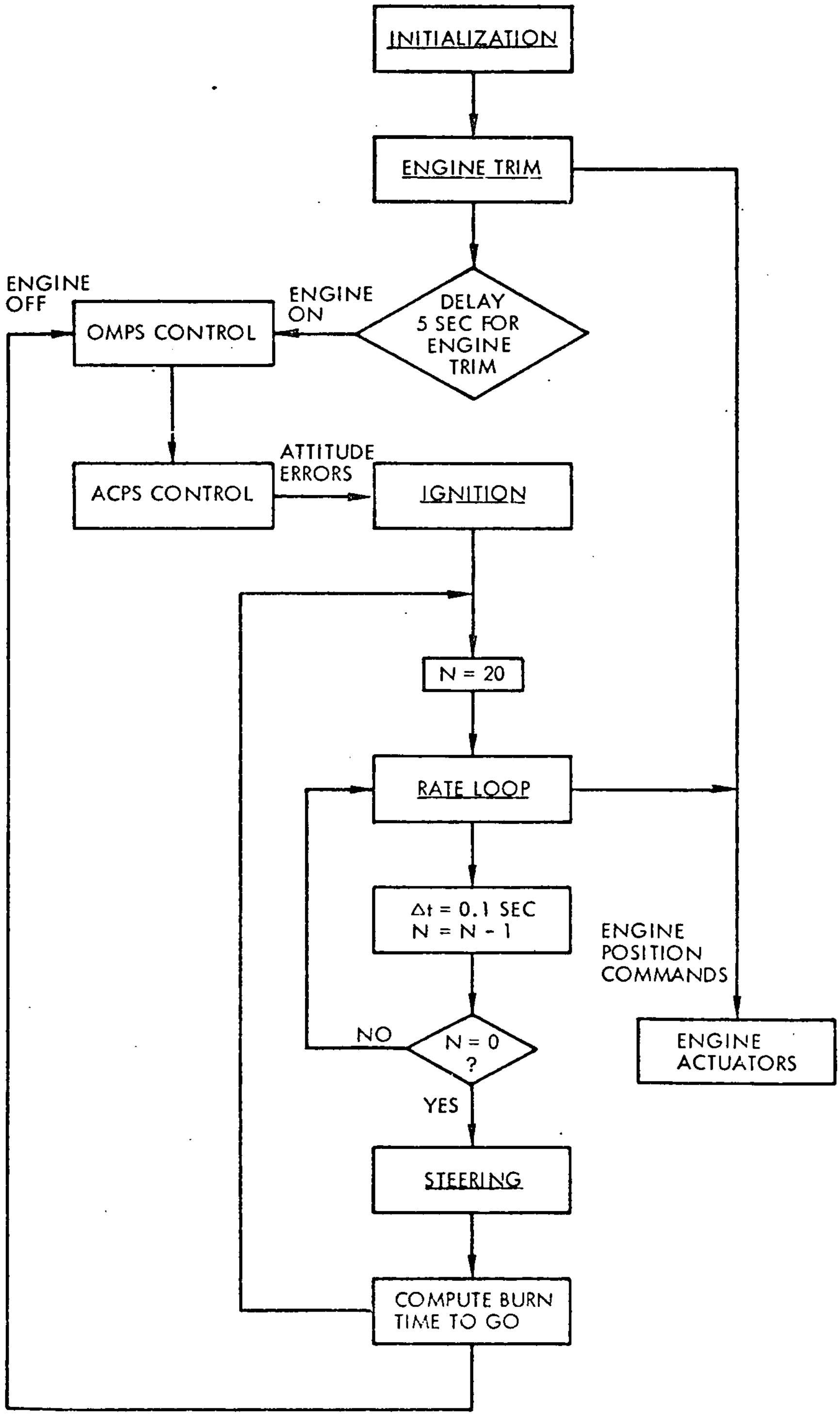
($\dot{\Theta}_C$ is the pitch component of RATE, $\dot{\Psi}_C$, the yaw component; the third component is not used).

Limit the magnitudes of $\dot{\Theta}_C$ and $\dot{\Psi}_C$ to $\pm K_{RL}$.

9.7.4.2 TVC, Gimballed (continued)

FIGURE 13

FLOW DIAGRAM



9. REVIEW

The preliminary design described in this report developes a basic concept of thrust vector control which has several special features; for example:

1) Integration of steering, stabilization and actuator loops.
2) No direct control of vehicle attitude.
3) Constant-gain steering for cross-axis velocity control.
4) High-gain engine trim estimater.
5) Design for maximum linearity using bandpass control.

The feasibility of performing thrust vectoring with the selected vehicle and actuator was demonstrated, although the damping of high-frequency oscillations was shown to be impractical with the low-performance actuator specified.

As a result of this design study, a minimum actuator acceleration of 1.5 radians/sec^2 is recommended to provide the gimbal control system with the capability to stabilize slosh and bending and to permit more rapid nulling of ignition transients. It is also recommended that the accelerometer threshold be considered in all future thrust vector control designs for low-thrust vehicle configurations.

The preliminary design was successful in identifying the major problems in designing a gimballed-engine TVC system for the shuttle. Some of these problems that require additional investigation are listed below:

1) Stabilization of slosh bending.
2) Quantization effects of sensors.
3) Accelerometer locations requiring compensation.
4) Minimum sampling frequencies.
5) Sensor biases.
6) Combinations of reaction-jet and gimballed-engine control.

10. REFERENCES

1) NASA/MSC EG2-71-132, "Design Data for Space Shuttle Orbiter (NR 161-C) ACPS and OMPS Control Equations", 14 June 1971.

2) TRW Memo 71:7153.5-87, "Work Plan for Shuttle On-orbit TVC Design Subtask", 14 June 1971.

9.8 RENDEZVOUS MISSION PHASE

The rendezvous mission phase begins at the completion of orbit insertion with the computation of the rendezvous plan. During this coasting period of time, the mission planning software will utilize the rendezvous targeting routines to insure that the insertion cut-off conditions (i.e., cut-off state vector) are within the rendezvous corridor defined by pre-mission and crew option inputs. When a satisfactory plan is established, the various tasks of the rendezvous will be assigned a preliminary schedule. The implementation of this plan will represent the remainder of the rendezvous mission phase.

The SW functions required in this mission phase are the following:

1. Estimate relative state of target vehicle based on external measurements (if available).
2. Estimate absolute states of both shuttle and target vehicle.
3. Target the rendezvous ΔV's required, their direction, and the time's of ignition.
4. Execute rendezvous maneuvers by commanding engine's on, providing attitude commands during the maneuvers, and commanding engines off.
5. Powered flight navigation.
6. Provide RCS engine commands to achieve commanded attitude during ΔV maneuvers and during coast periods (digital autopilot).
7. Provide data for failure analysis.
8. Provide data for crew display.

The guidance and navigation software during the burns will be the same as described in Orbital Powered Flight. The estimates of absolute states will be performed as described in Orbital Coast.

9.8.1 Targeting

SPACE SHUTTLE

GN&C SOFTWARE EQUATION SUBMITTAL

Software Equation Section: Rendezvous Targeting Submittal No. 21A

Function: Provide targeting solutions for rendezvous

Module No. OG3 Function No. 1,2,3,4, & 6 (MSC 03690 Rev.A)

Submitted by: W. H Tempelman Co. MIT No. 7 (Rev. 1)

Date: 21 October 1971

NASA Contract: J. Suddath Organization: GCD

Approved by Panel III K.J. Cox Date: 10/21/71

Summary Description: This submittal represents a single targeting program for providing targeting parameters to powered flight guidance for all of the series of maneuvers that make up a rendezvous sequence. The program can handle any given number of maneuvers. Many types of maneuver constraints are incorporated in the program such that virtually any sequence of rendezvous maneuvers can be accommodated. In addition, the Astronaut is provided a large, well-defined series of options by which he may modify the nominal sequence.

Shuttle Configuration: This software is essentially independent of the shuttle configuration.

Comments:

(Design Status)

(Verification Status)

Panel Comments:

Revision: A. Original submittal, now totally replaced, in February 1971.

9.8.1.1 Rendezvous Targeting

1. INTRODUCTION

The rendezvous of the Orbiter (primary vehicle) with a target vehicle (e.g. the Space Station) is accomplished by maneuvering the Orbiter into a trajectory that intercepts the target vehicle orbit at a time that results in the rendezvous of the two vehicles. The function of rendezvous targeting is to determine the targeting parameters for the powered flight guidance for each of the maneuvers made by the Orbiter during the rendezvous sequence.

In order to construct the multimaneuver rendezvous trajectory, sufficient constaints must be imposed to determine the desired trajectory. Constraints associated with the Orbiter mission will involve such considerations as fuel, lighting, navigation, communication, time, and altitude. The function of premission analysis is to convert these—which are generally qualitative constraints-into a set of secondary quantitative constraints that can be used by the onboard targeting program. By judicious selection of the secondary constraints, it should be possible to determine off-nominal trajectories that come close to satisfying the primary constraints.

The proposed onboard rendezvous targeting program consists primarily of a main program and a generalized multiple-option maneuver subroutine. The driving program automatically and sequentially calls the maneuver subroutine to construct the rendezvous configuration from a series of maneuver segments. The main program is capable of handling rendezvous sequences involving any given number of maneuvers. Enough different types of maneuver constraints are incorporated into the subroutine to provide the flexibility required to select the best set of secondary constraints during premission planning. In addition, the astronaut has a large, well defined list of maneuver options if he chooses to modify the selected nominal rendezvous scheme.

As the new approach represents, in essence, just one targeting program, there is considerable savings in computer-storage requirements compared to former approaches in which each maneuver used in the rendezvous scheme had a separate targeting program. The programming and verification processes of this unified approach will also result in implementation efficiencies.

1.1 Number of Independent Constraints Involved in a Rendezvous Sequence

During the Gemini and Apollo flights and in the design of the Skylab rendezvous scheme various numbers of maneuvers were utilized in the rendezvous sequence. The range went from two (Apollo 14 and 15) to six (Skylab).

The number of independent constraints (i.e., the number of explicitly satisfied constraints) in each rendezvous sequence must equal the number of degrees of freedom implicitly contained in the sequence. To establish this number, a rendezvous configuration can be constructed by imposing arbitrary constraints until the configuration is uniquely defined. For example, a four maneuver coplanar sequence is shown in Figure 1, followed by a coast to a terminal point. Using the constraints v_i (velocity magnitude), r_i and θ_i, it is easy to establish that the total number involved is 12, assuming the time of the first maneuver has been established. Removing one maneuver will reduce the number of degrees of freedom by three. Hence, the number of independent constraints necessary to uniquely determine the maneuver sequences are

Number of maneuvers in sequence	Number of independent constraints required
1	3
2	6
3	9
4	12
etc	

If the above rendezvous are not coplanar, one additional constraint has to be added to each sequence to allow for the out-of-plane component.

In some cases the number of primary constraints may be insufficient to uniquely determine a rendezvous trajectory for the desired number of maneuvers. One way of overcoming this deficiency in constraints is by introducing sufficient variables to complete the determination of the rendezvous trajectory and then determining values for these variables by minimizing the fuel used.

In order to take advantage of updated state vectors due to navigation or ground updates, the rendezvous targeting program is called prior to each maneuver to compute the upcoming maneuver. In general, each maneuver computation will involve a multimaneuver sequence as the nature of the targeting constraints do not allow the maneuvers to be independently computed. These sequences must have the same number of independent constraints as tabulated above.

1.2 The Construction of a Maneuver Segment

Each n-maneuver sequence can be divided into n-maneuver segments. Each segment involves, basically, the addition of a maneuver to the primary vehicle's velocity vector and an update of both vehicle's state vectors to the next maneuver point.

A maneuver segment can be generated in one of three ways:

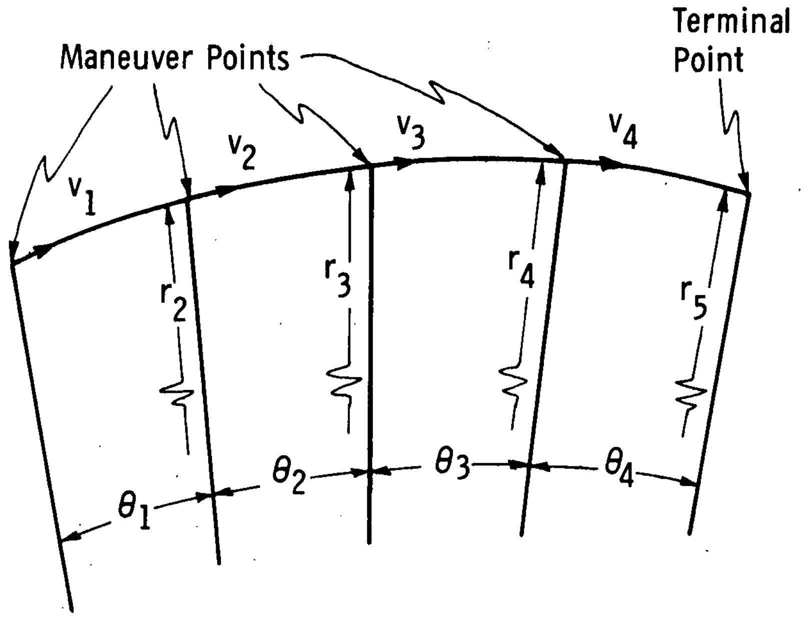


Figure 1. A Possible Set of Constraints Involved in a Four Maneuver Rendezvous Sequence

9.8.1.1 Rendezvous Targeting (continued)

Forward generation

A maneuver $\Delta \underline{v}$ is computed and added to the velocity vector in a specified direction. The state vector of the primary vehicle is then updated through a specified amount to arrive at the next maneuver position.

Target generation

The target vehicle is updated through a specified amount to establish a target vector for the maneuver. The maneuver is then computed by uniquely specifying the nature of the traverse between the primary vehicle's position and the target vector.

Integrated generation

In this case, the maneuver segment is computed as an integral part of a maneuver sequence involving more than one maneuver segment. The nature of the constraints are such that the maneuver sequence cannot be subdivided into uniquely defined maneuver segments. The maneuver segment will usually have one degree of freedom, which will generally be assumed to be the magnitude of the maneuver.

Each of the above methods is defined by the specification of three trajectory constraints (four in the case of noncoplanar traverses). Before introducing the maneuver constraints associated with the three ways of generating a maneuver segment, the constraints associated with updating a state vector will be listed. These constraints are specified with the update switch s_{update}

$$s_{\text{update}} = \begin{cases} 1 & \text{Update from time } t \text{ to time } t_F \\ 2 & \text{Update through time interval } \Delta t \\ 3 & \text{Update through } n \text{ revolutions} \\ 4 & \text{Update through } \theta \text{ radians} \\ 5 & \text{Update to be colinear with a specified position vector} \end{cases}$$

In the remainder of this section, the choice of a maneuver option is equivalent to the selection of a constraint.

1.2.1 Maneuver Options in Forward Generation of Maneuver Segment

The forward generation of a maneuver segment is accomplished in one of two ways. Either the maneuver magnitude is uniquely determined in terms of the state vector at the maneuver time or the maneuver is determined by an iterative search to satisfy a terminal constraint.

The maneuver magnitude Δv is either calculated or assumed depending on the maneuver switch s_{man}, and it is applied in a direction controlled by the direction switch s_{direct}. The options associated with the maneuver switch are:

$$s_{\text{man}} = \begin{cases} 1 & \Delta v \text{ is assumed specified} \\ 2 & \Delta v \text{ is computed based on a post maneuver velocity vector being "coelliptic" with the state vector of the target vehicle} \\ 3 & \Delta v \text{ is computed from the conic circular velocity constraint} \\ 4 & \Delta v \text{ is computed based on a Hohmann type transfer resulting in a } \Delta h \text{ change in altitude} \end{cases}$$

The options associated with the maneuver direction switch are:

$$s_{\text{direct}} = \begin{cases} -1 & \text{Apply } \Delta v \text{ is horizontal direction in plane of primary vehicle} \\ 1 & \text{Apply } \Delta v \text{ in horizontal direction parallel to orbital plane of the target vehicle} \\ -2 & \text{Apply } \Delta v \text{ along velocity vector in plane of primary vehicle} \\ 2 & \text{Apply } \Delta v \text{ along velocity vector parallel to orbital plane of the target vehicle} \end{cases}$$

The selection of the update switch s_{update_P} determines the update of the primary vehicle's trajectory following the maneuver to the position of the next maneuver. A terminal constraint can be imposed at this point by setting the terminal switch s_{term}:

$$s_{term} = \begin{cases} 1 & \text{Terminal constraint is a height constraint} \\ -1 & \text{Terminal constraint is a phasing constraint} \end{cases}$$

Following the computation of the height/phasing error, the maneuver magnitude is varied in an iterative search to satisfy the height/phasing constraint.

1.2.2 Maneuver Options in Target Generation of Maneuver Segment

The target generation of a maneuver segment starts with the selection of the update switch for the target vehicle. If this switch equals four, θ will be augmented by the central angle between the primary and target vehicles before being used. The nature of the traverse between the primary vehicle's initial state vector and the updated position of the target vehicle is controlled by the maneuver switch s_{man}:

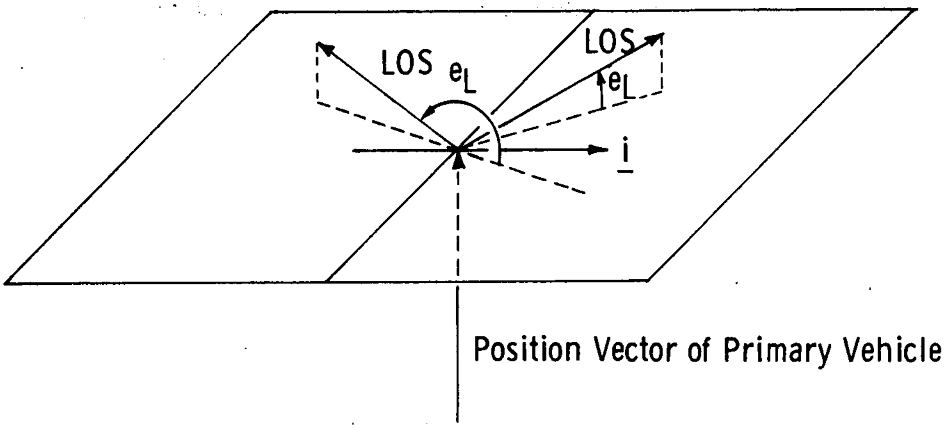
$s_{man} =$

12 Establish a primary vehicle's position vector by solving the TPI geometry problem based on e_L and Δh (see Desired Position Routine in Section 5 and Figure 2). Compute the primary vehicle's coelliptic velocity at this point. Update this state vector through Δt to establish the target vector. Compute Lambert solution to establish maneuver.

13 Establish a target vector by solving the TPI geometry problem based on e_L and Δh. Compute Lambert solution to establish maneuver.

(Before calculating the following maneuvers, the target position vector is offset Δr and Δh.)

14 Compute Lambert solution to establish maneuver.



$\underline{i}$ = Unit horizontal in forward direction for primary vehicle

LOS = Line of Sight

1. If the LOS projection on $\underline{i}$ is positive:
   a. When the LOS is above the horizontal plane, $0 < e_L < \pi/2$
   b. When the LOS is below the horizontal plane, $3\pi/2 < e_L < 2\pi$
2. If the LOS projection on $\underline{i}$ is negative:
   a. When the LOS is above the horizontal plane, $\pi/2 < e_L < \pi$
   b. When the LOS is below the horizontal plane, $\pi < e_L < 3\pi/2$

Figure 2. Definition of the Elevation Angle e_L

s_{man} (cont.) =

21 Compute a horizontal maneuver to hit the target aimpoint.

22 Compute a maneuver along the velocity vector to hit the target aimpoint.

23 Compute maneuver to establish perigee/ apogee at target aimpoint.

There is a minimum Δv option associated with maneuvers $s_{man} = 12$ and 14. This option is controlled with the optimum switch s_{opt} :

s_{opt} =

-1 Minimize the magnitude of the first maneuver by varying Δt, the time of update of the target vehicle.

-2 Minimize the sum of the magnitudes of the first and next maneuvers (based on a coelliptic parting velocity) by varying Δt , the target's update time.

1 Minimize the magnitude of the first maneuver by varying Δt, the time between the next maneuver and the TPI time.

2 Minimize the sum of the magnitude of the first and next maneuvers (based on a coelliptic parting velocity) by varying Δt, the time between the next maneuver and the TPI time.

This minimization is accomplished by driving the slope (Δv / independent variable) to zero using a Newton Raphson iteration scheme.

1.2.3 Maneuver Options in Integrated Generation of Maneuver Segment

The integrated generation of a maneuver segment involves an iterative solution to determine a maneuver sequence which cannot be sequentially solved for its maneuver segment components. The maneuver is computed by guessing its magnitude, assigning a direction and plane through selection of the direction switch s_{direct}, updating the primary vehicle' s state vector after selecting switch s_{update_p}

and then-calling additional maneuver segments until reaching the point at which the terminal constraint is to be attained. The maneuver is then iteratively determined by satisfying the terminal constraint. The number of additional maneuver segments and the nature of the terminal constraint are controlled by the terminal constraint switch s_{term}

$$
s_{term} = \begin{cases}
-2, -3, \ldots & \text{The terminal constraint is a phasing constraint and it occurs at the } |s_{term}| \text{ maneuver point from the start of the maneuver segment.} \\
2, 3 \ldots < 10 & \text{The terminal constraint is a height constraint and it occurs at the } s_{term} \text{ maneuver point from the start of the maneuver segment.} \\
(10 < s_{term} < 100) & \text{Both a height and phasing constraint occur at the same maneuver point. The first digit } n_1 \text{ of } s_{term} \text{ represents a phasing constraint that occurs at the } n_1 \text{ maneuver point from the start of the phasing maneuver segment. The last digit } n_2 \text{ of } s_{term} \text{ represents a height constraint that occurs at the } n_2 \text{ maneuver point from the start of the height maneuver segment.}
\end{cases}
$$

1.2.4 Summary of the Maneuver Constraints

The maneuver constraints can be divided into the following catagories (see Figure 3).

Primary vehicle update constraints
Target vehicle update constraints
Initial velocity constraints
Offset constraints
Terminal constraints
Traverse constraints

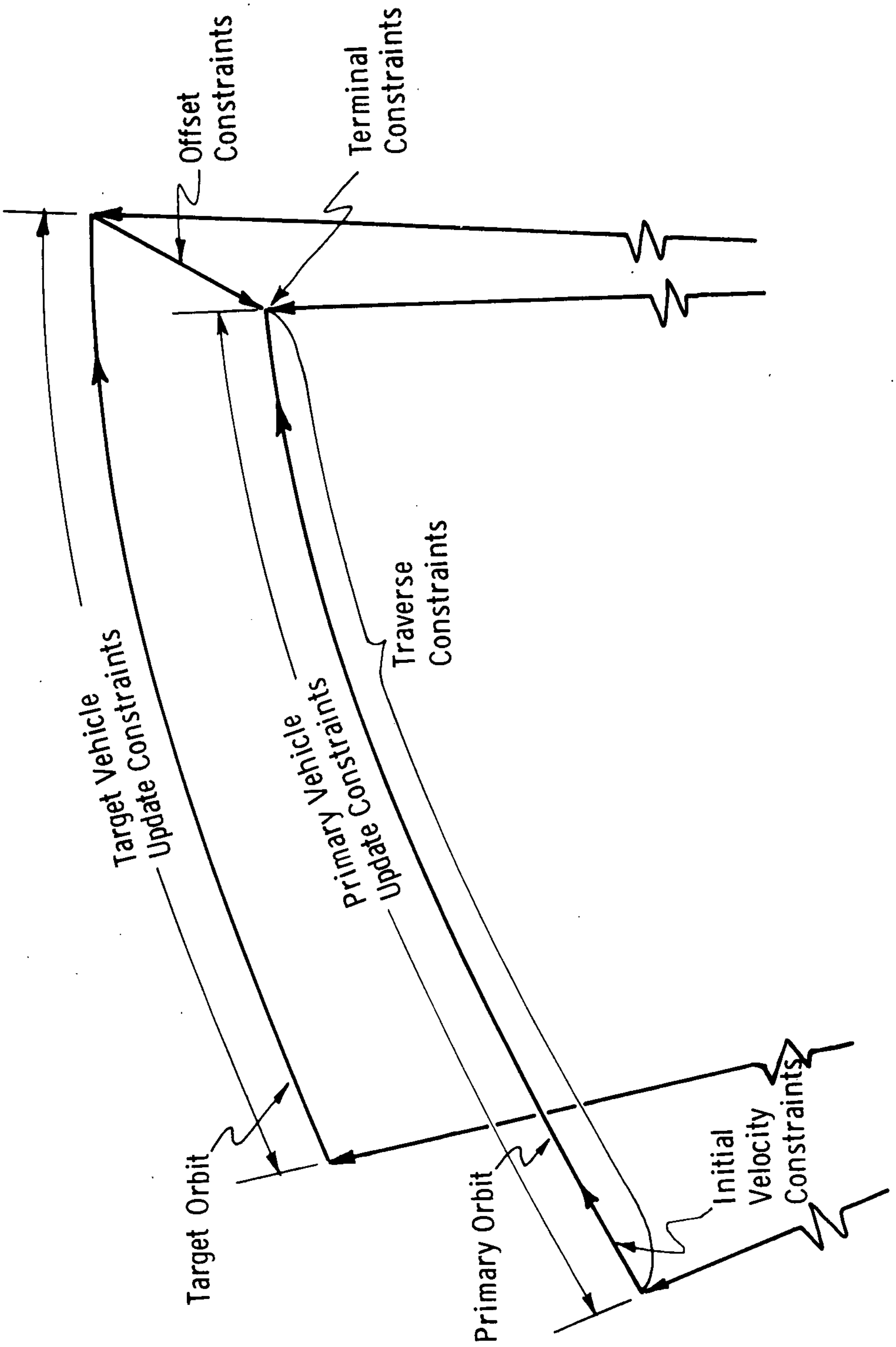

Offset Constraints
Terminal Constraints
Traverse Constraints
Target Vehicle Update Constraints
Primary Vehicle Update Constraints
Target Orbit
Primary Orbit
Initial Velocity Constraints

Figure 3. Constraint Catagories on a Maneuver Segment

9.8.1.1 Rendezvous Targeting (continued)

Table 1 contains a detailed listing of the constraints. The three constraints (four in the case of noncoplanar traverses) which govern a maneuver segment cannot be chosen arbitrarily from this list. One of the justifications for presenting the three methods of generating a maneuver segment was to allow the constructor of the rendezvous sequence to easily choose compatible sets of constraints.

9.8.1.1 Rendezvous Targeting (continued)

TABLE 1

DETAILED LISTING OF CONSTRAINTS

(Sheet 1 of 2)

Primary and Target Vehicle Update Constraints

- Delta time
- Initial and final time
- Central angle
- Number of revolutions
- Terminal position vector

Initial Velocity Constraints

- Plane
  - Parallel to target orbit
  - Parallel to primary orbit
- Direction
  - Horizontal
  - Along velocity vector
- Magnitude
  - Circular
  - Coelliptic
  - Altitude change
  - Specified

Offset Constraints

- Range
- Altitude
- Elevation angle

Terminal Constraints

- Height
- Phase

9.8.1.1 Rendezvous Targeting (continued)

TABLE 1

DETAILED LISTING OF CONSTRAINTS

(Sheet 2 of 2)

Traverse Constraints

- Minimum Fuel
  - One maneuver optimization
  - Two maneuver optimization
- Apogee/Perigee designation
- Horizontal maneuver
- Tangential maneuver
- Lambert (time)

NOMENCLATURE

a	Semi-major axis of a conic
a_i	Alarm code i
a_1	Failure in fuel optimization loop
a_2	Failure in height loop
a_3	Failure in phasing loop
a_4	Failure in obtaining Lambert solution in General Maneuver Routine
a_5	Failure to find perigee/apogee in Search Routine
a_6	Failure to find time corresponding to elevation angle in Search Routine
a_7	Failure to find desired position vector in Desired Position Routine
a_8	Failure to update through θ in Update Routine
c	Iteration counter
c_h	Height iteration counter
c_p	Phase iteration counter
c_1, c_2, c_3	Intermediate variables
Δh	Delta altitude
Δr	Delta range along orbit (determines update time $= \Delta r / v_{TF}$)
Δr_{proj}	Delta projected position

9.8.1.1 Rendezvous Targeting (continued)

Δt	Delta time
$\Delta \underline{v}$	Maneuver velocity
$\Delta \underline{v}_{LOS}$	Maneuver in line-of-sight coordinates
$\Delta \underline{v}_{LV}$	Maneuver in local vertical coordinates
Δv_h	Δv used during height maneuver
Δv_p	Δv used during phasing maneuver
Δv_T	Delta velocity used in fuel minimization loop
Δx	Delta independent variable
e	Error
e_c	Eccentricity
e_h	Height error
e_p	Phasing error
e_L	Elevation angle (defined in Figure 2)
$\underline{i}$	Unit vector
$\underline{i}_N$	Unit normal to the plane used in powered flight guidance
i	Number of the maneuver
i_{max}	Maximum number of maneuvers in rendezvous sequence currently being computed
m	Estimated vehicle mass
M	Rotational matrix
$\underline{n}$	Vector normal to the orbital plane
n_r	Number of revolutions

n_{rev}	Number of complete revolutions in multi-revolution transfer
p	Partial used in Newton Raphson iteration
r	Distance ratio
$\underline{r}$	Position vector
$\underline{r}_D$	Desired position vector
$\underline{r}_{1c}$	Target vector used in powered flight guidance
s_{astro}	Astronaut overwrite switch
s_{coplan}	Coplanar switch
s_{direct}	Maneuver direction switch
s_{eng}	Engine select switch
s_{exit}	Program exit switch
s_{fail}	Failure switch
s_{man}	Maneuver switch
s_{opt}	Maneuver optimizing switch
s_{outp}	Out-of-plane switch
s_{pert}	Perturbation switch
s_{phase}	Phase match switch
s_{proj}	Projection switch
s_{rdes}	Desired position switch
s_{soln}	Solution switch
s_{search}	Search switch
s_{term}	Terminal constraint switch
s_{update}	Update switch

9.8.1. Rendezvous Targeting (continued)

t	Time
t_F	Final time
$\underline{v}$	Velocity vector
v_v	Vertical component of velocity
v_c	Circular velocity
x	Independent variable in Iteration Routine
y_P, $\dot{y}_P$, $\dot{y}_T$	Out-of-plane parameters (see Figure 6a)
α	Radial component of velocity divided by v_C
β	Horizontal component of velocity divided by v_C
ϵ_1	Tolerance on time in fuel optimizing loop
ϵ_2	Tolerance on height in height loop
ϵ_3	Tolerance on central angle in phasing loop
ϵ_4	Tolerance on transfer angle's proximity to 180 degrees in General Maneuver Routine
ϵ_5	Tolerance on central angle in Search Routine
ϵ_6	Tolerance on elevation angle in Search Routine
ϵ_7	Tolerance on central angle in Desired Position Routine
ϵ_8	Tolerance on central angle is Update Routine
γ	Flight path angle
μ	Gravitational constant
θ	Central angle
θ_p	Perigee angle

Subscripts

F	Final
i	Number of the maneuver
I	Initial
LOS	Line-of-sight
LV	Local vertical
N	New
0	Old
P	Primary
S	Stored
T	Target
TA	Target for primary vehicle

2. FUNCTIONAL FLOW DIAGRAMS

The rendezvous targeting program consists of two major parts—a generalized maneuver subroutine which basically computes a maneuver and updates the state vectors of both vehicles to the time of the next maneuver and a main program which sequentially calls the subroutine to assemble a rendezvous sequence. These programs call a number of subroutines which are briefly described below and in detail in Section 5.

Search - To update the state vectors to either a specified apsidal crossing, a time, or an elevation angle.

Phase Match - To phase match the target vehicle's state vector to the primary vehicle' s position vector.

Desired Position - To compute a desired position vector to be used in a phasing constraint.

Update - To update a state vector through a specified interval.

Coelliptic Maneuver - To compute a coelliptic velocity vector.

Iteration - To determine a new estimate of the independent variable in a Newton Raphson iteration scheme.

The functional flow diagram for the main program is shown in Figure 4. The main function of this program is to sequentially call the General Maneuver Routine to compute each maneuver segment for maneuvers numbered from i to i_{max}. There are three major options that can be exercised prior to the calculation of the first maneuver segment:

(1) A search for the time of the first maneuver. This time can be specified by:

(a) An elevation angle, which is to be attained at the maneuver time.

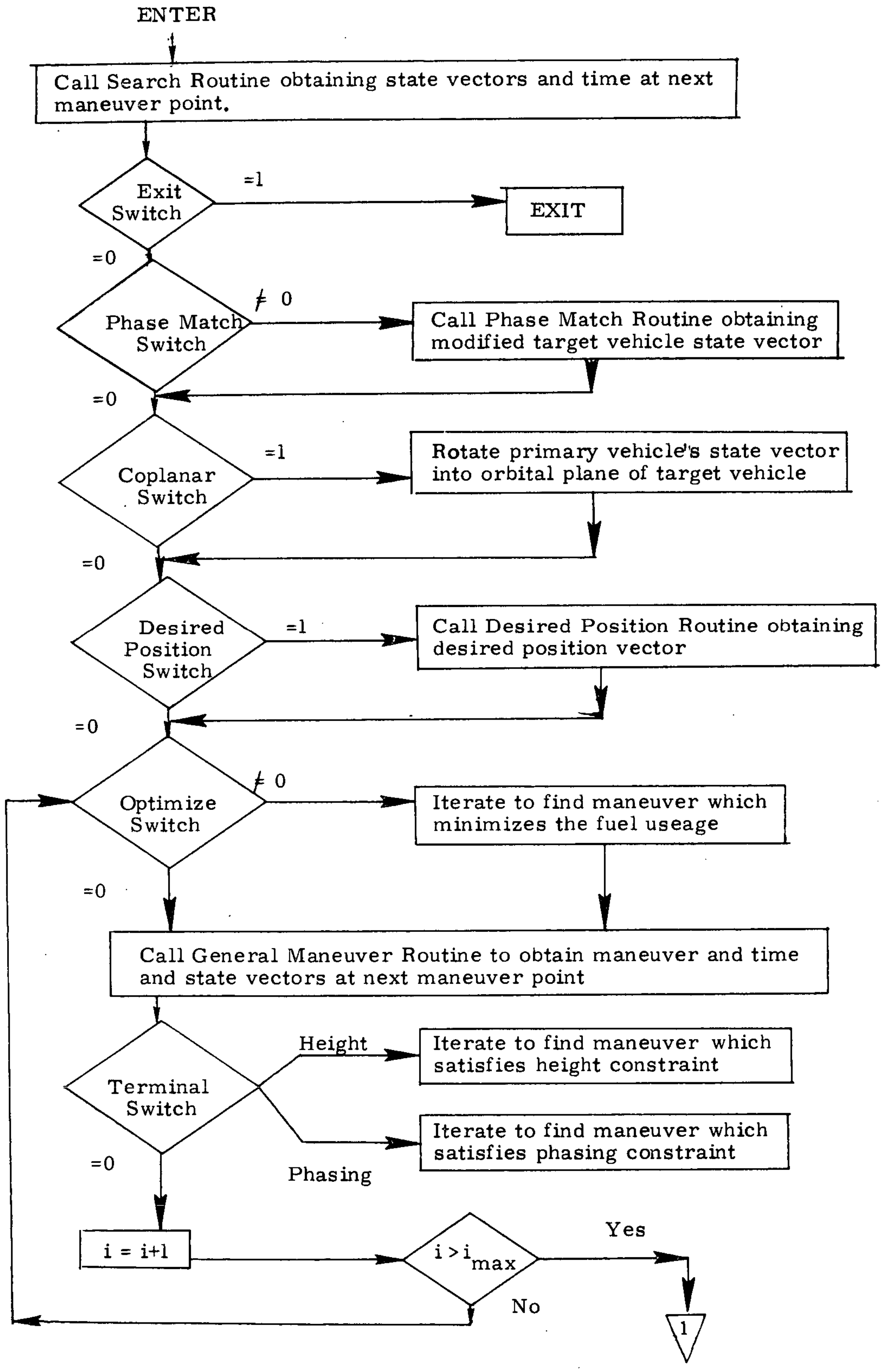

ENTER
Call Search Routine obtaining state vectors and time at next maneuver point.
Exit Switch
=1
EXIT
=0
Phase Match Switch
≠ 0
Call Phase Match Routine obtaining modified target vehicle state vector
=0
Coplanar Switch
=1
Rotate primary vehicle's state vector into orbital plane of target vehicle
=0
Desired Position Switch
=1
Call Desired Position Routine obtaining desired position vector
=0
Optimize Switch
≠ 0
Iterate to find maneuver which minimizes the fuel useage
=0
Call General Maneuver Routine to obtain maneuver and time and state vectors at next maneuver point
Terminal Switch
Height
Iterate to find maneuver which satisfies height constraint
Iterate to find maneuver which satisfies phasing constraint
Phasing
=0
i = i+1
i > i max
Yes
No
1

Figure 4a. Main Program - Functional Flow Diagram

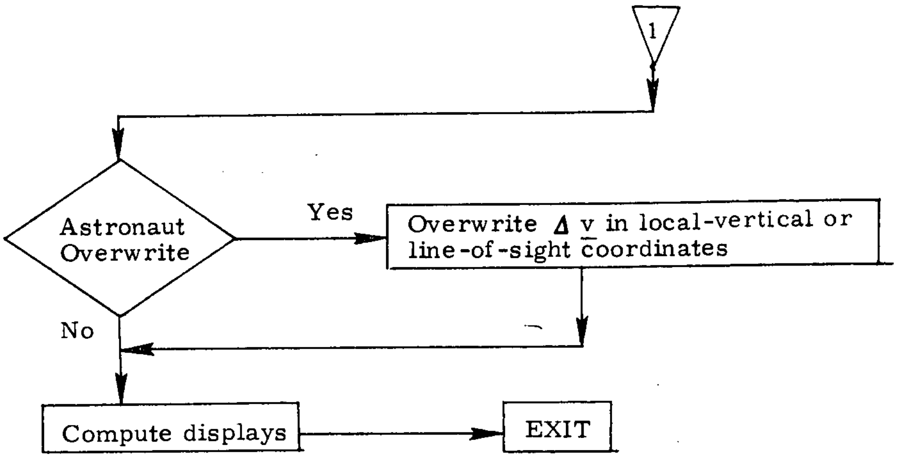

1
Astronaut Overwrite
Yes
No
Overwrite Δ v in local-vertical or line-of-sight coordinates
Compute displays
EXIT

Figure 4b. Main Program - Function Flow Diagram

(b) Whether the next maneuver should occur at the next apsidal crossing, the next perigee crossing or the nth apsidal crossing.

(2) A phase matching of the state vector.

(3) A rotation of the primary vehicle' s state vector into the plane of the target vehicle.

There are three separate iterative loops built around the call to the general maneuver routine. One loop serves to minimize the fuel used during a maneuver segment with the options determined by the optimizing switch.

The other two iterative loops involve maneuver segments which contain constraints that do not allow the explicit calculation of the maneuver. These constraints are height and phasing constraints imposed at the end of a maneuver segment and controlled with the terminal switch. The iterative loop will involve several maneuver segments if sufficient constraints are not imposed to solve each segment uniquely.

The functional flow diagram for the general maneuver subroutine is shown in Figure 5. This routine generates a maneuver in one of three ways:

(1) As an explicit function of the initial state vector.

(2) As a Lambert maneuver following an update of the passive vehicle to establish the target vector.

(3) As a horizontal, tangential or perigee/apogee maneuver following an update of the passive vehicle to establish the target vector.

Following an update of both vehicle' s state vectors to the time of the next maneuver, the Δv used or the terminal height/phase errors are calculated as required.

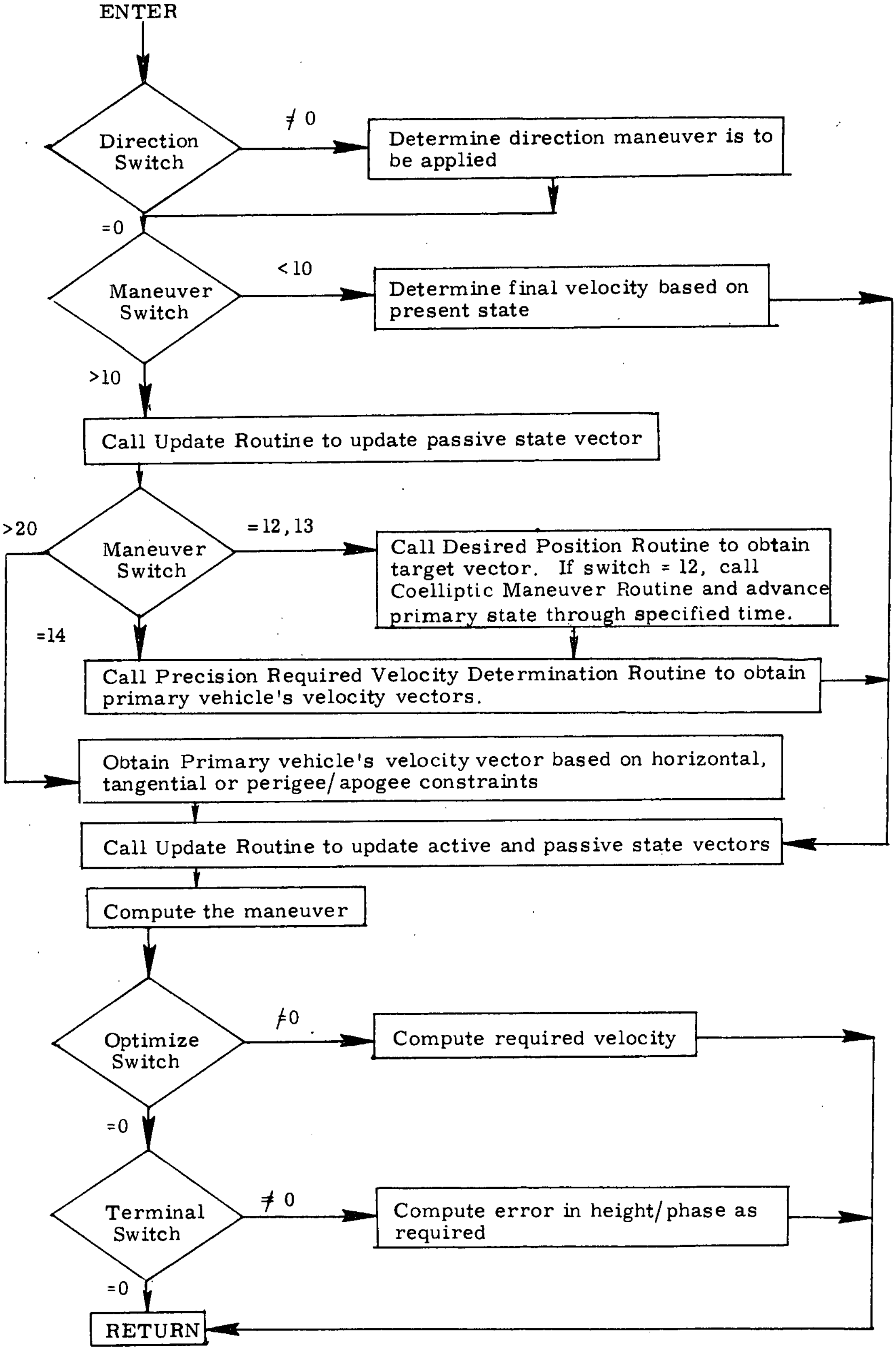

ENTER
Direction Switch
≠ 0
Determine direction maneuver is to be applied
=0
Maneuver Switch
< 10
Determine final velocity based on present state
>10
Call Update Routine to update passive state vector
>20
Maneuver Switch
=12,13
Call Desired Position Routine to obtain target vector. If switch = 12, call Coelliptic Maneuver Routine and advance primary state through specified time.
=14
Call Precision Required Velocity Determination Routine to obtain primary vehicle's velocity vectors.
Obtain Primary vehicle's velocity vector based on horizontal, tangential or perigee/apogee constraints
Call Update Routine to update active and passive state vectors
Compute the maneuver
Optimize Switch
≠0
Compute required velocity
=0
Terminal Switch
≠ 0
Compute error in height/phase as required
=0
RETURN

Figure 5. General Maneuver Routine - Functional Flow Diagram

3. INPUT AND OUTPUT VARIABLES

The inputs to the orbiter rendezvous targeting program can be divided into five catagories.

Pre-Maneuver Switches

Upon selecting a maneuver from the rendezvous sequence, these switches (specified for each maneuver) serve in determining the state vectors at the maneuver point, the out-of-plane parameters and the calculation of a desired position vector. These inputs can also be used in determining the time of a specified apsidal crossing or the time at which a specified elevation angle is to be attained.

Coplanar switch

$$s_{coplan} = \begin{cases} 0 & \text{Bypass} \\ 1 & \text{Rotate primary state vector into plane} \\ & \text{of target vehicle's orbit} \end{cases}$$

Exit switch

$$s_{exit} = \begin{cases} 0 & \text{Bypass} \\ 1 & \text{Exit from routine} \end{cases}$$

Out-of-plane switch

$$s_{outp} = \begin{cases} 0 & \text{Bypass} \\ 1 & \text{Compute out-of-plane parameters} \\ 2 & \text{Compute out-of-plane parameters and} \\ & \text{modify maneuver by } -\dot{y}_P \end{cases}$$

Perturbation switch

$$s_{pert} = \begin{cases} 0 & \text{Do conic state vector updates} \\ 1 & \text{Include oblateness based on } J_2 \\ \ldots & \text{Other perturbations as required} \end{cases}$$

Phase match switch

$$s_{phase} = \begin{cases} 0 & \text{Bypass} \\ 1 & \text{Phase match state vectors (target leading} \\ & \text{primary)} \\ 2 & \text{Phase match state vectors based on target} \\ & \text{leading primary by more than } 360^o \\ -1 & \text{Phase match state vectors (primary} \\ & \text{leading target)} \\ -2 & \text{Phase match state vectors based on primary} \\ & \text{leading target by more than } 360^o \end{cases}$$

9.8.1.1 Rendezvous Targeting (continued)

Desired position switch

$$s_{rdes} = \begin{cases} 0 & \text{Bypass} \\ 1 & \text{Compute desired position vector} \end{cases}$$

Search switch

$$s_{search} = \begin{cases} -4 & \text{Compute elevation angle} \\ -3 & \text{Search for elevation angle} \\ -2 & \text{Update to time } t_i \\ -1 & \text{Search for next perigee crossing} \\ 0 & \text{Bypass} \\ n & \text{Search for the nth apsidal crossing} \\ & (n>0) \end{cases}$$

Maneuver Switches

These switches (specified for each maneuver) set the constraints employed in determining the maneuver segments.

Direction switch

$$s_{direct} = \begin{cases} -2 & \Delta v \text{ in direction of primary's velocity} \\ & \text{vector, parallel to primary's orbital} \\ & \text{plane} \\ -1 & \Delta v \text{ in horizontal direction, parallel} \\ & \text{to primary's orbital plane} \\ 0 & \text{Bypass} \\ 1 & \Delta v \text{ in horizontal direction, parallel to} \\ & \text{target's orbital plane} \\ 2 & \Delta v \text{ in direction of primary's velocity} \\ & \text{vector, parallel to target's orbital plane} \end{cases}$$

Maneuver switch

$$s_{man} = \begin{cases} 1 & \Delta v \text{ is specified} \\ 2 & \Delta v \text{ is based on coelliptic velocity} \\ 3 & \Delta v \text{ is based on circular velocity} \\ 4 & \Delta v \text{ is based on altitude change} \\ 12 & \text{Lambert maneuver using target vector} \\ & \text{obtained by updating the primary vehicle's} \\ & \text{coelliptic state at TPI} \\ 13 & \text{Lambert maneuver using primary vehicle's} \\ & \text{TPI position as a target vector} \\ 14 & \text{Lambert maneuver to offset target vector}^{*} \\ 21 & \text{Horizontal maneuver to offset target vector}^{*} \end{cases}$$

*Obtained by updating target state and offsetting $(\Delta r, \Delta h)$.

C.7

9.8.1.1 Rendezvous Targeting (continued)

Maneuver switch (cont.)

$$s_{man} = \begin{cases} 22 & \text{Tangential maneuver to offset target point*} \\ 23 & \text{Perigee/apogee insertion at offset target point*} \end{cases}$$

Maneuver optimizing switch

$$s_{opt} = \begin{cases} 0 & \text{Bypass} \\ 1 & \text{Minimize } \Delta v_i \\ 2 & \text{Minimize } \Delta v_i + \Delta v_{i+1} \end{cases}$$

Multi-revolution solution switch

$$s_{sohn} = \begin{cases} -1 & \text{Solution with smallest initial flight path angle (measured from local vertical)} \\ 1 & \text{Solution with largest initial flight path angle} \end{cases}$$

Terminal constraint switch

$$s_{term} = \begin{cases} n & (<0) \text{ compute phasing error and back up } -(n+1) \text{ maneuvers for start of phase loop} \\ 0 & \text{Bypass} \\ n & (0 < n < 10) \text{ Compute height error and back up } n - 1 \text{ maneuvers for start of height loop} \\ n & (10 < n < 100) \text{ Phase and height loop terminate on same maneuver. For phase loop back up } x - 1 \text{ (where } x \text{ is first digit of } n\text{) maneuvers for start of phase loop. For height loop back up } y - 1 \text{ (where } y \text{ is last digit of } n\text{) maneuvers for start of height loop} \end{cases}$$

Update switch

$$s_{update} = \begin{cases} 0 & \text{Bypass} \\ 1 & \text{Update through } t_F - t \\ 2 & \text{Update through } \Delta t \\ 3 & \text{Update through } n \\ 4 & \text{Update through } \theta \\ 5 & \text{Update to be colinear with } \underline{r}_D \end{cases}$$

Parameter Values

The parameter values (specified for each maneuver) are values for the constrained parameters.

*Obtained by updating target state and offsetting . (Δr , Δh)

9.8.1.1 Rendezvous Targeting ;(continued)

Δh	Delta altitude
Δh_F	Delta altitude, final
Δr	Delta range along orbit
Δt	Delta time
Δv	Maneuver magnitude
n_r	Number of revolutions
n_{rev}	Number of complete revolutions in multi-revolution transfer
t_F	Final time
e_L	Elevation angle

Post-Maneuver Switches

These switches (specified for each maneuver) determine the options available following the calculation of the maneuver(s).

Astronaut overwrite switch

$$s_{astro} = \begin{cases} 0 & \text{Bypass} \\ 1 & \text{Overwrite maneuver in local vertical coordinates} \\ 2 & \text{Overwrite maneuver in line-of-sight coordinates} \end{cases}$$

Maneuver Call Variables

The maneuver call variables have to be specified for each call to the maneuver sequence.

$\underline{r}_P, \underline{v}_P$	State vector of the primary vehicle
$\underline{r}_T, \underline{v}_T$	State vector of the target vehicle
i	Maneuver number
t	Current time
t_i	Time of the i^{th} maneuver
s_{eng}	Engine select switch
m	Estimated vehicle mass

Depending on the rendezvous sequence, there may also be some switches that have to be modified as a function of the maneuver number.

Excluding the maneuver call variables, all the input variables can be set prior to the flight.

The output parameters for the initial maneuver in the sequence are more complete than for the succeeding maneuvers.

Output Parameters for the Initial Maneuver

Δv_i	Maneuver magnitude
$\Delta \underline{v}_{LOS\,i}$	Maneuver in line of sight coordinates
$\Delta \underline{v}_{LV\,i}$	Maneuver in local vertical coordinates
$\underline{r}_{1c}$	Target vector used in Powered Flight Guidance Routine (See Ref. 1)
$\underline{i}_N$	Unit normal to plane used in same routine

Other parameters such as delta altitude, phasing angle, elevation angle and perigee altitude can be computed as required.

Output Parameters for the Other Maneuvers in the Sequence

t	Time of the maneuver
Δv	Maneuver magnitude

Illustration of Inputs

Table 2 contains a set of inputs for the Orbiter targeting program based on the five maneuver Skylab rendezvous configuration. The following switches and parameters are not used

$$s_{astro},\ s_{exit},\ s_{opt},\ s_{outp},\ s_{soln},\ \Delta r,\ n_{rev}$$

The elevation angle e_L and perturbation switch s_{pert} must also be specified. These inputs, plus those in Table 2, are all set prior to the mission so they will not have to be inserted by the astronaut. The astronaut will have to modify the following quantities upon resetting the maneuver number as well as inserting the time of the next maneuver.

$i = 2$: $s_{term_2} = 0$, $s_{term_4} = 32$

$i = 3$: $s_{updateP_3} = 1$, $s_{man_3} = 12$, $\Delta t_3 = t_{NSR\text{-}TPI}$

$i = 4$: $s_{term_4} = 0$

TABLE 2

INPUT VARIABLES FOR SKYLAB RENDEZVOUS CONFIGURATION

Input Variable	Maneuver				
	1 (NC1)	2 (NC2)	3 (NCC)	4 (NSR)	5 (TPI)
s_{coplan}	1	1	0	1	0
s_{direct}	1	1	0	0	0
s_{man}	1	1	1	2	14
s_{phase}	1	1	0	0	0
s_{search}	-2	-2	-2	-2	-3
s_{rdes}	1	1	0	0	0
s_{term}	0	1	0	42	0
s_{update_P}	3	3	2	1	0
s_{update_T}	2	2	2	2	4
i_{max}	4	4	3	4	5
t_F	t_{TPI}	t_{TPI}	t_{TPI}	t_{TPI}	0
θ	0	0	0	0	$\theta_{TPF-TPI}$
Δt	0	0	$\Delta t_{NSR-NCC}$	0	0
Δh	0	Δh_{NCC}	Δh_{TPI}	Δh_{TPI}	0
Δh_F	Δh_{TPI}	Δh_{TPI}	0	0	0
Δv	Δv_{NC1}	Δv_{NC2}	Δv_{NCC}	0	0
n_r	$n_{rNC1-NC2}$	$n_{rNC2-NCC}$	0	0	0

4. DESCRIPTION OF EQUATIONS

The only equations contained in this document which are not trivial are those involved in computing the traverse between two specified position vectors. The required equations can be derived from the equation of the conic expressed in the form

$$r = r_F / r_I = \beta^2 / \left[1 + e_c \cos (\theta + \theta_P) \right] \qquad (1)$$

where

$$e_c = [\alpha^2 \beta^2 + (\beta^2 - 1)^2]^{1/2}$$

$$\theta_p = \cos^{-1} \left[(\beta^2 - 1) / e_c \right] \quad \text{(perigee angle)}$$

$$\alpha = \underline{v}_I \cdot \underline{r}_I / r_I v_c$$

$$\beta = (v_I^2 / v_c^2 - \alpha^2)^{1/2}$$

$$v_c = (\mu / r_I)^{1/2}$$

α and β are the normalized radial and horizontal components of velocity; $\underline{r}_I$ and $\underline{v}_I$ constitute the initial state vector.

For a maneuver that is constrained to be in a horizontal direction, Eq. (1) can be solved directly for β

$$\beta = [\alpha \pm (\alpha^2 - 4 c_1 c_2)^{1/2}] / 2 c_1$$

where

$$c_1 = (\cos \theta - 1/r) / \sin \theta$$

$$c_2 = (1 - \cos \theta) / \sin \theta$$

As there has to be both a positive and negative β solution to this equation (one trajectory in each rotational direction), the sign choice is resolved in favor of plus β.

For a maneuver that is applied along the velocity vector, the flight path angle γ_I is to be held fixed. Using Eq. (1)

$$\tan \gamma_I = \alpha / \beta = (c_1 \beta^2 + c_2) / \beta^2$$

Therefore

$$\beta = \left[c_2 / (\tan \gamma_I - c_1) \right]^{1/2}$$

For a maneuver which is to result in a perigee point at the terminal position vector, Eq. (1) can be solved for the following conditions

9.8.1.1 Rendezvous Targeting (continued)

$$\theta + \theta_p = 0 : \; r = \beta^2 / (1 + e_c)$$

$$\theta + \theta_p = \theta \text{ and } \dot{r} = 1 : \; 1 = \beta^2 / (1 + e_c \cos\theta)$$

Eliminating e_c between these two equations results in

$$\beta = [\, r(\cos\theta - 1) / (\cos\theta - r)\,]^{1/2}$$

In all three of these cases, α can be obtained after finding β

$$\alpha = (c_1 \beta^2 + c_2) / \beta$$

As the above equations are undefined for a 180 degree transfer, transfers in the near vicinity of 180 degrees are based on

$$\beta = [\, 2r / (1 + r)\,]^{1/2},$$

the Hohmann horizontal component of velocity. For the 180 degree transfer, the final radial component of velocity equals the initial radial component of velocity.

5. DETAILED FLOW DIAGRAMS

Figures 6 and 7 contain the detailed flow diagrams of the main Orbiter rendezvous targeting program and the general maneuver routine, respectively. The following six routines are called by these two programs.

Iteration Routine

This routine contains a Newton Raphson iterative driver based on numerically computed partials. The routine computes a new estimate of the dependent variable x and returns the old values of the error e and x. If the iteration counter c exceeds 15, a convergence switch s_{conv} is set equal to one.

Coelliptic Maneuver Routine

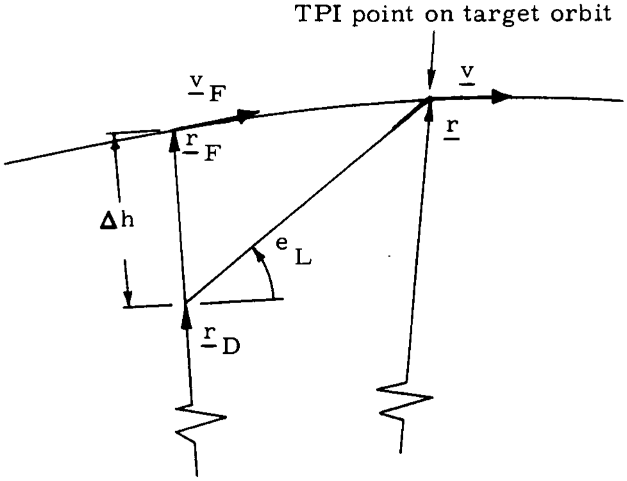
This routine computes a coelliptic velocity vector $\underline{v}_N$ based on a target vehicle's state vector and a delta altituae.

Phase Match

This routine phase matches the target state vector to the primary state vector. The controlling switch (s_{phase}) equals tvo if the leading vehicle leads the other vehicle by more than one revolution: otherwise the switch equals one. If the primary vehicle leads to target vehicle, the switch is negative.

Desired Position Routine

The routine contains an iterative search to determine a position vector $\underline{r}_D$ which satisfies the elevation angle e_L and delta altitude Δh constraints as shown below. (This represents the TPI geometry used in Apollo and Skylab.)



Update Routine

This routine updates a state vector based on the update switch s_{update}

$$
s_{update} \begin{cases} = 1 & \text{Updates through the time } t_F - t \\ = 2 & \text{Updates through the time } \Delta t \\ = 3 & \text{Updates through } n_r \text{ revolutions} \\ = 4 & \text{Updates through the angle } \theta \\ = 5 & \text{Updates to where the orbit intersects} \\ & \text{the line defined by } \underline{r}_D \end{cases}
$$

Search Routine

This routine makes the following computations depending on the setting of the search switch s_{search}

$$
s_{search} \begin{cases} = n & \text{Finds the time of the } n^{th} \text{ apsidal crossing} \\ (>0) & \text{and updates the state vector to that time} \\ = -1 & \text{Finds the time of the next perigee crossing} \\ & \text{and updates the state vector to that time} \\ = -2 & \text{Updates the state vector through the time} \\ & t_F - t \\ = -3 & \text{Finds the time associated with a specified} \\ & \text{elevation angle and updates the state vector} \\ & \text{to that time} \\ = -4 & \text{Computes an elevation angle} \end{cases}
$$

The detailed flow charts for these routines are shown in Figures 8 to 13.

Each input and output variable in the routine and subroutine call statements can be followed by a symbol in brackets. This symbol identifies the notation for the corresponding variable in the detailed description and flow diagrams of the called routine. When identical notation is used, the bracketed symbol is omitted.

9.8.1.1 Rendezvous Targeting (continued)

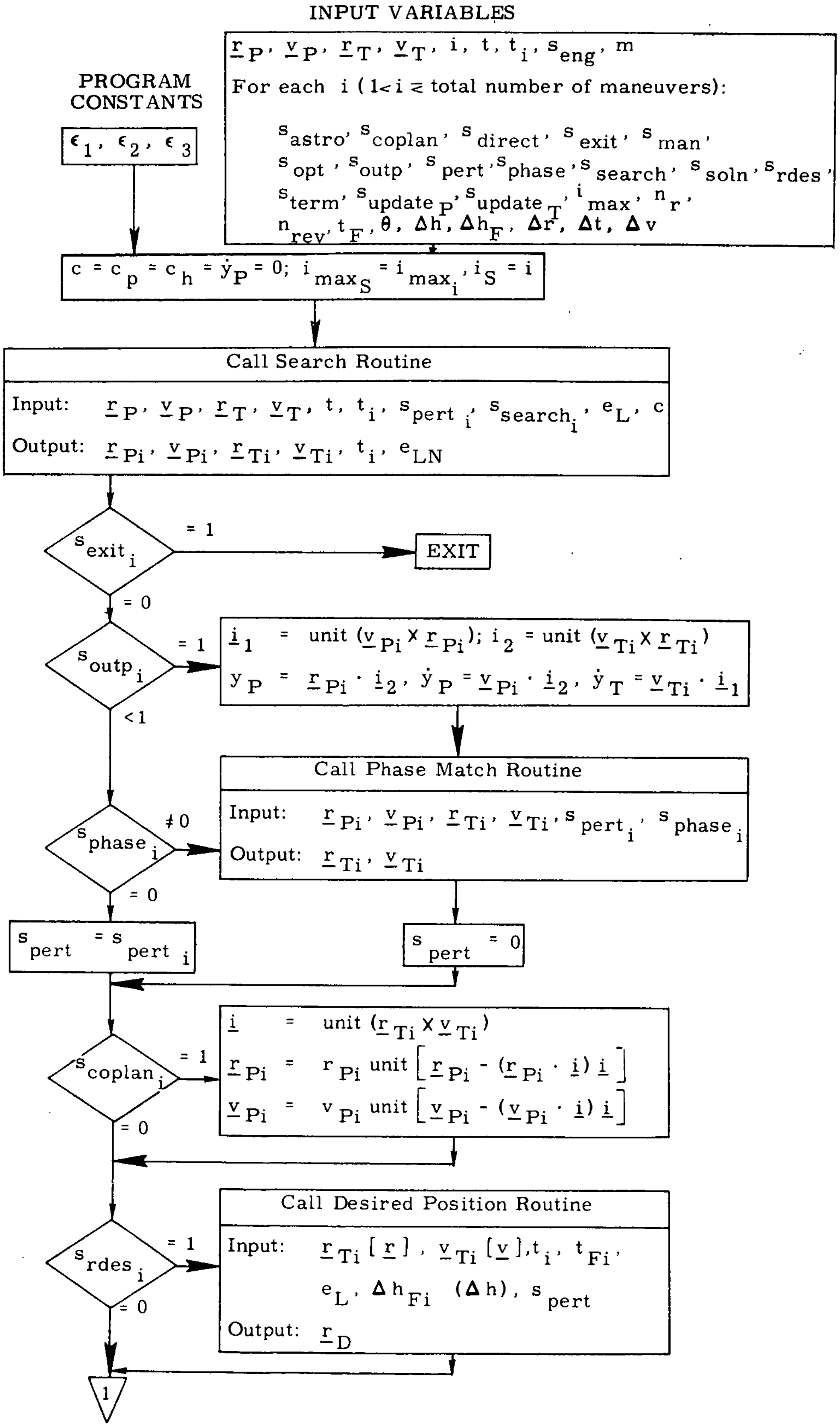

INPUT VARIABLES
$\underline{r}_P, \underline{v}_P, \underline{r}_T, \underline{v}_T$, i, t, t_i, s_{eng}, m
For each i (1< i ≤ total number of maneuvers):
$s_{astro}, s_{coplan}, s_{direct}, s_{exit}, s_{man}, s_{opt}, s_{outp}, s_{pert}, s_{phase}, s_{search}, s_{soln}, s_{rdes}, s_{term}, s_{update_P}, s_{update_T}, i_{max}, n_r, n_{rev}, t_F, \theta, \Delta h, \Delta h_F, \Delta r, \Delta t, \Delta v$
PROGRAM CONSTANTS
$\epsilon_1, \epsilon_2, \epsilon_3$
$c = c_p = c_h = \dot{y}_P = 0;\ i_{max_S} = i_{max_i}, i_S = i$
Call Search Routine
Input: $\underline{r}_P, \underline{v}_P, \underline{r}_T, \underline{v}_T$, t, t_i, s_{pert_i}, s_{search_i}, e_L, c
Output: $\underline{r}_{Pi}, \underline{v}_{Pi}, \underline{r}_{Ti}, \underline{v}_{Ti}, t_i, e_{LN}$
s_{exit_i}
= 1
EXIT
= 0
s_{outp_i}
= 1
$\underline{i}_1$ = unit $(\underline{v}_{Pi} \times \underline{r}_{Pi})$; i_2 = unit $(\underline{v}_{Ti} \times \underline{r}_{Ti})$
$y_P = \underline{r}_{Pi} \cdot \underline{i}_2,\ \dot{y}_P = \underline{v}_{Pi} \cdot \underline{i}_2,\ \dot{y}_T = \underline{v}_{Ti} \cdot \underline{i}_1$
< 1
Call Phase Match Routine
Input: $\underline{r}_{Pi}, \underline{v}_{Pi}, \underline{r}_{Ti}, \underline{v}_{Ti}, s_{pert_i}, s_{phase_i}$
Output: $\underline{r}_{Ti}, \underline{v}_{Ti}$
s_{phase_i}
≠ 0
= 0
$s_{pert} = s_{pert_i}$
$s_{pert} = 0$
s_{coplan_i}
= 1
$\underline{i}$ = unit $(\underline{r}_{Ti} \times \underline{v}_{Ti})$
$\underline{r}_{Pi} = r_{Pi}$ unit $\left[\underline{r}_{Pi} - (\underline{r}_{Pi} \cdot \underline{i})\,\underline{i}\right]$
$\underline{v}_{Pi} = v_{Pi}$ unit $\left[\underline{v}_{Pi} - (\underline{v}_{Pi} \cdot \underline{i})\,\underline{i}\right]$
= 0
Call Desired Position Routine
Input: $\underline{r}_{Ti}\,[\underline{r}]$, $\underline{v}_{Ti}\,[\underline{v}]$, t_i, t_{Fi}, e_L, Δh_{Fi} (Δh), s_{pert}
Output: $\underline{r}_D$
s_{rdes_i}
= 1
= 0
1

Figure 6a. Main Program - Detailed Flow Diagram

9.8.1.1 Rendezvous Targeting (continued)

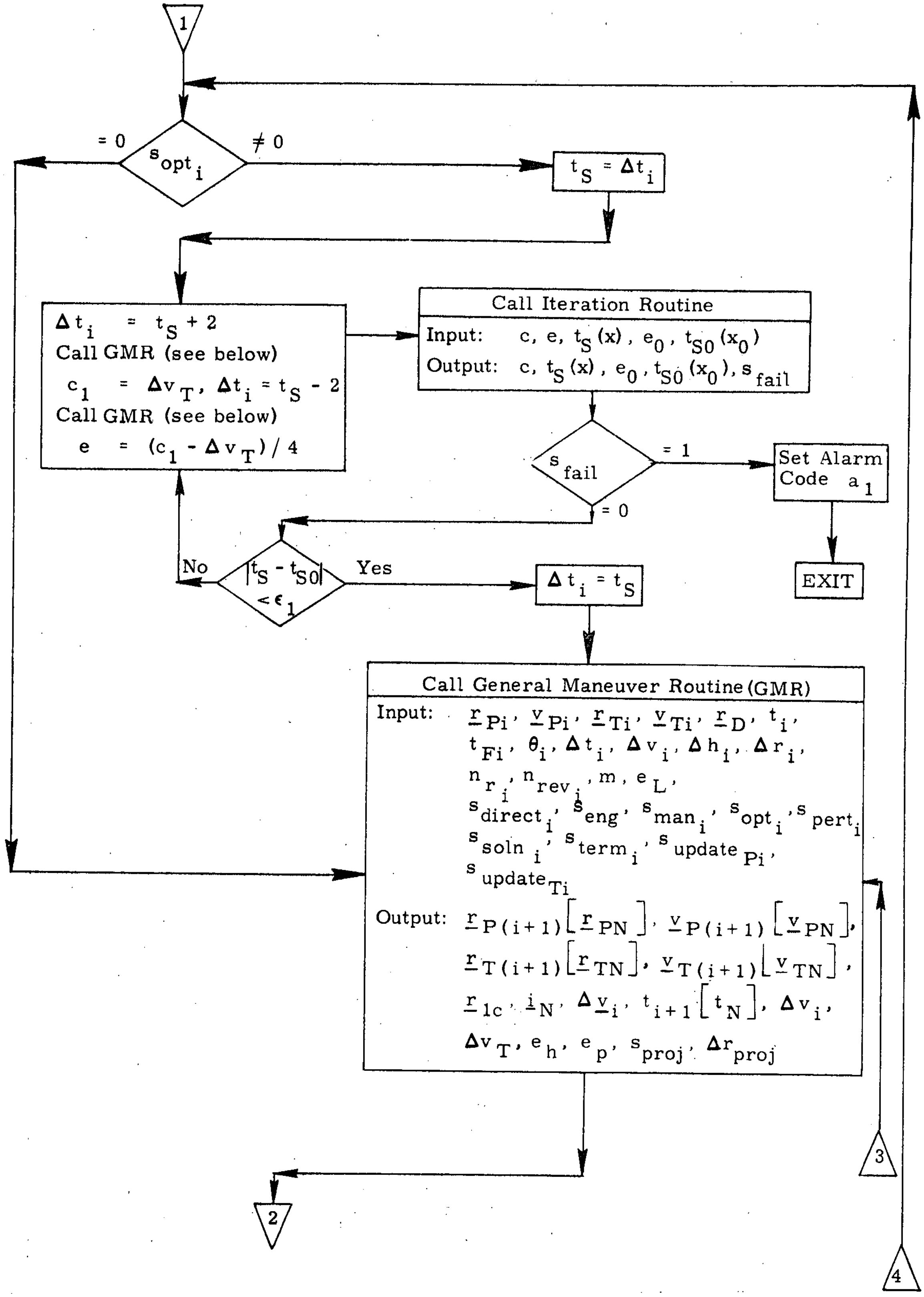

1
s_{opt_i}
= 0
≠ 0
$t_S = \Delta t_i$
$\Delta t_i = t_S + 2$
Call GMR (see below)
$c_1 = \Delta v_T$, $\Delta t_i = t_S - 2$
Call GMR (see below)
$e = (c_1 - \Delta v_T) / 4$
Call Iteration Routine
Input: $c, e, t_S(x), e_0, t_{S0}(x_0)$
Output: $c, t_S(x), e_0, t_{S0}(x_0), s_{fail}$
s_{fail}
= 1
= 0
Set Alarm Code a_1
EXIT
No
$|t_S - t_{S0}| < \epsilon_1$
Yes
$\Delta t_i = t_S$
Call General Maneuver Routine (GMR)
Input: $\underline{r}_{Pi}, \underline{v}_{Pi}, \underline{r}_{Ti}, \underline{v}_{Ti}, \underline{r}_D, t_i,$
$t_{Fi}, \theta_i, \Delta t_i, \Delta v_i, \Delta h_i, \Delta r_i,$
$n_{r_i}, n_{rev_i}, m, e_L,$
$s_{direct_i}, s_{eng}, s_{man_i}, s_{opt_i}, s_{pert_i}$
$s_{soln_i}, s_{term_i}, s_{update_{Pi}},$
$s_{update_{Ti}}$
Output: $\underline{r}_{P(i+1)} [\underline{r}_{PN}], \underline{v}_{P(i+1)} [\underline{v}_{PN}],$
$\underline{r}_{T(i+1)} [\underline{r}_{TN}], \underline{v}_{T(i+1)} [\underline{v}_{TN}],$
$\underline{r}_{1c}, \underline{i}_N, \Delta \underline{v}_i, t_{i+1} [t_N], \Delta v_i,$
$\Delta v_T, e_h, e_p, s_{proj}, \Delta r_{proj}$
2
3
4

Figure 6b. Main Program - Detailed Flow Diagram

9.8.1.1 Rendezvous Targeting (continued)

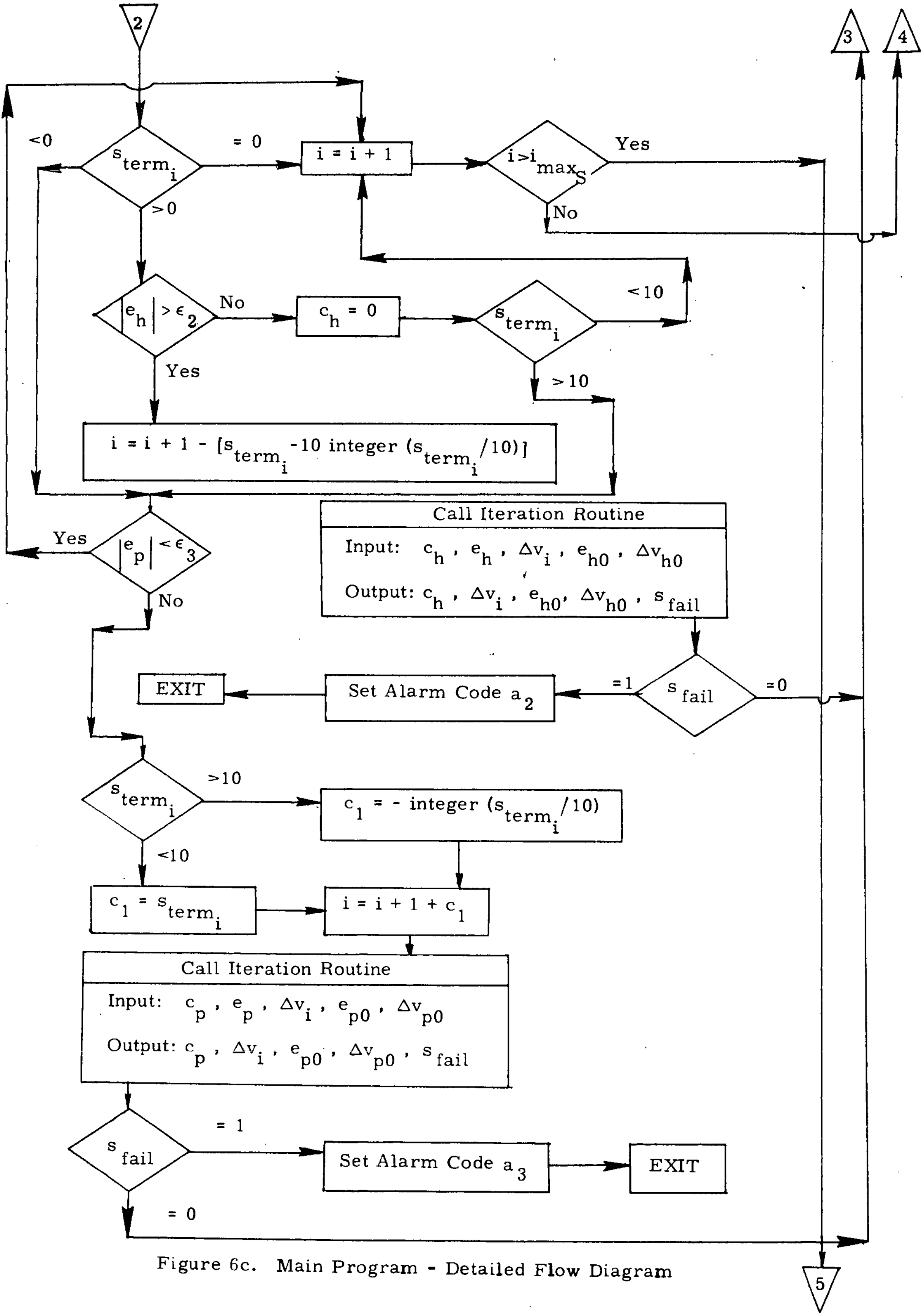

2
3
4
s_{term_i}
<0
= 0
>0
i = i + 1
$i > i_{max_S}$
Yes
No
$|e_h| > \epsilon_2$
No
Yes
$c_h = 0$
s_{term_i}
<10
>10
$i = i + 1 - [s_{term_i} - 10 \text{ integer } (s_{term_i}/10)]$
$|e_p| < \epsilon_3$
Yes
No
Call Iteration Routine
Input: c_h, e_h, Δv_i, e_{h0}, Δv_{h0}
Output: c_h, Δv_i, e_{h0}, Δv_{h0}, s_{fail}
s_{fail}
=1
=0
Set Alarm Code a_2
EXIT
s_{term_i}
>10
<10
$c_1 = -$ integer $(s_{term_i}/10)$
$c_1 = s_{term_i}$
$i = i + 1 + c_1$
Call Iteration Routine
Input: c_p, e_p, Δv_i, e_{p0}, Δv_{p0}
Output: c_p, Δv_i, e_{p0}, Δv_{p0}, s_{fail}
s_{fail}
= 1
= 0
Set Alarm Code a_3
EXIT
5

Figure 6c. Main Program - Detailed Flow Diagram

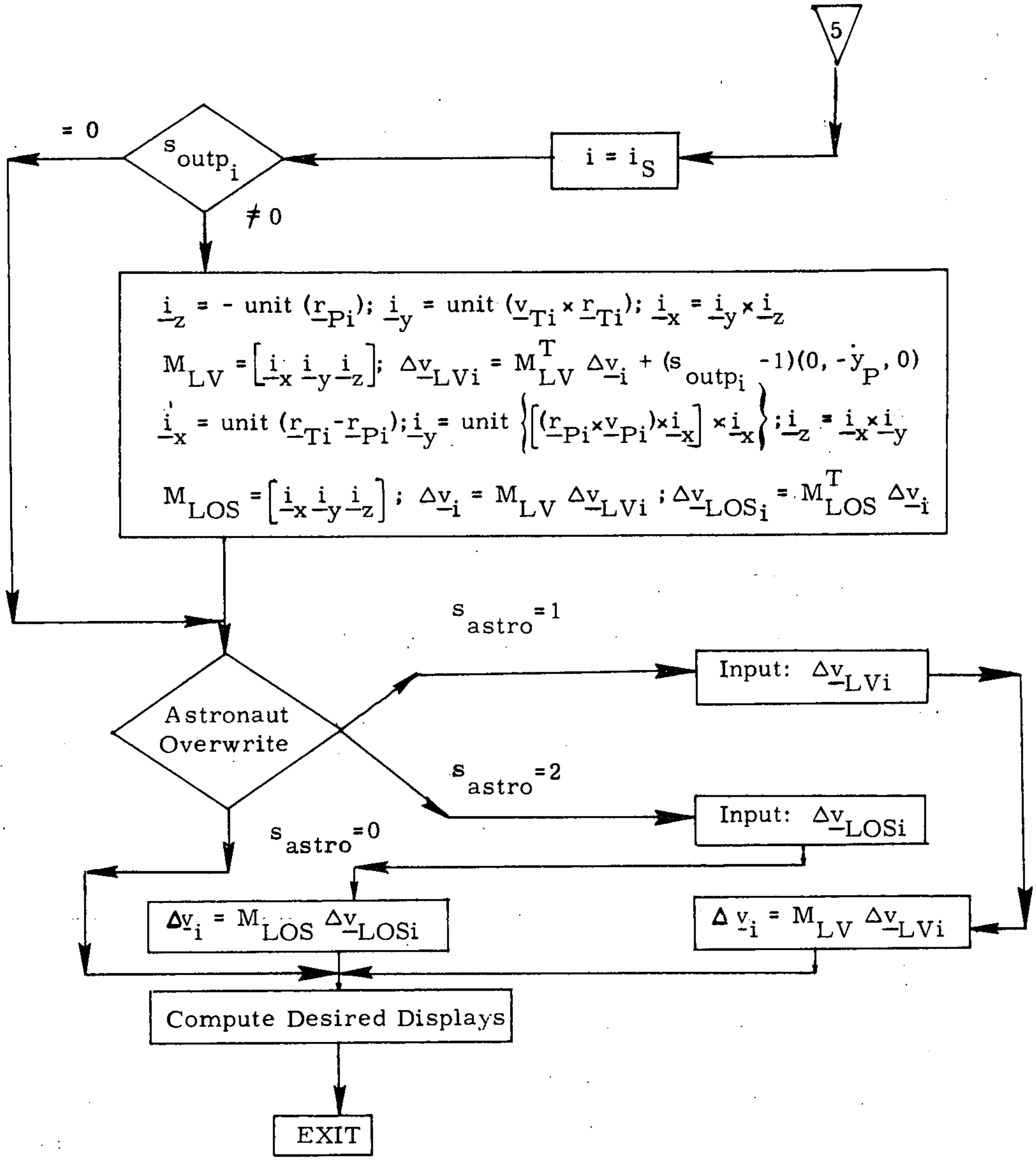

5
$i = i_S$
s_{outp_i}
= 0
≠ 0
$\underline{i}_z = -\,\text{unit}\,(\underline{r}_{Pi});\ \underline{i}_y = \text{unit}\,(\underline{v}_{Ti} \times \underline{r}_{Ti});\ \underline{i}_x = \underline{i}_y \times \underline{i}_z$
$M_{LV} = \left[\underline{i}_x\, \underline{i}_y\, \underline{i}_z\right];\ \Delta\underline{v}_{LVi} = M_{LV}^T\, \Delta\underline{v}_i + (s_{outp_i} - 1)(0, -\dot{y}_P, 0)$
$\underline{i}'_x = \text{unit}\,(\underline{r}_{Ti} - \underline{r}_{Pi});\ \underline{i}_y = \text{unit}\left\{\left[(\underline{r}_{Pi} \times \underline{v}_{Pi}) \times \underline{i}_x\right] \times \underline{i}_x\right\};\ \underline{i}_z = \underline{i}_x \times \underline{i}_y$
$M_{LOS} = \left[\underline{i}_x\, \underline{i}_y\, \underline{i}_z\right];\ \Delta\underline{v}_i = M_{LV}\, \Delta\underline{v}_{LVi};\ \Delta\underline{v}_{LOS_i} = M_{LOS}^T\, \Delta\underline{v}_i$
Astronaut Overwrite
$s_{astro} = 1$
$s_{astro} = 2$
$s_{astro} = 0$
Input: $\Delta\underline{v}_{LVi}$
Input: $\Delta\underline{v}_{LOSi}$
$\Delta\underline{v}_i = M_{LOS}\, \Delta\underline{v}_{LOSi}$
$\Delta\underline{v}_i = M_{LV}\, \Delta\underline{v}_{LVi}$
Compute Desired Displays
EXIT

Figure 6d. Main Program - Detailed Flow Diagram

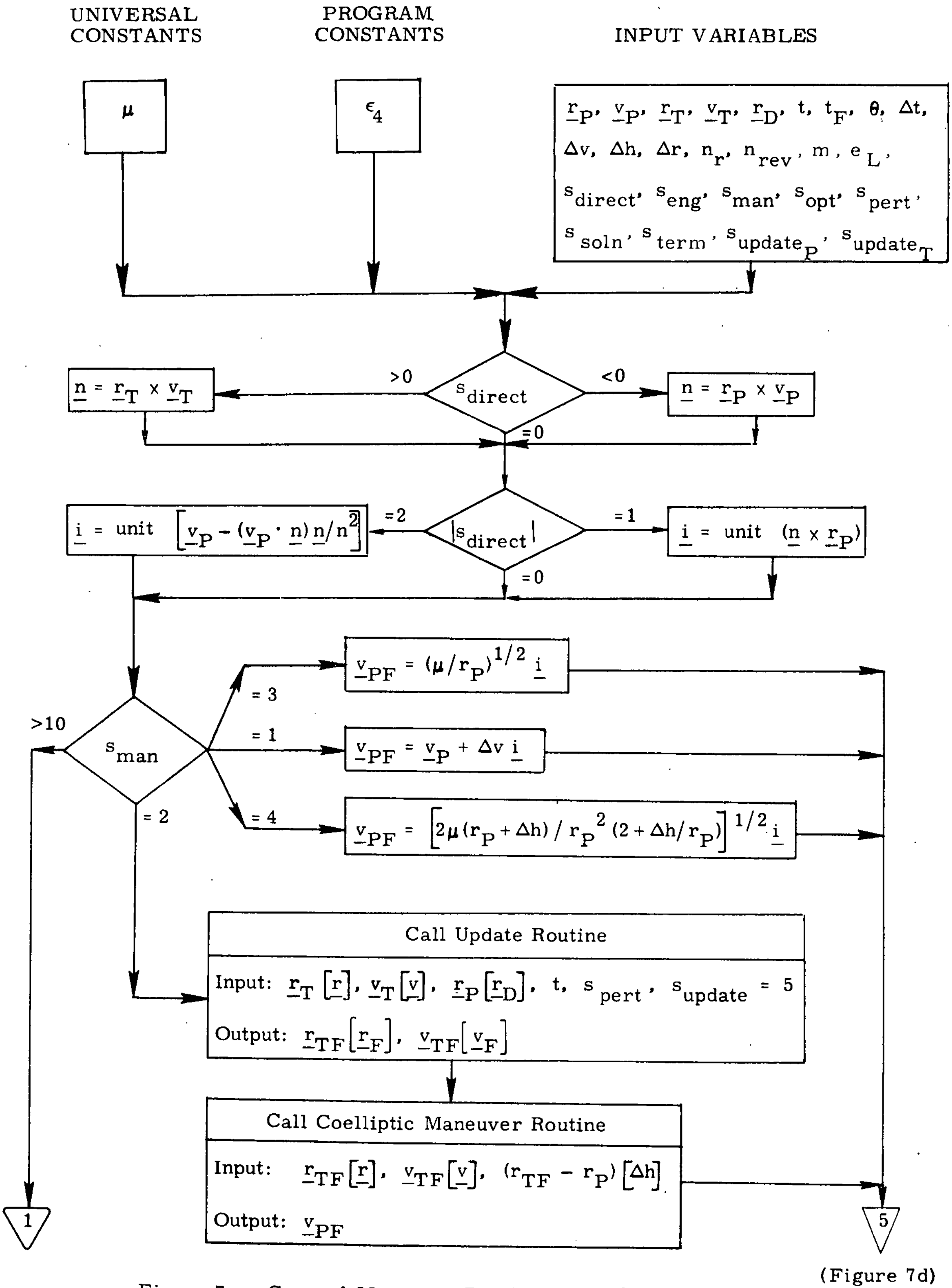

UNIVERSAL CONSTANTS
PROGRAM CONSTANTS
INPUT VARIABLES
μ
ϵ_4
$\underline{r}_P, \underline{v}_P, \underline{r}_T, \underline{v}_T, \underline{r}_D, t, t_F, \theta, \Delta t,$
$\Delta v, \Delta h, \Delta r, n_r, n_{rev}, m, e_L,$
$s_{direct}, s_{eng}, s_{man}, s_{opt}, s_{pert},$
$s_{soln}, s_{term}, s_{update_P}, s_{update_T}$
$\underline{n} = \underline{r}_T \times \underline{v}_T$
>0
s_{direct}
<0
$\underline{n} = \underline{r}_P \times \underline{v}_P$
=0
$\underline{i} = \text{unit} \left[\underline{v}_P - (\underline{v}_P \cdot \underline{n})\underline{n}/n^2\right]$
=2
$|s_{direct}|$
=1
$\underline{i} = \text{unit} \; (\underline{n} \times \underline{r}_P)$
=0
$\underline{v}_{PF} = (\mu/r_P)^{1/2} \; \underline{i}$
=3
>10
s_{man}
= 1
$\underline{v}_{PF} = \underline{v}_P + \Delta v \; \underline{i}$
= 2
= 4
$\underline{v}_{PF} = \left[2\mu(r_P + \Delta h) / r_P^{\,2} (2 + \Delta h/r_P)\right]^{1/2} \underline{i}$
Call Update Routine
Input: $\underline{r}_T \, [\underline{r}], \; \underline{v}_T \, [\underline{v}], \; \underline{r}_P \, [\underline{r}_D], \; t, \; s_{pert}, \; s_{update} = 5$
Output: $\underline{r}_{TF} \, [\underline{r}_F], \; \underline{v}_{TF} \, [\underline{v}_F]$
Call Coelliptic Maneuver Routine
Input: $\underline{r}_{TF} \, [\underline{r}], \; \underline{v}_{TF} \, [\underline{v}], \; (r_{TF} - r_P) \, [\Delta h]$
Output: $\underline{v}_{PF}$
1
5
(Figure 7d)

Figure 7a. General Maneuver Routine - Detailed Flow Diagram

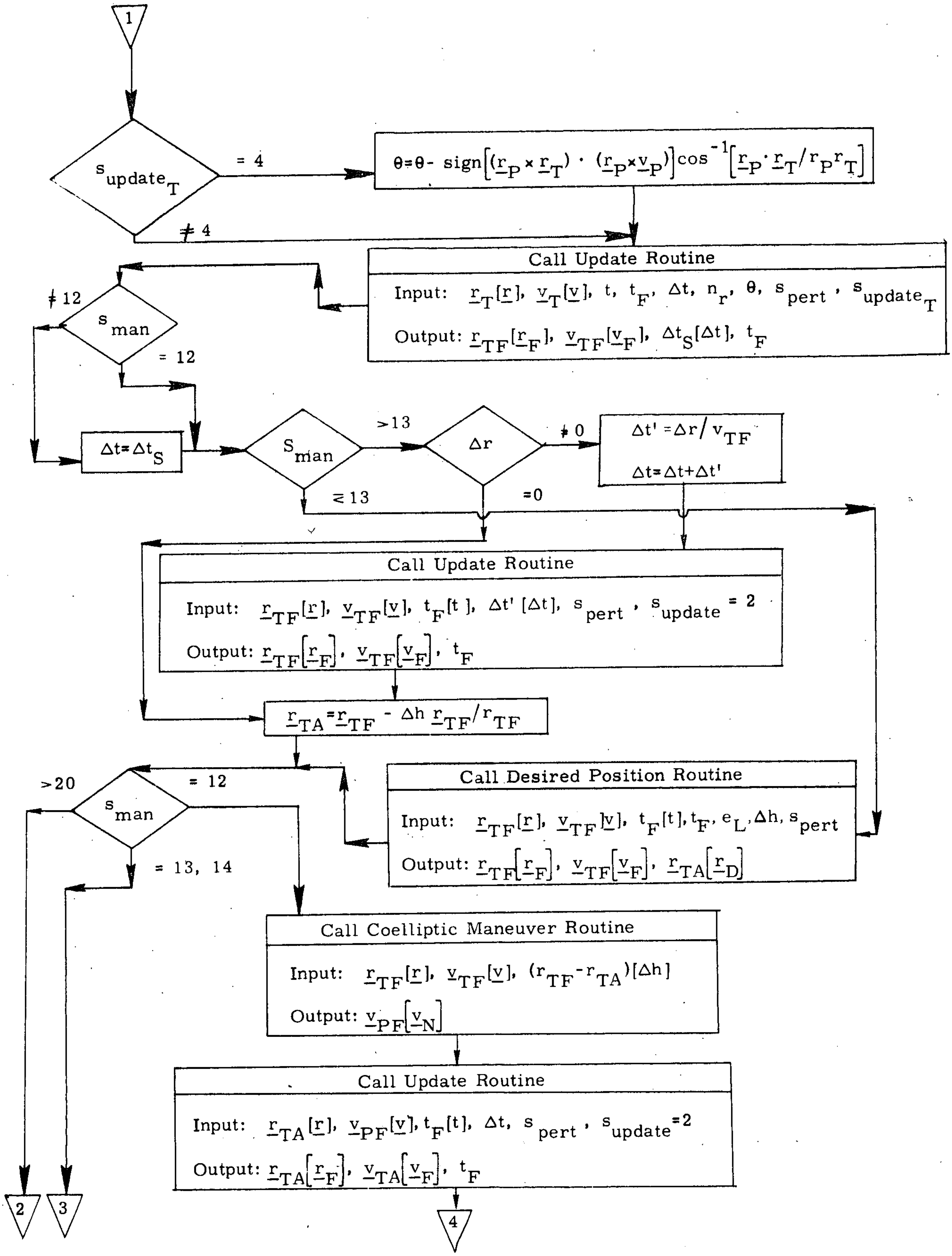

1
s_{update_T}
= 4
≠ 4
$\theta = \theta - \text{sign}[(\underline{r}_P \times \underline{r}_T) \cdot (\underline{r}_P \times \underline{v}_P)] \cos^{-1}[\underline{r}_P \cdot \underline{r}_T / r_P r_T]$
Call Update Routine
Input: $\underline{r}_T[\underline{r}]$, $\underline{v}_T[\underline{v}]$, t, t_F, Δt, n_r, θ, s_{pert}, s_{update_T}
Output: $\underline{r}_{TF}[\underline{r}_F]$, $\underline{v}_{TF}[\underline{v}_F]$, $\Delta t_S[\Delta t]$, t_F
≠ 12
s_{man}
= 12
$\Delta t = \Delta t_S$
S_{man}
>13
≤ 13
Δr
≠ 0
=0
$\Delta t' = \Delta r / v_{TF}$
$\Delta t = \Delta t + \Delta t'$
Call Update Routine
Input: $\underline{r}_{TF}[\underline{r}]$, $\underline{v}_{TF}[\underline{v}]$, $t_F[t]$, $\Delta t'[\Delta t]$, s_{pert}, $s_{update} = 2$
Output: $\underline{r}_{TF}[\underline{r}_F]$, $\underline{v}_{TF}[\underline{v}_F]$, t_F
$\underline{r}_{TA} = \underline{r}_{TF} - \Delta h\, \underline{r}_{TF} / r_{TF}$
>20
= 12
s_{man}
= 13, 14
Call Desired Position Routine
Input: $\underline{r}_{TF}[\underline{r}]$, $\underline{v}_{TF}[\underline{v}]$, $t_F[t]$, t_F, e_L, Δh, s_{pert}
Output: $\underline{r}_{TF}[\underline{r}_F]$, $\underline{v}_{TF}[\underline{v}_F]$, $\underline{r}_{TA}[\underline{r}_D]$
Call Coelliptic Maneuver Routine
Input: $\underline{r}_{TF}[\underline{r}]$, $\underline{v}_{TF}[\underline{v}]$, $(r_{TF} - r_{TA})[\Delta h]$
Output: $\underline{v}_{PF}[\underline{v}_N]$
Call Update Routine
Input: $\underline{r}_{TA}[\underline{r}]$, $\underline{v}_{PF}[\underline{v}]$, $t_F[t]$, Δt, s_{pert}, $s_{update} = 2$
Output: $\underline{r}_{TA}[\underline{r}_F]$, $\underline{v}_{TA}[\underline{v}_F]$, t_F
2
3
4

Figure 7b. General Maneuver Routine - Detailed Flow Diagram

9.8.1.1 Rendezvous Targeting (continued)

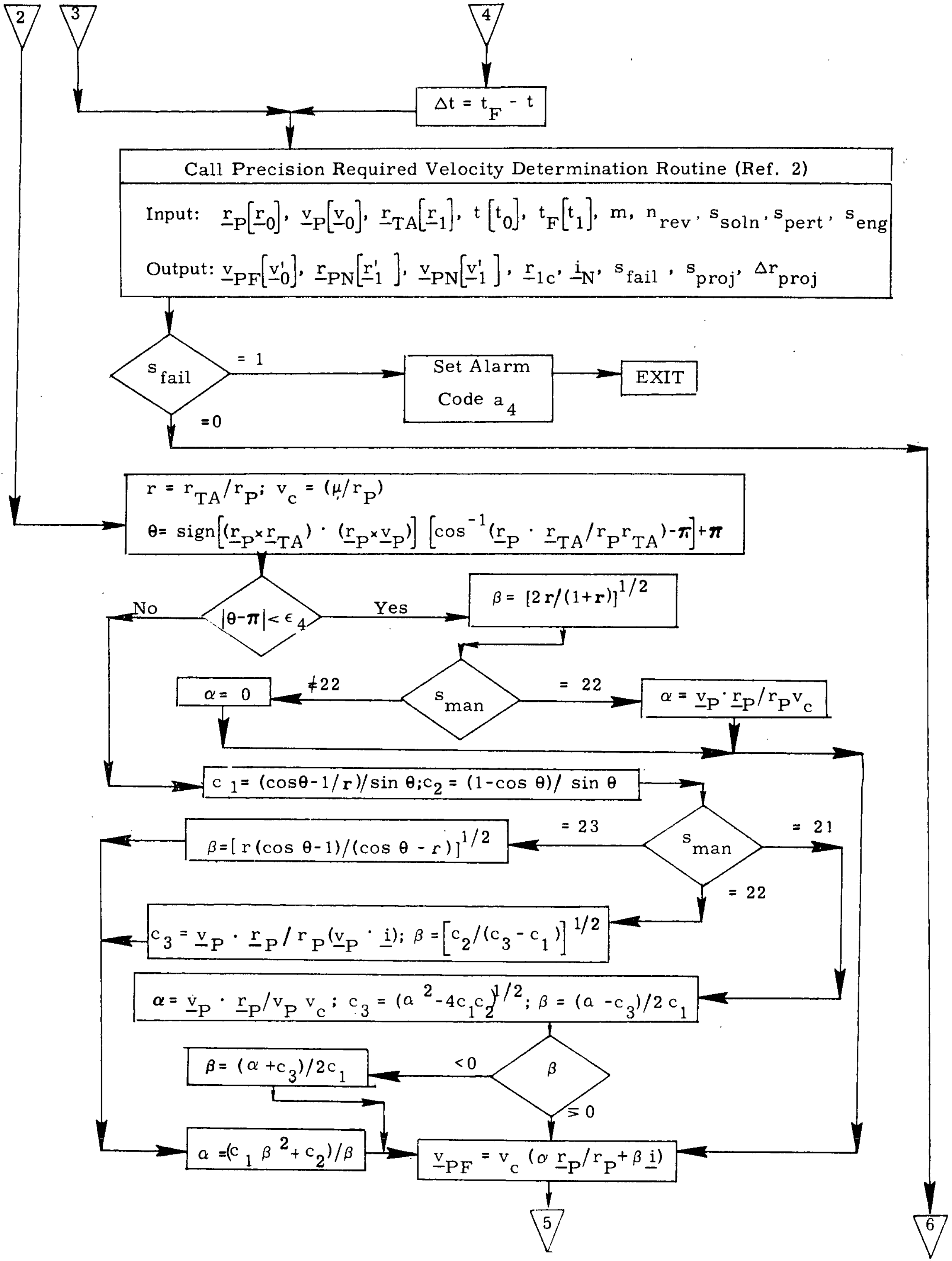

2
3
4
$\Delta t = t_F - t$
Call Precision Required Velocity Determination Routine (Ref. 2)
Input: $\underline{r}_P[\underline{r}_0]$, $\underline{v}_P[\underline{v}_0]$, $\underline{r}_{TA}[\underline{r}_1]$, $t\,[t_0]$, $t_F[t_1]$, m, n_{rev}, s_{soln}, s_{pert}, s_{eng}
Output: $\underline{v}_{PF}[\underline{v}'_0]$, $\underline{r}_{PN}[\underline{r}'_1]$, $\underline{v}_{PN}[\underline{v}'_1]$, $\underline{r}_{1c}$, $\underline{i}_N$, s_{fail}, s_{proj}, Δr_{proj}
s_{fail}
= 1
=0
Set Alarm Code a_4
EXIT
$r = r_{TA}/r_P$; $v_c = (\mu/r_P)$
$\theta = \text{sign}[(\underline{r}_P \times \underline{r}_{TA}) \cdot (\underline{r}_P \times \underline{v}_P)]\,[\cos^{-1}(\underline{r}_P \cdot \underline{r}_{TA}/r_P r_{TA}) - \pi] + \pi$
$|\theta - \pi| < \epsilon_4$
No
Yes
$\beta = [2r/(1+r)]^{1/2}$
s_{man}
≠22
= 22
$\alpha = 0$
$\alpha = \underline{v}_P \cdot \underline{r}_P / r_P v_c$
$c_1 = (\cos\theta - 1/r)/\sin\theta$; $c_2 = (1 - \cos\theta)/\sin\theta$
s_{man}
= 23
= 21
= 22
$\beta = [r(\cos\theta - 1)/(\cos\theta - r)]^{1/2}$
$c_3 = \underline{v}_P \cdot \underline{r}_P / r_P(\underline{v}_P \cdot \underline{i})$; $\beta = [c_2/(c_3 - c_1)]^{1/2}$
$\alpha = \underline{v}_P \cdot \underline{r}_P / v_P v_c$; $c_3 = (\alpha^2 - 4c_1c_2)^{1/2}$; $\beta = (\alpha - c_3)/2c_1$
β
<0
≥ 0
$\beta = (\alpha + c_3)/2c_1$
$\alpha = (c_1\beta^2 + c_2)/\beta$
$\underline{v}_{PF} = v_c(\alpha\,\underline{r}_P/r_P + \beta\,\underline{i})$
5
6

Figure 7c. General Maneuver Routine - Detailed Flow Diagram

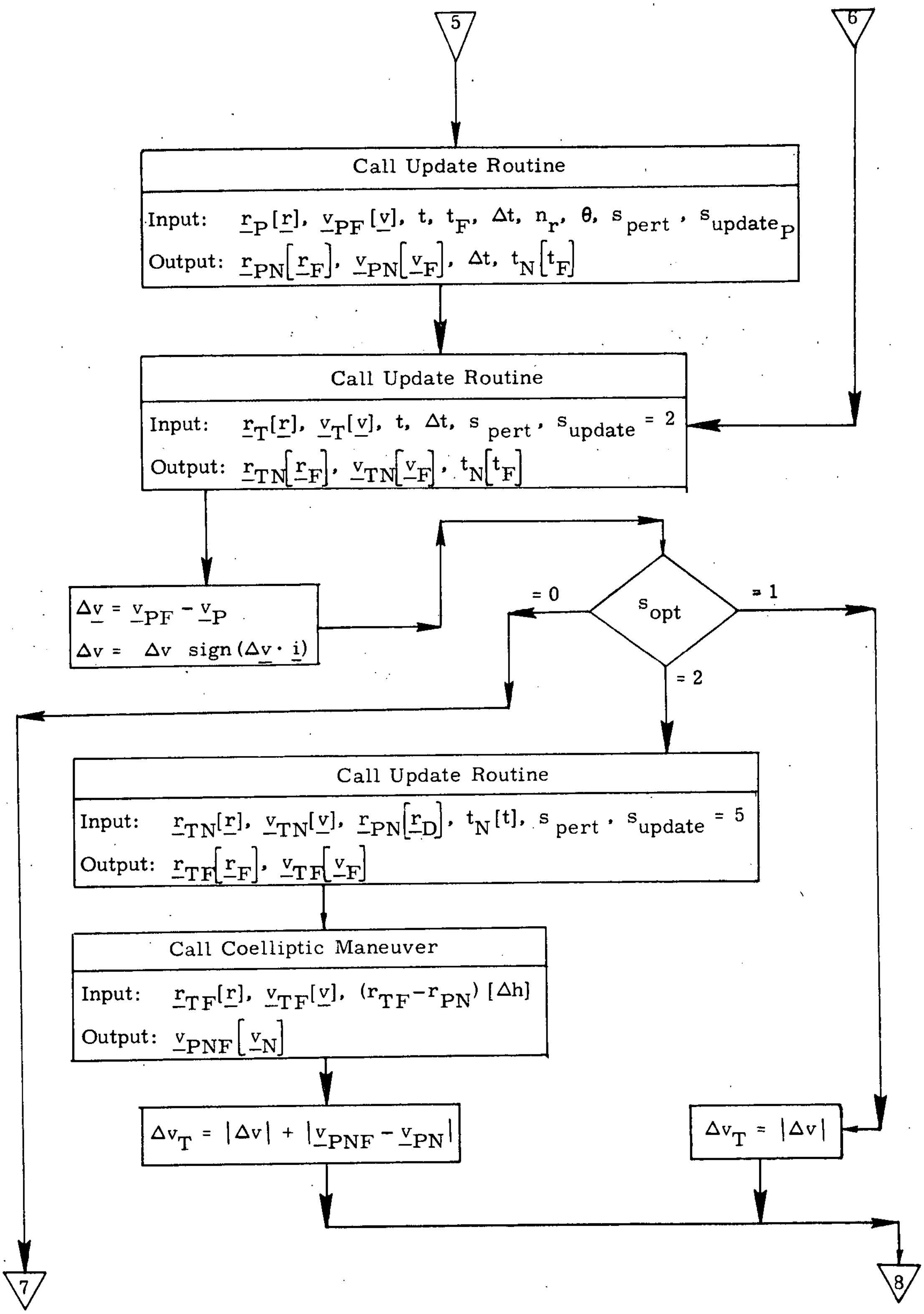

5
6
Call Update Routine
Input: $\underline{r}_P[\underline{r}]$, $\underline{v}_{PF}[\underline{v}]$, t, t_F, Δt, n_r, θ, s_{pert}, s_{update_P}
Output: $\underline{r}_{PN}[\underline{r}_F]$, $\underline{v}_{PN}[\underline{v}_F]$, Δt, $t_N[t_F]$
Call Update Routine
Input: $\underline{r}_T[\underline{r}]$, $\underline{v}_T[\underline{v}]$, t, Δt, s_{pert}, s_{update} = 2
Output: $\underline{r}_{TN}[\underline{r}_F]$, $\underline{v}_{TN}[\underline{v}_F]$, $t_N[t_F]$
$\Delta\underline{v} = \underline{v}_{PF} - \underline{v}_P$
$\Delta v = \Delta v \ \mathrm{sign}(\Delta\underline{v} \cdot \underline{i})$
s_{opt}
= 0
= 1
= 2
Call Update Routine
Input: $\underline{r}_{TN}[\underline{r}]$, $\underline{v}_{TN}[\underline{v}]$, $\underline{r}_{PN}[\underline{r}_D]$, $t_N[t]$, s_{pert}, s_{update} = 5
Output: $\underline{r}_{TF}[\underline{r}_F]$, $\underline{v}_{TF}[\underline{v}_F]$
Call Coelliptic Maneuver
Input: $\underline{r}_{TF}[\underline{r}]$, $\underline{v}_{TF}[\underline{v}]$, $(r_{TF} - r_{PN})$ [Δh]
Output: $\underline{v}_{PNF}[\underline{v}_N]$
$\Delta v_T = |\Delta v| + |\underline{v}_{PNF} - \underline{v}_{PN}|$
$\Delta v_T = |\Delta v|$
7
8

Figure 7d. General Maneuver Routine - Detailed Flow Diagram

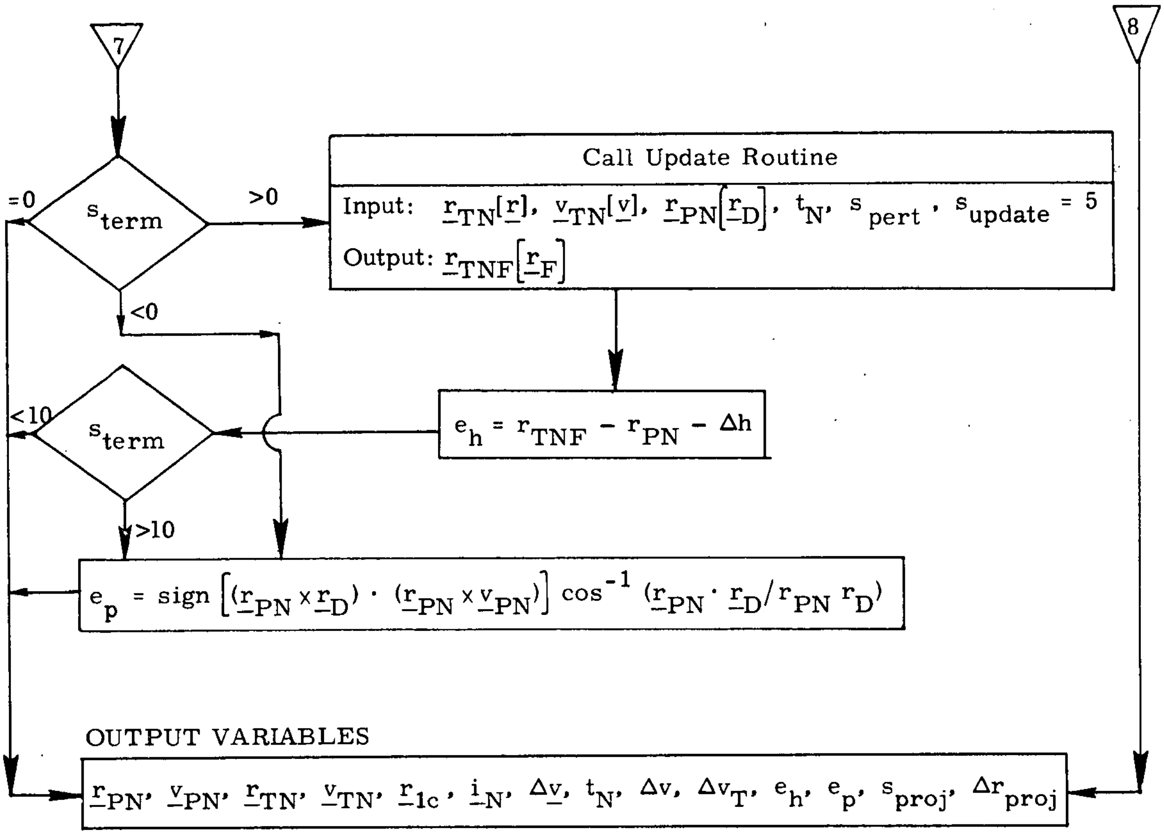

7
8
s_{term}
=0
>0
<0
Call Update Routine
Input: $\underline{r}_{TN}[\underline{r}]$, $\underline{v}_{TN}[\underline{v}]$, $\underline{r}_{PN}[\underline{r}_D]$, t_N, s_{pert}, $s_{update} = 5$
Output: $\underline{r}_{TNF}[\underline{r}_F]$
$e_h = r_{TNF} - r_{PN} - \Delta h$
s_{term}
<10
>10
$e_p = \text{sign}\left[(\underline{r}_{PN} \times \underline{r}_D) \cdot (\underline{r}_{PN} \times \underline{v}_{PN})\right] \cos^{-1} (\underline{r}_{PN} \cdot \underline{r}_D / r_{PN}\, r_D)$
OUTPUT VARIABLES
$\underline{r}_{PN}$, $\underline{v}_{PN}$, $\underline{r}_{TN}$, $\underline{v}_{TN}$, $\underline{r}_{1c}$, $\underline{i}_N$, $\Delta\underline{v}$, t_N, Δv, Δv_T, e_h, e_p, s_{proj}, Δr_{proj}

Figure 7e. General Maneuver Routine - Detailed Flow Diagram

9.8.1.1 Rendezvous Targeting (continued)

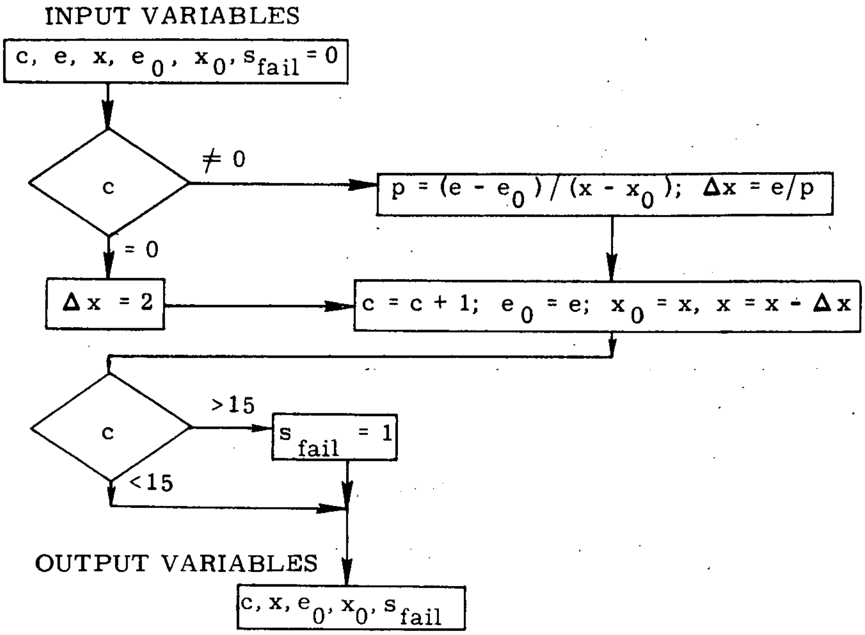

INPUT VARIABLES
$c, e, x, e_0, x_0, s_{fail} = 0$
c
$\neq 0$
$p = (e - e_0)/(x - x_0);\ \Delta x = e/p$
$= 0$
$\Delta x = 2$
$c = c + 1;\ e_0 = e;\ x_0 = x,\ x = x - \Delta x$
c
>15
$s_{fail} = 1$
<15
OUTPUT VARIABLES
c, x, e_0, x_0, s_{fail}

Figure 8. Iteration Routine - Detailed Flow Diagram

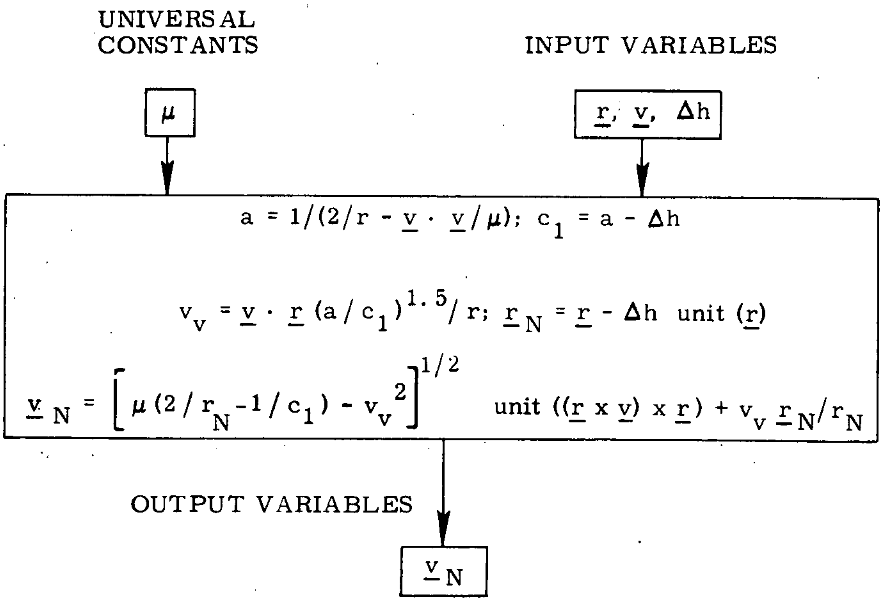

UNIVERSAL CONSTANTS
INPUT VARIABLES
μ
$\underline{r}, \underline{v}, \Delta h$
$a = 1/(2/r - \underline{v} \cdot \underline{v}/\mu);\ c_1 = a - \Delta h$
$v_v = \underline{v} \cdot \underline{r}\,(a/c_1)^{1.5}/r;\ \underline{r}_N = \underline{r} - \Delta h \text{ unit } (\underline{r})$
$\underline{v}_N = \left[\mu(2/r_N - 1/c_1) - v_v^2\right]^{1/2} \text{ unit } ((\underline{r} \times \underline{v}) \times \underline{r}) + v_v\, \underline{r}_N/r_N$
OUTPUT VARIABLES
$\underline{v}_N$

Figure 9. Coelliptic Maneuver Routine - Detailed Flow Diagram

9.8.1.1 Rendezvous Targeting (continued)

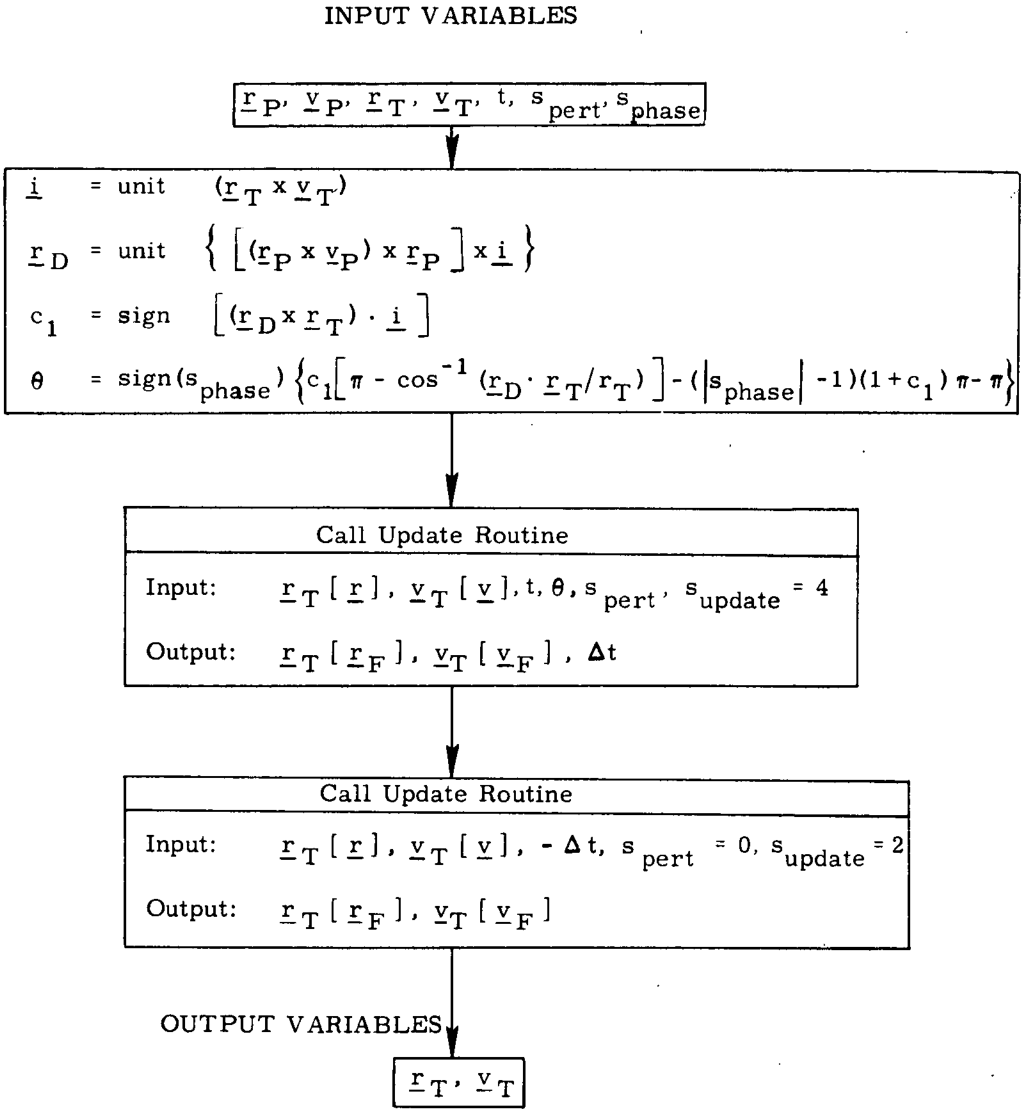

INPUT VARIABLES
$\underline{r}_P, \underline{v}_P, \underline{r}_T, \underline{v}_T, t, s_{pert}, s_{phase}$
$\underline{i} = \text{unit}\ (\underline{r}_T \times \underline{v}_T)$
$\underline{r}_D = \text{unit}\ \left\{ \left[(\underline{r}_P \times \underline{v}_P) \times \underline{r}_P \right] \times \underline{i} \right\}$
$c_1 = \text{sign}\ \left[(\underline{r}_D \times \underline{r}_T) \cdot \underline{i} \right]$
$\theta = \text{sign}(s_{phase}) \left\{ c_1 \left[\pi - \cos^{-1} (\underline{r}_D \cdot \underline{r}_T / r_T) \right] - (|s_{phase}| - 1)(1 + c_1)\pi - \pi \right\}$
Call Update Routine
Input: $\underline{r}_T [\underline{r}], \underline{v}_T [\underline{v}], t, \theta, s_{pert}, s_{update} = 4$
Output: $\underline{r}_T [\underline{r}_F], \underline{v}_T [\underline{v}_F], \Delta t$
Call Update Routine
Input: $\underline{r}_T [\underline{r}], \underline{v}_T [\underline{v}], -\Delta t, s_{pert} = 0, s_{update} = 2$
Output: $\underline{r}_T [\underline{r}_F], \underline{v}_T [\underline{v}_F]$
OUTPUT VARIABLES
$\underline{r}_T, \underline{v}_T$

Figure 10. Phase Match Routine - Detailed Flow Diagram

9.8.1.1 Rendezvous Targeting (continued)

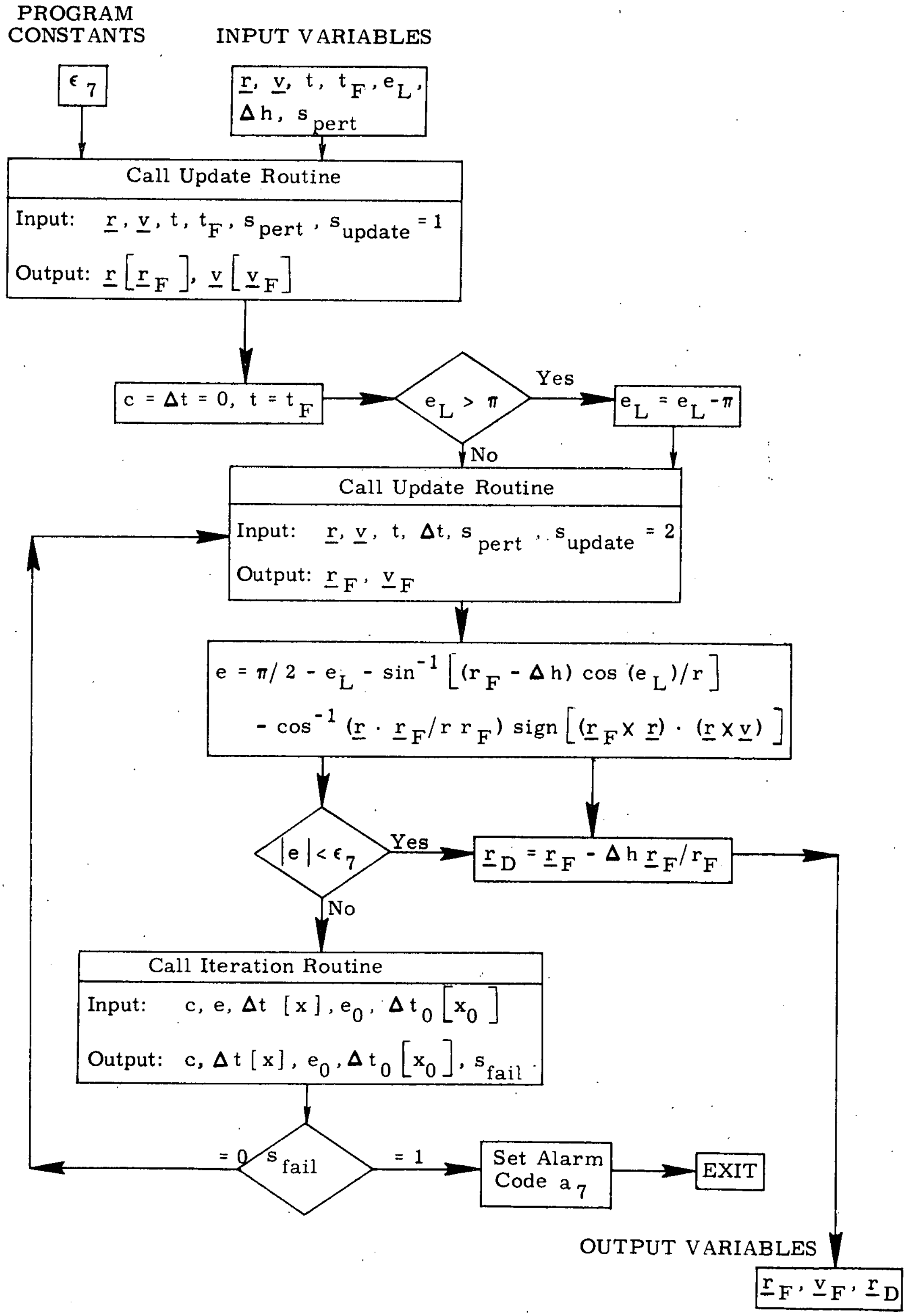

PROGRAM CONSTANTS
INPUT VARIABLES
ϵ_7
$\underline{r}, \underline{v}, t, t_F, e_L, \Delta h, s_{pert}$
Call Update Routine
Input: $\underline{r}, \underline{v}, t, t_F, s_{pert}, s_{update} = 1$
Output: $\underline{r}\left[\underline{r}_F\right], \underline{v}\left[\underline{v}_F\right]$
$c = \Delta t = 0, \ t = t_F$
$e_L > \pi$
Yes
$e_L = e_L - \pi$
No
Call Update Routine
Input: $\underline{r}, \underline{v}, t, \Delta t, s_{pert}, s_{update} = 2$
Output: $\underline{r}_F, \underline{v}_F$
$e = \pi/2 - e_L - \sin^{-1}\left[(r_F - \Delta h)\cos(e_L)/r\right] - \cos^{-1}(\underline{r} \cdot \underline{r}_F / r\, r_F)\ \mathrm{sign}\left[(\underline{r}_F \times \underline{r}) \cdot (\underline{r} \times \underline{v})\right]$
$|e| < \epsilon_7$
Yes
$\underline{r}_D = \underline{r}_F - \Delta h\, \underline{r}_F / r_F$
No
Call Iteration Routine
Input: $c, e, \Delta t\ [x], e_0, \Delta t_0 \left[x_0\right]$
Output: $c, \Delta t\ [x], e_0, \Delta t_0 \left[x_0\right], s_{fail}$
s_{fail}
$= 0$
$= 1$
Set Alarm Code a_7
EXIT
OUTPUT VARIABLES
$\underline{r}_F, \underline{v}_F, \underline{r}_D$

Figure 11. Desired Position Routine - Detailed Flow Diagram

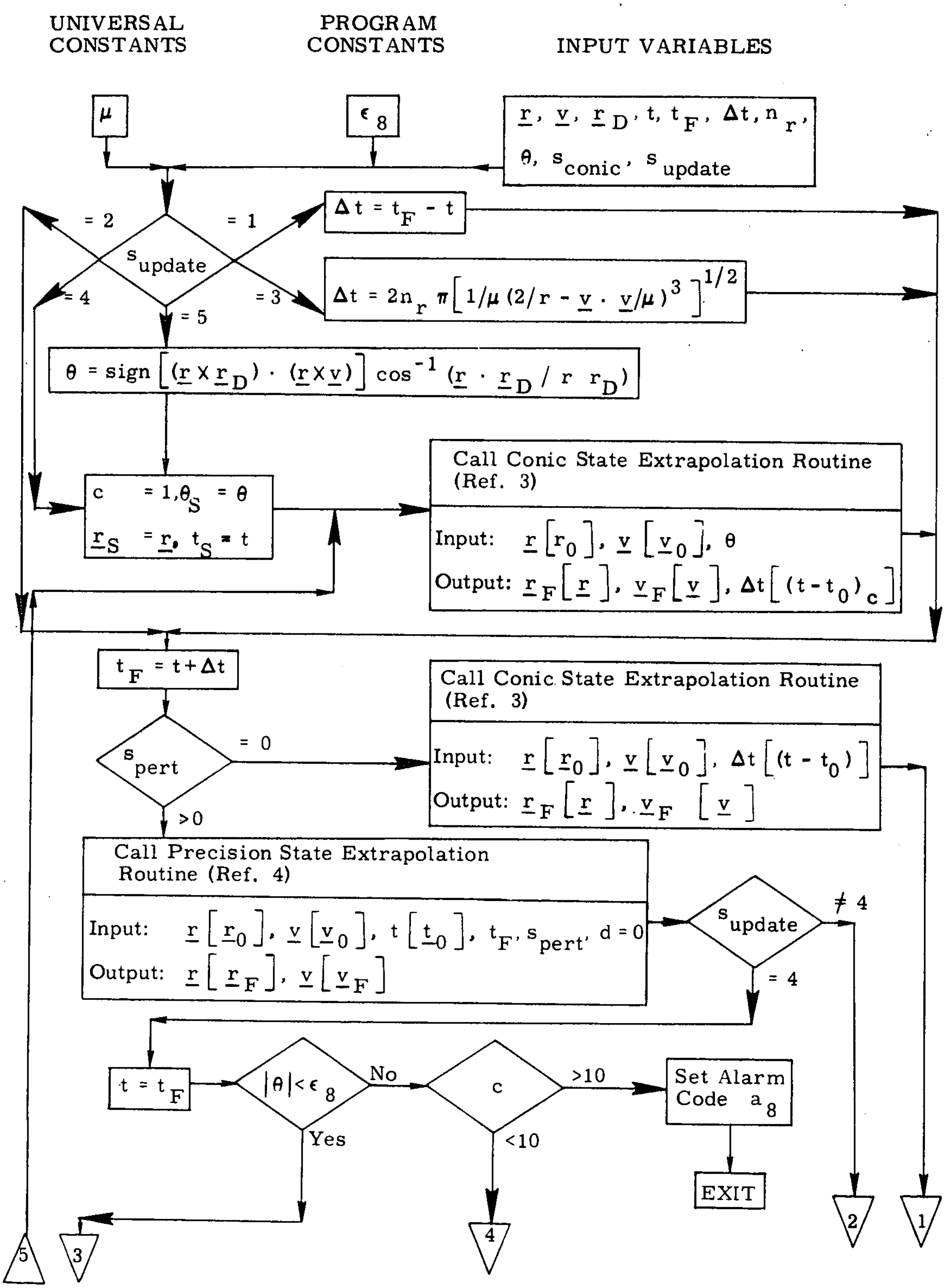

UNIVERSAL CONSTANTS
PROGRAM CONSTANTS
INPUT VARIABLES
μ
ϵ_8
$\underline{r}, \underline{v}, \underline{r}_D, t, t_F, \Delta t, n_r, \theta, s_{conic}, s_{update}$
s_{update}
= 1
= 2
= 3
= 4
= 5
$\Delta t = t_F - t$
$\Delta t = 2n_r \pi [1/\mu (2/r - \underline{v} \cdot \underline{v}/\mu)^3]^{1/2}$
$\theta = \text{sign}[(\underline{r} \times \underline{r}_D) \cdot (\underline{r} \times \underline{v})] \cos^{-1} (\underline{r} \cdot \underline{r}_D / r\ r_D)$
$c = 1, \theta_S = \theta$
$\underline{r}_S = \underline{r}, t_S = t$
Call Conic State Extrapolation Routine (Ref. 3)
Input: $\underline{r} [\underline{r}_0], \underline{v} [\underline{v}_0], \theta$
Output: $\underline{r}_F [\underline{r}], \underline{v}_F [\underline{v}], \Delta t [(t - t_0)_c]$
$t_F = t + \Delta t$
s_{pert}
= 0
>0
Call Conic State Extrapolation Routine (Ref. 3)
Input: $\underline{r} [\underline{r}_0], \underline{v} [\underline{v}_0], \Delta t [(t - t_0)]$
Output: $\underline{r}_F [\underline{r}], \underline{v}_F [\underline{v}]$
Call Precision State Extrapolation Routine (Ref. 4)
Input: $\underline{r} [\underline{r}_0], \underline{v} [\underline{v}_0], t [\underline{t}_0], t_F, s_{pert}, d = 0$
Output: $\underline{r} [\underline{r}_F], \underline{v} [\underline{v}_F]$
s_{update}
≠ 4
= 4
$t = t_F$
$|\theta| < \epsilon_8$
No
Yes
c
>10
<10
Set Alarm Code a_8
EXIT
5
3
4
2
1

Figure 12a. Update Routine - Detailed Flow Diagram

9.8.1.1 Rendezvous Targeting (continued)

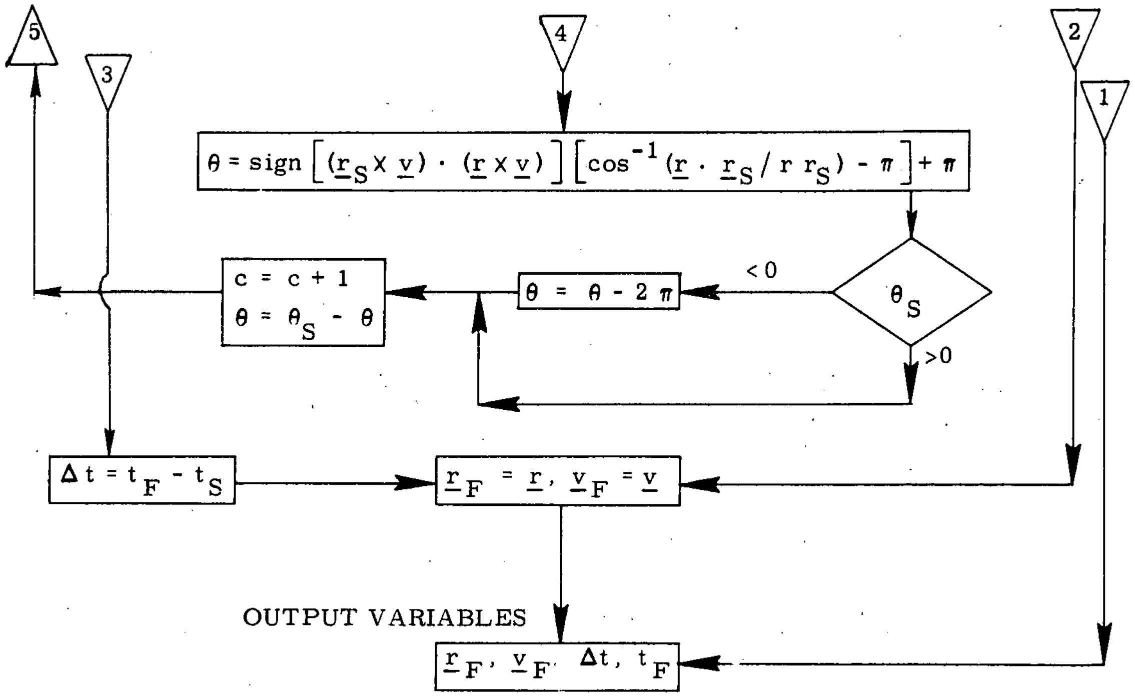

5
3
4
2
1
$\theta = \text{sign}\left[(\underline{r}_S \times \underline{v}) \cdot (\underline{r} \times \underline{v})\right]\left[\cos^{-1}(\underline{r} \cdot \underline{r}_S / r\, r_S) - \pi\right] + \pi$
θ_S
< 0
> 0
$\theta = \theta - 2\pi$
$c = c + 1$
$\theta = \theta_S - \theta$
$\Delta t = t_F - t_S$
$\underline{r}_F = \underline{r},\ \underline{v}_F = \underline{v}$
OUTPUT VARIABLES
$\underline{r}_F,\ \underline{v}_F\ \ \Delta t,\ t_F$

Figure 12b. Update Routine - Detailed Flow Diagram

9.8.1.1 Rendezvous Targeting (continued)

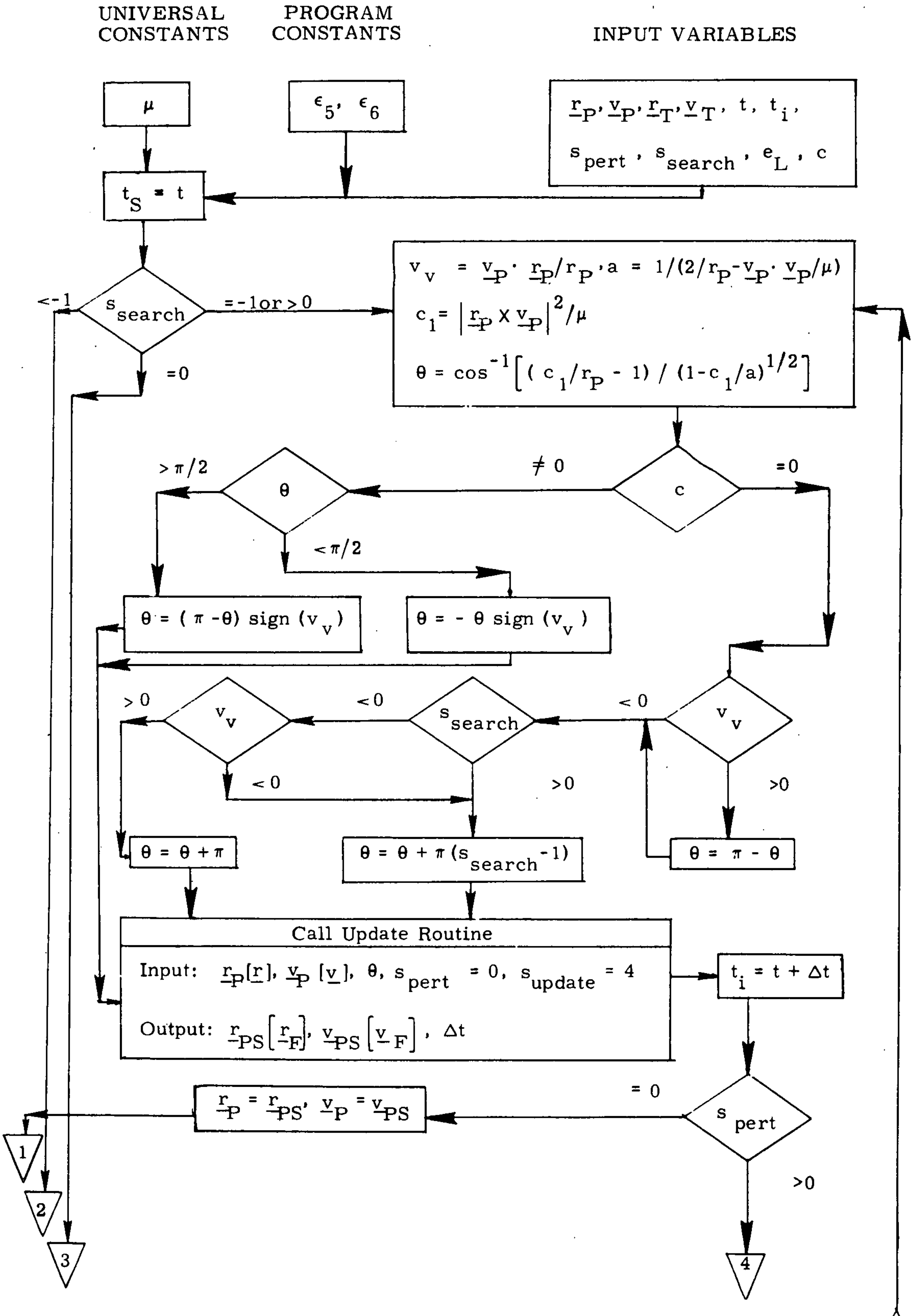
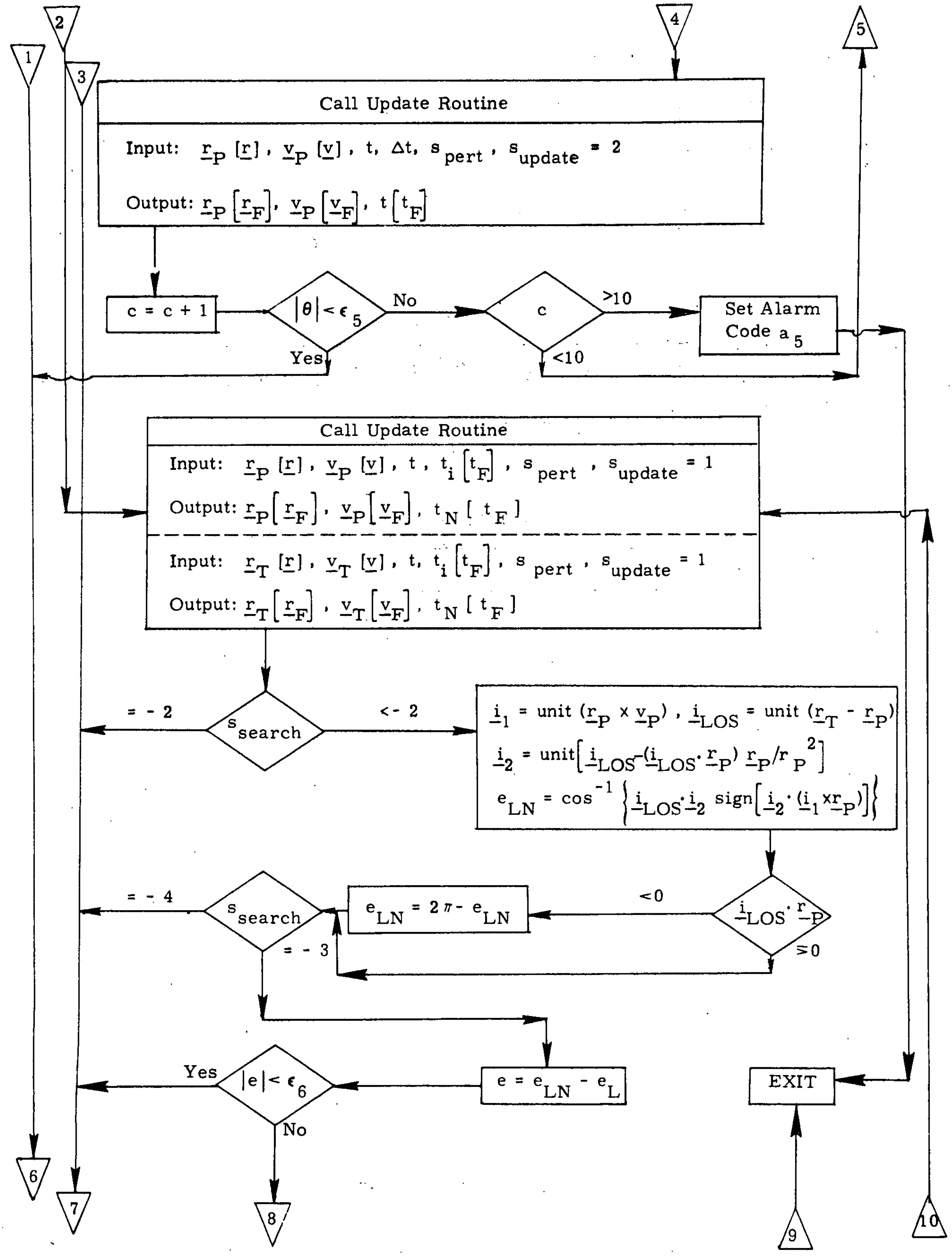

UNIVERSAL CONSTANTS
PROGRAM CONSTANTS
INPUT VARIABLES
μ
ϵ_5, ϵ_6
$\underline{r}_P, \underline{v}_P, \underline{r}_T, \underline{v}_T, t, t_i,$
$s_{pert}, s_{search}, e_L, c$
$t_S = t$
s_{search}
< -1
$= -1 \text{ or } > 0$
$= 0$
$v_V = \underline{v}_P \cdot \underline{r}_P / r_P, a = 1/(2/r_P - \underline{v}_P \cdot \underline{v}_P/\mu)$
$c_1 = |\underline{r}_P \times \underline{v}_P|^2/\mu$
$\theta = \cos^{-1}[(c_1/r_P - 1)/(1 - c_1/a)^{1/2}]$
c
$\neq 0$
$= 0$
θ
$> \pi/2$
$< \pi/2$
$\theta = (\pi - \theta) \text{ sign } (v_V)$
$\theta = -\theta \text{ sign } (v_V)$
v_V
> 0
< 0
s_{search}
< 0
> 0
v_V
< 0
> 0
$\theta = \theta + \pi$
$\theta = \theta + \pi(s_{search} - 1)$
$\theta = \pi - \theta$
Call Update Routine
Input: $\underline{r}_P[\underline{r}], \underline{v}_P[\underline{v}], \theta, s_{pert} = 0, s_{update} = 4$
Output: $\underline{r}_{PS}[\underline{r}_F], \underline{v}_{PS}[\underline{v}_F], \Delta t$
$t_i = t + \Delta t$
s_{pert}
$= 0$
> 0
$\underline{r}_P = \underline{r}_{PS}, \underline{v}_P = \underline{v}_{PS}$
1
2
3
4
5

Figure 13a. Search Routine - Detailed Flow Diagram

9.8.1.1 Rendezvous Targeting (continued)



1
2
3
4
5
Call Update Routine
Input: $\underline{r}_P\ [\underline{r}]$, $\underline{v}_P\ [\underline{v}]$, t, Δt, s_{pert}, s_{update} = 2
Output: $\underline{r}_P\left[\underline{r}_F\right]$, $\underline{v}_P\left[\underline{v}_F\right]$, $t\left[t_F\right]$
c = c + 1
$|\theta| < \epsilon_5$
No
Yes
c
>10
<10
Set Alarm Code a_5
Call Update Routine
Input: $\underline{r}_P\ [\underline{r}]$, $\underline{v}_P\ [\underline{v}]$, t, $t_i\left[t_F\right]$, s_{pert}, s_{update} = 1
Output: $\underline{r}_P\left[\underline{r}_F\right]$, $\underline{v}_P\left[\underline{v}_F\right]$, $t_N\ [\ t_F\ ]$
Input: $\underline{r}_T\ [\underline{r}]$, $\underline{v}_T\ [\underline{v}]$, t, $t_i\left[t_F\right]$, s_{pert}, s_{update} = 1
Output: $\underline{r}_T\left[\underline{r}_F\right]$, $\underline{v}_T\left[\underline{v}_F\right]$, $t_N\ [\ t_F\ ]$
= - 2
s_{search}
<- 2
$\underline{i}_1$ = unit $(\underline{r}_P \times \underline{v}_P)$, $\underline{i}_{LOS}$ = unit $(\underline{r}_T - \underline{r}_P)$
$\underline{i}_2$ = unit$\left[\underline{i}_{LOS} - (\underline{i}_{LOS} \cdot \underline{r}_P)\ \underline{r}_P / r_P^{\ 2}\right]$
$e_{LN} = \cos^{-1}\left\{\underline{i}_{LOS} \cdot \underline{i}_2\ \text{sign}\left[\underline{i}_2 \cdot (\underline{i}_1 \times \underline{r}_P)\right]\right\}$
= - 4
s_{search}
= - 3
$e_{LN} = 2\pi - e_{LN}$
<0
$\underline{i}_{LOS} \cdot \underline{r}_P$
≥0
Yes
$|e| < \epsilon_6$
No
$e = e_{LN} - e_L$
EXIT
6
7
8
9
10

Figure 13b. Search Routine - Detailed Flow Diagram

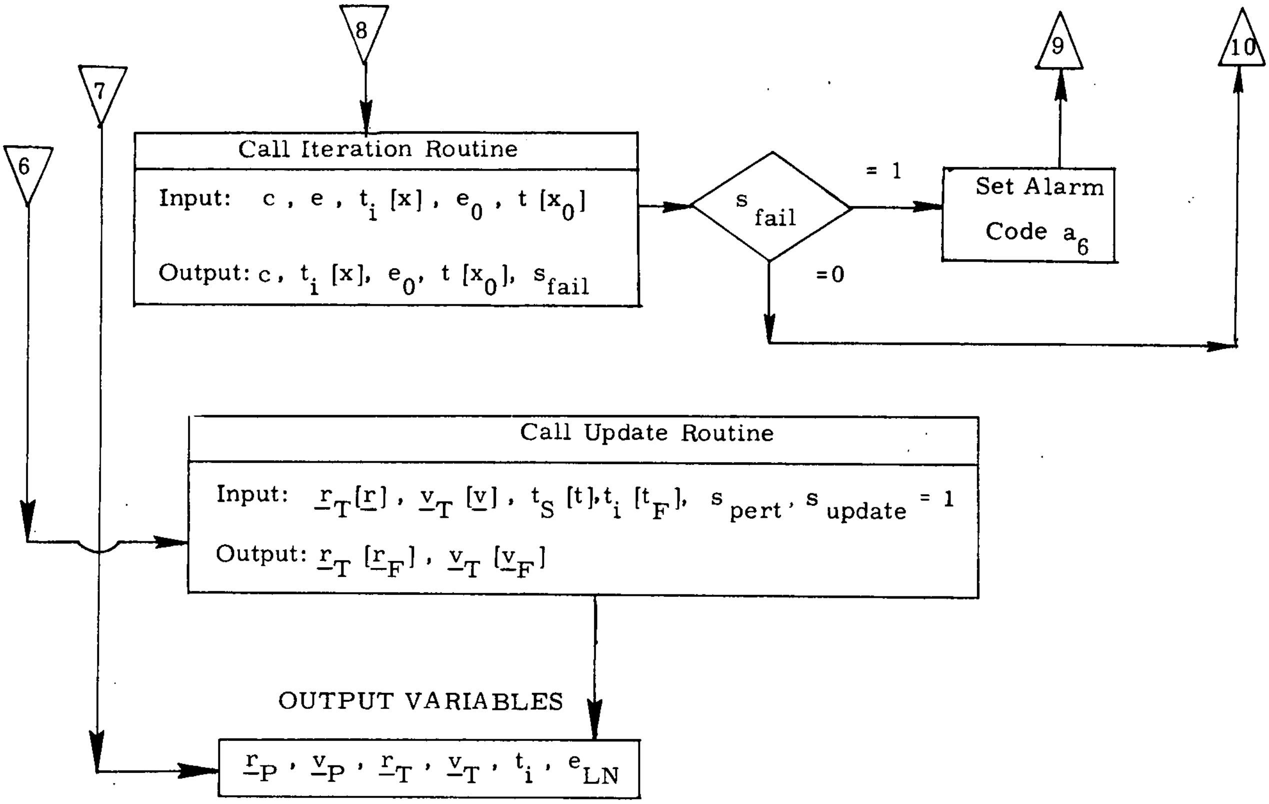

8
7
6
9
10
Call Iteration Routine
Input: c , e , t_i $[x]$, e_0 , t $[x_0]$
Output: c , t_i $[x]$, e_0, t $[x_0]$, s_{fail}
s_{fail}
= 1
=0
Set Alarm Code a_6
Call Update Routine
Input: $\underline{r}_T[\underline{r}]$, $\underline{v}_T$ $[\underline{v}]$, t_S $[t]$, t_i $[t_F]$, s_{pert}, s_{update} = 1
Output: $\underline{r}_T$ $[\underline{r}_F]$, $\underline{v}_T$ $[\underline{v}_F]$
OUTPUT VARIABLES
$\underline{r}_P$, $\underline{v}_P$, $\underline{r}_T$, $\underline{v}_T$, t_i , e_{LN}

Figure 13c. Search Routine - Detailed Flow Diagram

6. SUPPLEMENTARY INFORMATION

The Orbiter rendezvous targeting program proposed herein uses the basic targeting philosophy employed in Apollo and Skylab. (See Ref. 5 and 6.) The Orbiter program represents, to some degree, a general solution to the rendezvous targeting problem. The program has the capability of solving the Apollo and Skylab rendezvous configurations as well as many other configurations that can be determined by specifying sets of secondary constraints. The constraints contained in this document may require modification as experience is gained in generating sets of secondary constraints based on a variety of Orbiter missions. Reference 7 contains a more detailed discussion of the nature and number of the secondary constraints employed in the maneuver sequences.

The targeting program logic has been verified by a computer simulation that disclosed no iterative convergence problems for the range of trajectories considered. Reference 8 contains a simulation of a Skylab five-maneuver rendezvous configuration using the Orbiter targeting program. This program is currently being integrated with the Orbiter navigation programs to verify their combined effectiveness on a number of Orbiter missions.

Some of the maneuvers described in this document were based on conic orbit assumptions although the conic/precision switch was set to indicate that oblateness should be considered. One example is the computation of the "circular orbit" maneuver based on the conic circular velocity vector. This calculation can be modified to provide a maneuver that would result in a minimum altitude change orbit in an oblate gravity field. (See Ref. 9.)

REFERENCES

1. Pu, C., Higgins, J., Brand, T., "Powered Flight Guidance," Space Shuttle GN & C Equation Document No. 11, Rev. 1, M. I. T. / DL, September 1971.

2. Brand, T., "Precision Required Velocity Determination," Space Shuttle GN & C Equation Document No. 13, Rev. 1, M.I.T. /DL, September 1971.

3. Robertson, W., "Conic Required Velocity Determination," Space Shuttle GN & C Equation Document No. 10, Rev. 1, M. I. T. / DL, September 1971.

4. Robertson, W., "Precision State and Filter Weighting Matrix Extrapolation," Space Shuttle GN & C Equation Document No. 4, Rev. 1, M.I.T. /DL, October 1971.

5. "Guidance System Operations Plan for Manned CM Earth Orbital and Lunar Missions Using Program Colossus," R-577, Section 5 Guidance Equations (Rev. 14), M.I.T. /DL, March 1971.

6. "Guidance System Operations Plan for Manned CSM Earth Orbital Missions Using Program Skylark I," R-693, Section 5 Guidance Equations, M.I.T. /DL, October 1971.

7. Tempelman, W., "The Secondary Constraints Employed in the Orbiter Rendezvous Targeting Program", STS Memo No. 52-71, M. I. T. /DL, October 15, 1971.

8. Tempelman, W., "A Skylab Rendezvous Trajectory Computed with the Orbiter Targeting Program," STS Memo No. 51-71, M. I. T. /DL, October 12, 1971.

9. Geyling, F. and Westerman, H., Introduction to Orbital Mechanics (Addison Wesley Co., Reading, Mass., 1971), p 212.

SPACE SHUTTLE

GN&C SOFTWARE EQUATION SUBMITTAL

Software Equation Section Rendezvous Braking Submittal No. 41

Function: Automatically bring Orbiter within desired station-keeping boundries relative to target vehicle.

Module No. OG3 Function No. 7 (MSC 03690 Rev. A)

Submitted by: P. . Kachmar Co. MIT No. 15

Date: 23 November 1971

NASA Contract: I. Caro Organization EG2

Approved by Panel III K.T. Cox Date 11/29/71

Summary Description: The program includes navigation, targeting and guidance functions. Line-of-sight corrections, braking corrections and filtering of rendezvous measurement sensor data to improve vehicle and target state estimates are performed in a sequential manner. Provision is made to either use Relative State Updating routine (Sec. 9.8.2) or raw data and to use (Apollo-type) line-of-sight braking or Lambert targeting pending the result of on-going studies.

Comments:

(Design Status)

(Verification Status)

Panel Comments:

9.8.1.2 Braking (continued)

1. INTRODUCTION

The purpose of the Rendezvous Terminal Phase Braking Program is to provide the means of automatically bringing the primary vehicle (Orbiter) within desired station-keeping boundaries relative to the target vehicle (or satellite). To accomplish this task, the program of necessity contains navigation, targeting and guidance functions.

The program is initiated subsequent to the last midcourse maneuver of the rendezvous targeting sequence. Line-of-sight corrections, braking corrections, and filtering of rendezvous measurement sensor data to improve vehicle and target state estimates are performed in a sequential manner. At program initiation, the relative range is on the order of three to five miles.

When the primary vehicle has achieved a position (and velocity) relative to the target which places it within the desired station-keeping boundaries so that the station-keeping function can be initiated and maintained, the program is terminated.

NOMENCLATURE

c_i Measurement code identifying i^{th} measurement at t_m

f_i Thrust of the engine selected for the maneuver; used in the Powered Flight Guidance Routines

i_{prev} Previous range gate passed; subscript used in braking (range) gate loop

$\underline{i}_\rho$ Unit vector in direction of relative position vector, $\underline{\rho}$

$\underline{i}_s$ Unit vector which defines center of station-keeping boundary, relative to target vehicle

k_1 Constant used to determine the range at which each range gate search starts when approaching that particular range gate

k_2 Constant used to determine how often the line-of-sight targeting loop is entered; integer number of terminal phase program cycles

k_3 Constant value of range rate added to the minimum range rate at a given range to insure primary vehicle intercept of target vehicle

k_4 Constant used to determine how often the range-rate correction targeting loop is entered

m Current estimated primary vehicle mass

M_{R-B} Transformation matrix from reference coordinate frame to body axes coordinate frame

9.8.1.2 Braking (continued)

M_{R-LOS}	Transformation matrix from reference coordinate frame (in which vehicle states are expressed) to LOS coordinate frame axes
M_{R-M}	Transformation matrix from reference coordinate frame to measurement coordinate frame
M_{R-SM}	Transformation matrix from reference coordinate frame to stable member coordinate frame
M_{NB-B}	Transformation matrix from navigation base frame to body axes
M_{NB-M}	Transformation from navigation base to measurement coordinate frame
M_{SM-NB}	Transformation matrix from stable member coordinate frame to navigation base
n	Number of discrete braking gates in the range/ range rate correction schedule
q_i	i th measured relative parameter at t_m
$\underline{r}_P$	Primary vehicle position vector
$\underline{r}_T$	Target vehicle position vector
$\underline{r}(t_A)$	Aimpoint vector used in Lambert targeting calculations
s_B	Switch which controls braking gate targeting cycle
s_{eng}	Engine select switch
s_{GM}	Switch which indicates guidance mode to be used in Powered Flight Guidance Routine; "2"- two axis thrusting; "3"- modified Delta-v mode; "4" - modified Lambert mode
s_{LOS}	Switch which controls line-of-sight targeting cycle

9.8.1.2 Braking (continued)

s_{LAM} Switch used to select type of targeting scheme used in the Terminal Phase Braking Sequencing Program

s_{nav} Switch used to select method used to process the sensor data

$s_{\Delta v}$ Switch which indicates if a velocity correction is to be made or not

t_c Current time

t_{ig} Maneuver ignition time

t_m Measurement time

t_s Time associated with primary and target vehicle state vectors

$\underline{v}_P$ Primary vehicle velocity vector

$\underline{v}_T$ Target vehicle velocity vector

W_I Initial filter weighting matrix

$\overline{\alpha_i^2}$ Measurement variance used in filter to process i^{th} measurement data

β Elevation angle of line-of-sight in measurement frame

δt_B Delta time to ignition for a range-rate correction maneuver

δt_m Time between successive measurements within the measurement loop

Δt_m Basic sequencing cycle time

9.8.1.2 Braking (continued)

$\Delta\underline{v}_B$	Velocity change expressed in the body coordinate frame
Δv_{LIM}	Magnitude of velocity change below which no maneuver will be applied
$\Delta\underline{v}_{LOS}$	Velocity change expressed in line-of-sight coordinate frame
γ	Value of station-keeping boundary cone angle
μ	Gravitational constant of the earth
$\underline{\nu}$	Relative velocity vector
ν_u	Upper bound on station-keeping velocity
ν_ℓ	Lower bound on station-keeping velocity
$\underline{\omega}_{LIM}$	Angular velocity lower limit below which no line-of-sight correction is made; value to which line-of-sight angular velocity is driven if a line-of-sight correction is made
$\underline{\omega}_{LOS}$	Angular velocity vector of the line-of-sight between the primary and target vehicle
ω_{LOS}	Magnitude of $\underline{\omega}_{LOS}$
ρ	Magnitude of relative position vector, $\underline{\rho}$
$\dot{\rho}$	Range rate between the primary and target vehicles
$\underline{\rho}$	Relative position vector
$\rho_{B\,i}$	Range of the i^{th} braking gate
ρ_ℓ	Lower bound on station-keeping position
$\dot{\rho}_{max_i}$	Range rate desired at i^{th} braking gate and maximum between braking gates i and i + 1

9.8.1.2 Braking (continued)

$\dot{\rho}_{\min_i}$	Minimum range rate desired between braking gates i and i + 1
$\underline{\rho}_{\mathrm{off}(LV)}$	Offset aimpoint relative to target point expressed in target local vertical frame
ρ_u	Upper bound on station-keeping position
θ	Azimuth angle of line-of-sight in measurement frame
$[\ ]_m$	Vector expressed in measurement coordinate frame
$(\ )'$	Prime indicates previous values of a variable, e.g. prior measurement parameters, prior measurement time, etc.

2. FUNCTIONAL FLOW DIAGRAM

The functional flow diagram for the Rendezvous Terminal Phase Braking Program is shown in Figure 1. The program is initiated after the last rendezvous midcourse correction maneuver of the rendezvous targeting sequence. The relative range between the primary and target vehicle at this point is on the order of three to five miles and closing.

The program sequencing begins with the updating of the estimated primary and target vehicle relative state parameters with the appropriate sensor data.

These relative parameters are then used in the Terminal Phase Targeting Program where the necessary calculations are performed to see if a line-of-sight and/or a braking correction is required to maintain the desired characteristics of the rendezvous trajectory. The line-of-sight corrections (if performed) maintain the intercept by nulling out line-of-sight rates which exceed a desired rate. At selected ranges between the primary and target vehicles, braking corrections are performed to reduce the closing rate to that specified in the terminal range/range rate profile, if the closing rate exceeds the desired value. During the program sequencing a continuous check is made to insure that the closing rate is sufficiently high so that the primary vehicle will intercept the target.

If either a line-of-sight correction and/or range-rate correction is necessary, the velocity correction is applied using the appropriate guidance mode.

The program sequencing is then repeated. The program is terminated when the desired relative position and velocity conditions are achieved so that the station-keeping mode can be initiated and maintained.

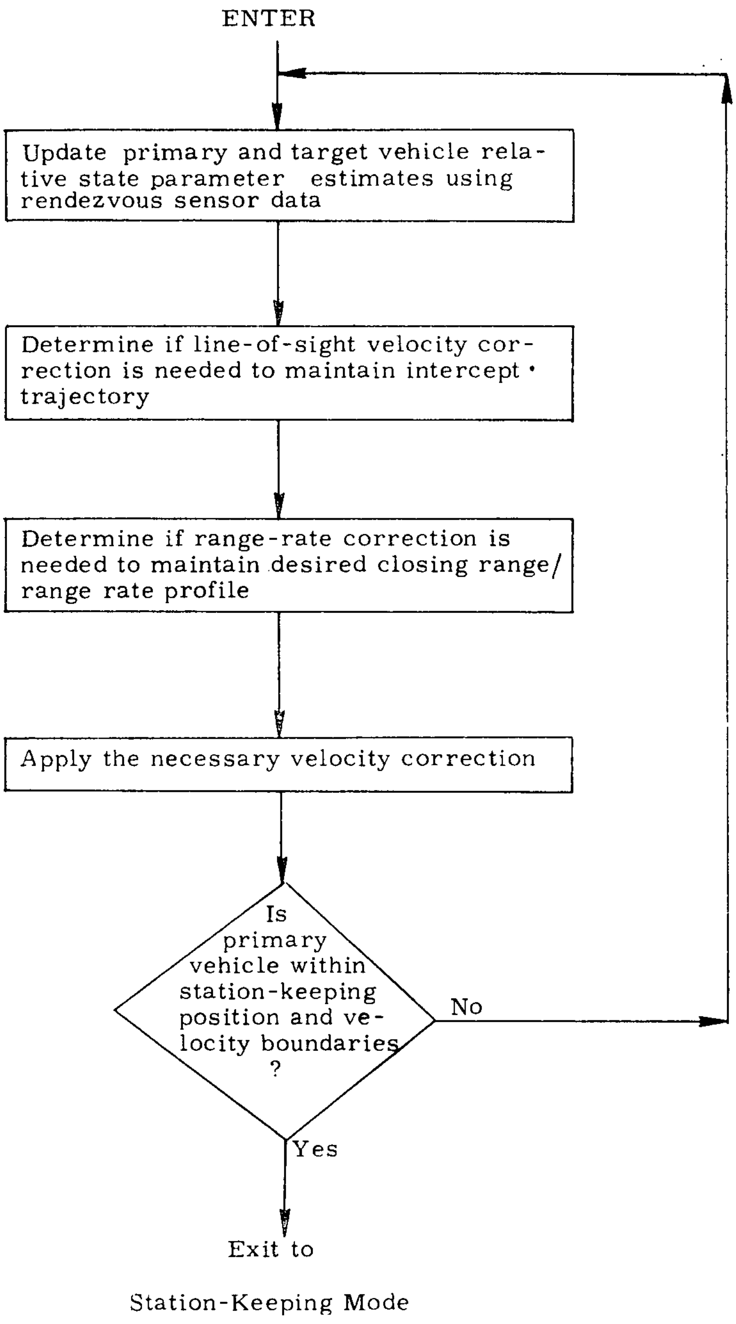

ENTER
Update primary and target vehicle relative state parameter estimates using rendezvous sensor data
Determine if line-of-sight velocity correction is needed to maintain intercept trajectory
Determine if range-rate correction is needed to maintain desired closing range/ range rate profile
Apply the necessary velocity correction
Is primary vehicle within station-keeping position and velocity boundaries ?
No
Yes
Exit to
Station-Keeping Mode

Figure 1. Rendezvous Terminal Phase Braking Program, Functional Flow Diagram

3. INPUT AND OUTPUT VARIABLES

The Terminal Phase Braking Program consists of three basic functions-navigation, targeting and guidance. The following is a description of the input and output variables for the basic sequencing program, the navigation program and the targeting program. The Powered Flight Guidance Program is described in Ref. 3.

3.1 Terminal Phase Braking Sequencing Program

Input Variables

$\underline{r}_P(t_s)$, $\underline{v}_P(t_s)$	Estimated primary vehicle state vector at time t_s
$\underline{r}_T(t_s)$, $\underline{v}_T(t_s)$	Estimated target vehicle state vector at time t_s
n	Number of discrete range gate corrections
ρ_{B0}, ... ρ_{Bn}	Range values of the n braking gates
$\dot{\rho}_{B0}$, ... $\dot{\rho}_{Bn}$	Range rates desired at the n braking gates
s_{nav}	Switch used to select method to process sensor data
s_{LAM}	Switch used to select type of targeting scheme used in the Terminal Phase Braking Program

Output Variables

$\underline{r}_P$, $\underline{v}_P$	Primary vehicle state vector for use in station-keeping phase
$\underline{r}_T$, $\underline{v}_T$	Target vehicle state vector for use in station-keeping phase
t_s	Time tag of above state vectors

3.2 Relative State Parameter Updating Routine

This routine utilizes raw sensor data at two successive times to determine certain required relative parameters. There are no input variables to this routine. The output variables are described below

Output Variables

$\underline{i}_\rho$	Unit vector in direction of relative range vector
$\rho(t_m)$	Magnitude of relative range vector at t_m
$\dot{\rho}(t_m)$	Range rate between the primary and target vehicles at t_m
$\underline{\omega}_{LOS}(t_m)$	Line-of-sight angular velocity vector between the primary and target vehicles at t_m
$\omega_{LOS}(t_m)$	Magnitude of the line-of-sight angular velocity
M_{R-LOS}	Transformation matrix from reference to line-of-sight coordinates at t_m
M_{R-B}	Transformation matrix from reference to body coordinates at t_m

3.3 Terminal Phase Targeting Routine

Input Variables

$\underline{r}_P, \underline{v}_P$	Primary vehicle state vector
$\underline{r}_T, \underline{v}_T$	Target vehicle state vector
ρ	Relative range between primary and target vehicle
$\dot{\rho}$	Range rate between primary and target vehicle

9.8.1.2 Braking (continued)

$\underline{i}_{\rho}$	Unit vector in direction of relative range vector
$\underline{\omega}_{LOS}$	Angular velocity vector of the line-of-sight between the primary and target vehicles
ω_{LOS}	Magnitude of $\underline{\omega}_{LOS}$
M_{R-B}	Matrix transformation between the reference coordinate frame and body coordinates
M_{R-LOS}	Matrix transformation between the reference coordinate frame and the line-of-sight coordinate frame
s_{LAM}	Switch used to select type of targeting scheme
t_c	Current time

Output Variables

$\underline{i}_N$	Unit normal to the trajectory plane (in the direction of the angular momentum at ignition)
$\underline{r}_{1c}$	Offset target position
t_{ig}	Time of upcoming maneuver
$\Delta\underline{v}_B$	Velocity change of upcoming maneuver in body coordinates
$\Delta\underline{v}_{LOS}$	Velocity change of upcoming maneuver in line-of-sight coordinates
$\Delta\underline{v}_{LV}$	Velocity correction in local vertical coordinates
s_{eng}	Engine select switch
$s_{\Delta v}$	Switch which indicates if velocity correction is to be performed during this sequencing of the Terminal Phase Braking Program

9.8.1.2 Braking (continued)

s_{proj} Switch set when the target vector must be projected into the plane defined by $\underline{i}_N$

s_{GM} Switch which indicates guidance mode to be used in the Powered Flight Guidance Sequencing Program

4. DESCRIPTION OF EQUATIONS

4.1 Terminal Phase Braking Sequencing Program

The Terminal Phase Braking Sequencing Program (Figure 4), which is the main sequencing program for the terminal phase, is initiated after the last midcourse correction in the rendezvous targeting sequence.

The range/range rate terminal braking schedule used in the program is determined prior to the initiation of the program and consists of discrete range gates and their associated desired range rates. A minimum range rate is also specified throughout the terminal phase to insure primary vehicle intercept of the target vehicle. An example of such a braking schedule is shown in Figure 2.

The sequencing begins with the processing of rendezvous sensor data to obtain estimates of range, range rate, line-of-sight rates, etc. These estimates are derived from either processing the sensor data in the Relative State Updating Routine (which is also used throughout the rendezvous sequence, Ref. 2) or from the raw sensor data itself, depending on the input value of the switch s_{nav}.

These relative parameter estimates are then used in the Terminal Phase Targeting Routine to determine if a maneuver (either a braking maneuver, line-of-sight correction or a combination of both) is to be performed. The associated maneuver time and guidance parameters are also computed.

If a maneuver is to be performed, the Powered Flight Guidance Sequencing Program (similar to the Servicer Routine in Apollo) is entered with the appropriate inputs to accomplish the maneuver.

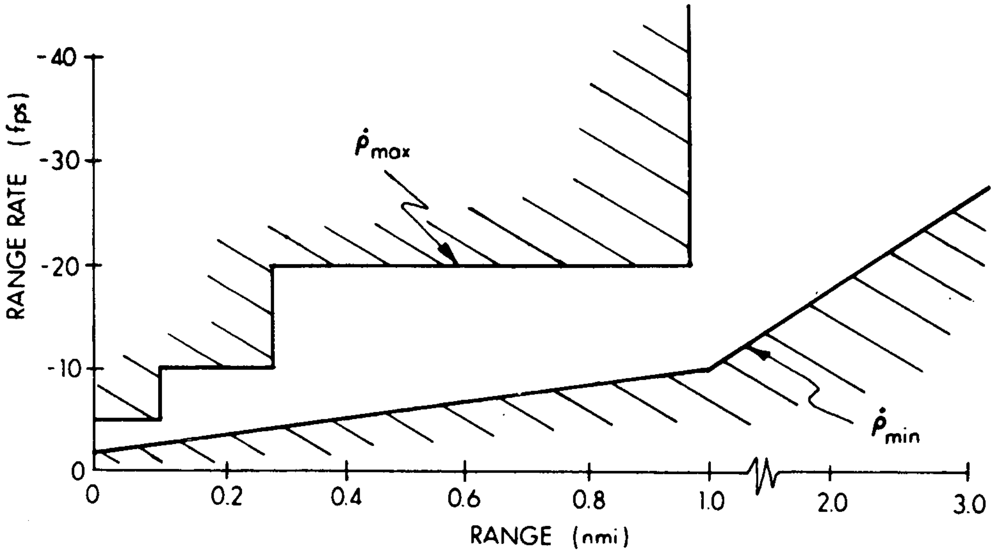
This basic sequencing is repeated until the primary vehicle is within desired station-keeping boundaries relative to the target vehicle (Figure 3).

4.2 Relative State Parameter Updating Routine

The Relative State Parameter Updating Routine (Figure 5) is used when it is desired to utilize the raw sensor data to obtain the necessary relative parameters for the targeting routine.

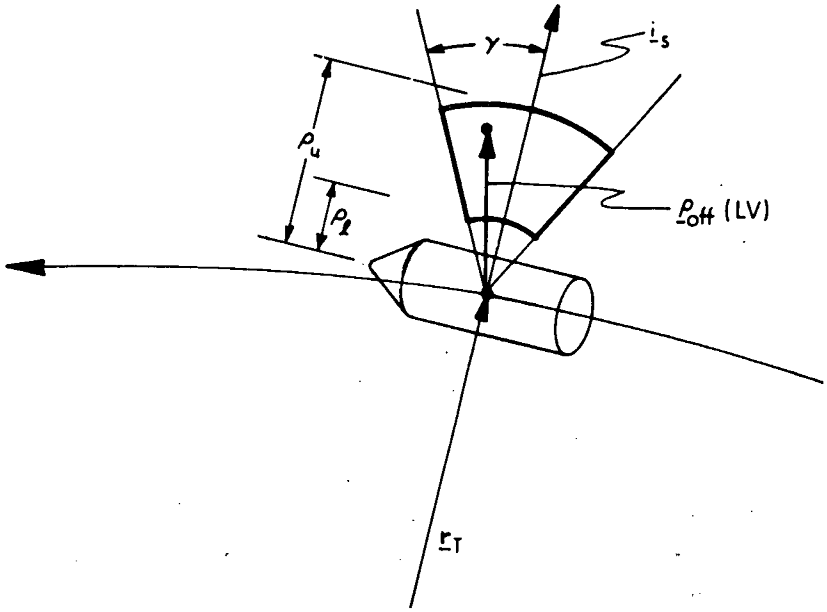
The rendezvous sensors are read at t_m' along with the associated IMU gimbal angles. The sensors typically will consist of a ranging device and a star tracker which measures the line-of-sight angles in the measurement frame. Using these azimuth and elevation angles, the line-of-sight vector in reference coordinates is computed.

δt_m seconds later the rendezvous sensors are again read and the line-of-sight vector again calculated in the reference coordinate system.



NOTE: Change of scale on range axis.

Figure 2. Typical Range/Range Rate Schedule



γ - Cone angle of station-keeping zone

ρ_u, ρ_ℓ - Upper and lower values of station-keeping boundaries

$\underline{\rho}_{off}$(LV) - Relative offset vector in target vehicle local vertical, used to target Lambert braking corrections; primary vehicle will intercept this point in the station-keeping zone

Figure 3. Station-Keeping Boundaries–Station-Keeping Above

The target vehicle state vector is extrapolated to t_m using the Precision State Extrapolation Routine and a primary vehicle state vector is constructed using the target vehicle state and the relative parameters just computed.

These state vectors will be used in the Powered Flight Guidance Routines during powered flight maneuvers if a maneuver is to be performed.

4.3 Terminal Phase Targeting Routine

The Terminal Phase Targeting Routine (Figure 6) computes the necessary maneuvers to maintain the primary vehicle on an intercept with the target vehicle while keeping the range/range rate profile within the desired boundaries.

Two modes of operation are available. The first mode is referred to as automatic line-of-sight control braking and the second automatic Lambert braking.

When s_{LAM} is set to zero, the automatic line-of-sight control braking mode is used. If the line-of-sight rate as determined from processing the sensor data is above a set limit (typically 0.1 m/sec), the line-of-sight correction necessary to drive the line-of-sight rate to some level is computed and the appropriate ignition time, engine selection and guidance mode switches are set. Since these line-of-sight corrections are made frequently, the maneuver magnitudes are small (several feet/second or less) and hence the small RCS thrusters are used to effect the maneuver. The maneuver is accomplished by using two-axis thrusting normal to the line-of-sight.

The line-of-sight correction check is typically made every two cycles of the main program. (Line-of-sight cycling is determined by k_2)

The range/range rate checks, to insure that the desired terminal profile is being followed, are made after the line-of-sight checks. If the range rate at certain pre-selected ranges exceeds the desired range rate a braking maneuver is performed to reduce the closing rate. Continuous checks are made to insure that the closing rate is above the minimum value to maintain intercept. If it is not, then the closing rate is increased.

If a range-rate correction is necessary, the appropriate guidance switches and modes are set. The ignition time is set δt_B seconds from the present time since these corrections typically involve significant maneuver sizes, and attitude maneuvers to use the appropriate engines will be necessary.

The second mode of operation, the automatic Lambert braking, targets for an intercept point (either the target vehicle or a point offset from the target vehicle indicated by $\underline{\rho}_{off}$, Figure 3) at each pre-selected braking gate. No independent line-of-sight corrections are made in this mode since the targeting

9.8.1.2 Braking (continued)

implicitly corrects line-of-sight rate to insure intercept, at each braking gate correction.

When the range between the vehicles reaches $(1 + k_1)$ times the preselected range gate, the time of arrival at the range gate is computed. The calculation assumes the present range-rate remains constant until the range gate is reached. The primary and target vehicle state vectors are then advanced to this ignition time.

The time of arrival at the intercept point is redefined by the equation

$$t_{go} = \frac{\text{(Range at ignition)}}{\text{(Desired range rate at this range gate)}}$$

This t_{go} is then used to calculate a new target vector for use in the Lambert routine to determine the necessary velocity correction.

By redefining the intercept point in this manner, the Lambert solution forces a reduction in range rate to the desired range rate, insuring intercept in a length of time equivalent to the time it would take to travel the present range at the constant desired range rate. The line-of-sight rate is automatically corrected in the Lambert solution to assure intercept.

The new target vector, time-of-arrival, ignition time and guidance mode switches are then used in the Powered Flight Guidance Routines (Ref. 3) to effect the maneuver.

5. DETAILED FLOW DIAGRAMS

This section contains detailed flow diagrams of the Terminal Phase Braking Sequencing Program, the Relative State Parameter Updating Routine, and the Terminal Phase Targeting Routine.

Each input and output variable in the routine and subroutine call statments can be followed by a symbol in brackets. This symbol identifies the notation for the corresponding variable in the detailed description and flow diagrams of the called routine. When identical notation is used, the bracketed symbol is omitted.

9.8.1.2 Braking (continued)

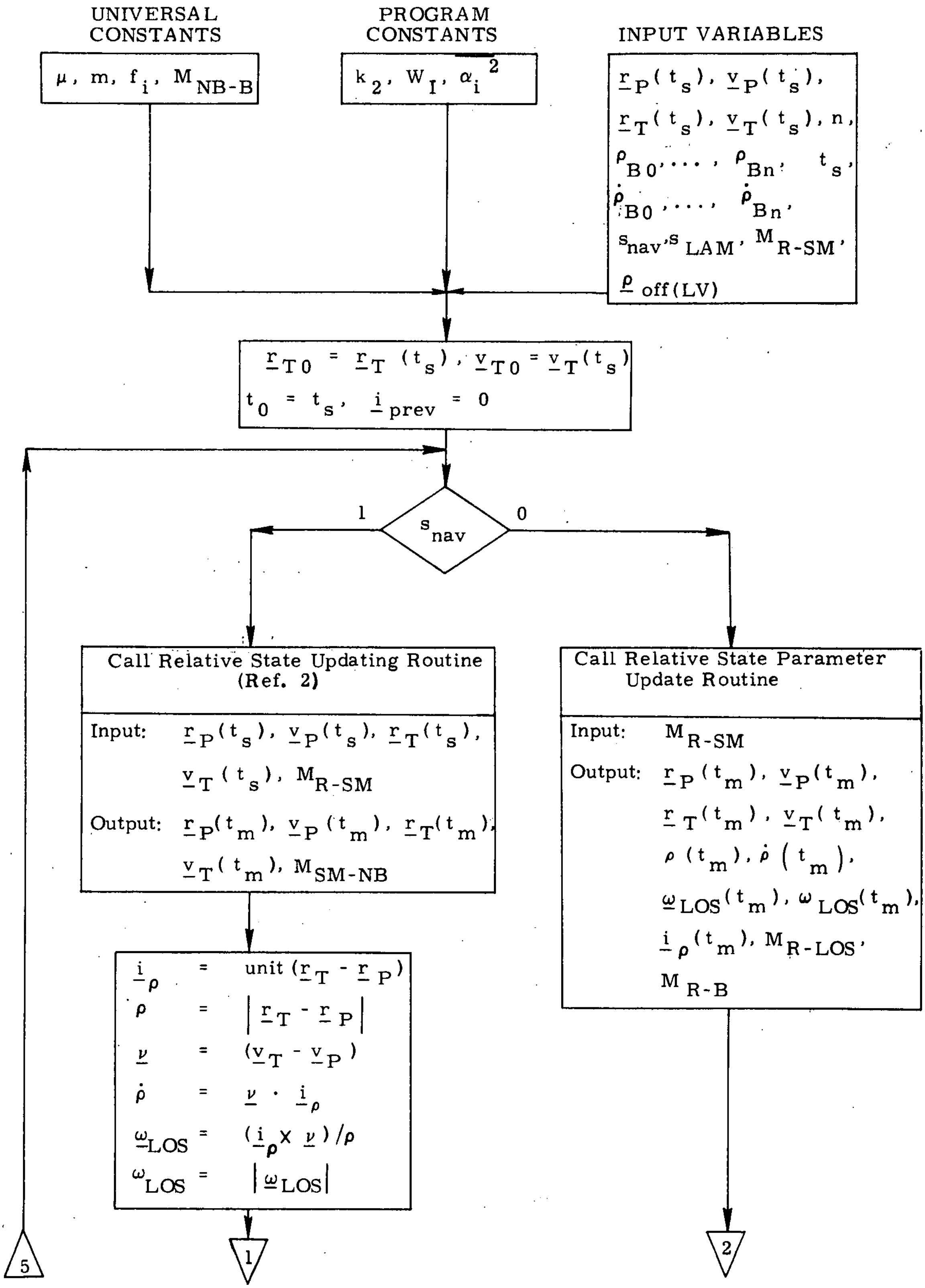
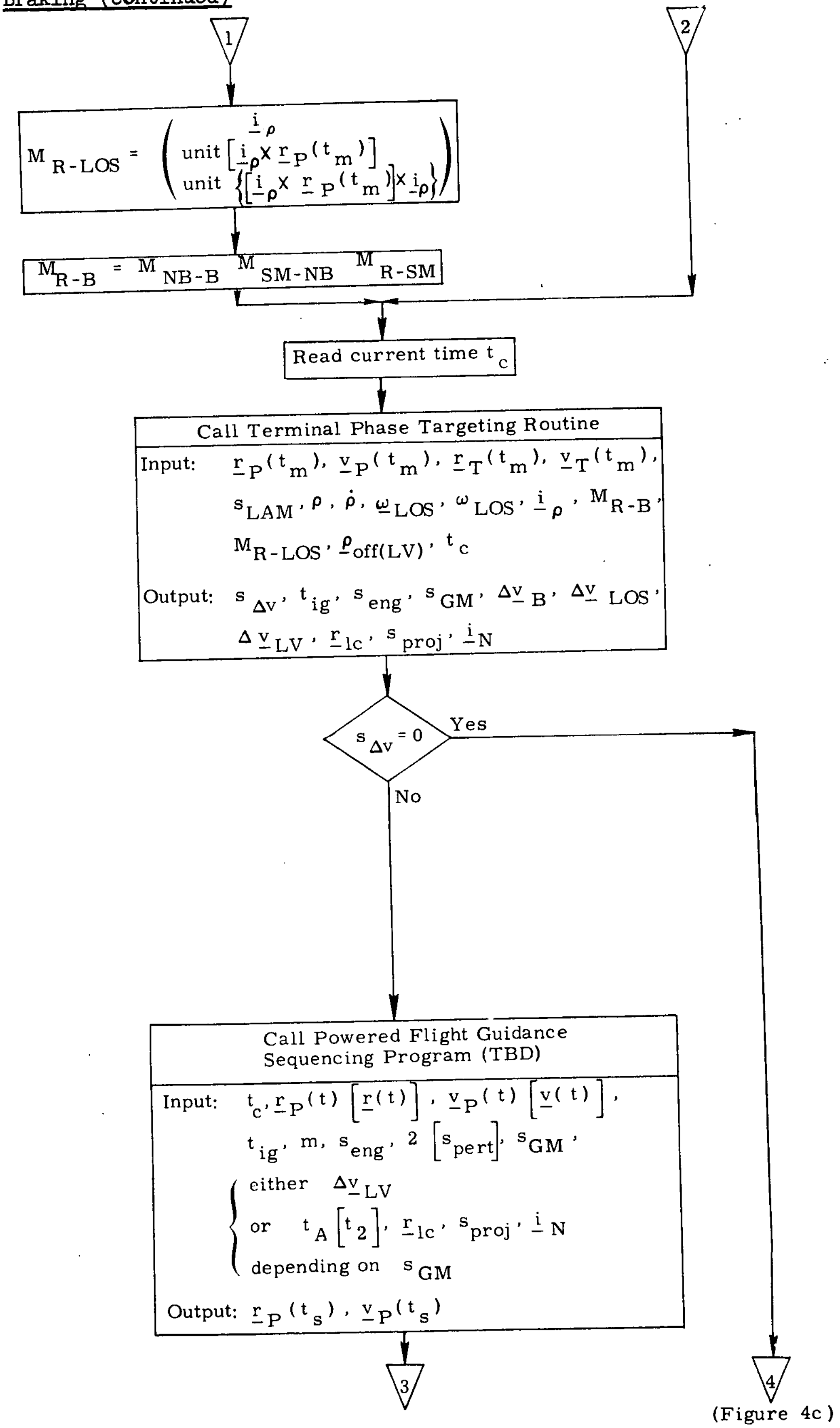

UNIVERSAL CONSTANTS
μ, m, f_i, M_{NB-B}
PROGRAM CONSTANTS
k_2, W_I, $\overline{\alpha_i}^2$
INPUT VARIABLES
$\underline{r}_P(t_s)$, $\underline{v}_P(t_s)$, $\underline{r}_T(t_s)$, $\underline{v}_T(t_s)$, n, $\rho_{B0}, \ldots, \rho_{Bn}$, t_s, $\dot{\rho}_{B0}, \ldots, \dot{\rho}_{Bn}$, s_{nav}, s_{LAM}, M_{R-SM}, $\underline{\rho}_{off(LV)}$
$\underline{r}_{T0} = \underline{r}_T(t_s)$, $\underline{v}_{T0} = \underline{v}_T(t_s)$
$t_0 = t_s$, $\underline{i}_{prev} = 0$
s_{nav}
1
0
Call Relative State Updating Routine (Ref. 2)
Input: $\underline{r}_P(t_s)$, $\underline{v}_P(t_s)$, $\underline{r}_T(t_s)$, $\underline{v}_T(t_s)$, M_{R-SM}
Output: $\underline{r}_P(t_m)$, $\underline{v}_P(t_m)$, $\underline{r}_T(t_m)$, $\underline{v}_T(t_m)$, M_{SM-NB}
$\underline{i}_\rho$ = unit $(\underline{r}_T - \underline{r}_P)$
$\rho = |\underline{r}_T - \underline{r}_P|$
$\underline{\nu} = (\underline{v}_T - \underline{v}_P)$
$\dot{\rho} = \underline{\nu} \cdot \underline{i}_\rho$
$\underline{\omega}_{LOS} = (\underline{i}_\rho \times \underline{\nu})/\rho$
$\omega_{LOS} = |\underline{\omega}_{LOS}|$
Call Relative State Parameter Update Routine
Input: M_{R-SM}
Output: $\underline{r}_P(t_m)$, $\underline{v}_P(t_m)$, $\underline{r}_T(t_m)$, $\underline{v}_T(t_m)$, $\rho(t_m)$, $\dot{\rho}(t_m)$, $\underline{\omega}_{LOS}(t_m)$, $\omega_{LOS}(t_m)$, $\underline{i}_\rho(t_m)$, M_{R-LOS}, M_{R-B}
5
1
2

Figure 4a. Terminal Phase Braking Sequencing Program, Detailed Flow Diagram

9.8-1.2 <u>Braking (continued)</u>



1
2
$M_{R-LOS} = \begin{pmatrix} \underline{i}_\rho \\ \text{unit}\left[\underline{i}_\rho \times \underline{r}_P(t_m)\right] \\ \text{unit}\left\{\left[\underline{i}_\rho \times \underline{r}_P(t_m)\right] \times \underline{i}_\rho\right\} \end{pmatrix}$
$M_{R-B} = M_{NB-B}\, M_{SM-NB}\, M_{R-SM}$
Read current time t_c
Call Terminal Phase Targeting Routine
Input: $\underline{r}_P(t_m)$, $\underline{v}_P(t_m)$, $\underline{r}_T(t_m)$, $\underline{v}_T(t_m)$, s_{LAM}, ρ, $\dot{\rho}$, $\underline{\omega}_{LOS}$, ω_{LOS}, $\underline{i}_\rho$, M_{R-B}, M_{R-LOS}, $\underline{\rho}_{off(LV)}$, t_c
Output: $s_{\Delta v}$, t_{ig}, s_{eng}, s_{GM}, $\Delta\underline{v}_B$, $\Delta\underline{v}_{LOS}$, $\Delta\underline{v}_{LV}$, $\underline{r}_{lc}$, s_{proj}, $\underline{i}_N$
$s_{\Delta v} = 0$
Yes
No
Call Powered Flight Guidance Sequencing Program (TBD)
Input: t_c, $\underline{r}_P(t)$ $\left[\underline{r}(t)\right]$, $\underline{v}_P(t)$ $\left[\underline{v}(t)\right]$, t_{ig}, m, s_{eng}, 2 $\left[s_{pert}\right]$, s_{GM},
either $\Delta\underline{v}_{LV}$
or t_A $\left[t_2\right]$, $\underline{r}_{lc}$, s_{proj}, $\underline{i}_N$
depending on s_{GM}
Output: $\underline{r}_P(t_s)$, $\underline{v}_P(t_s)$
3
4
(Figure 4c)

Figure 4b. Terminal Phase Braking Sequencing Program, Detailed Flow Diagram

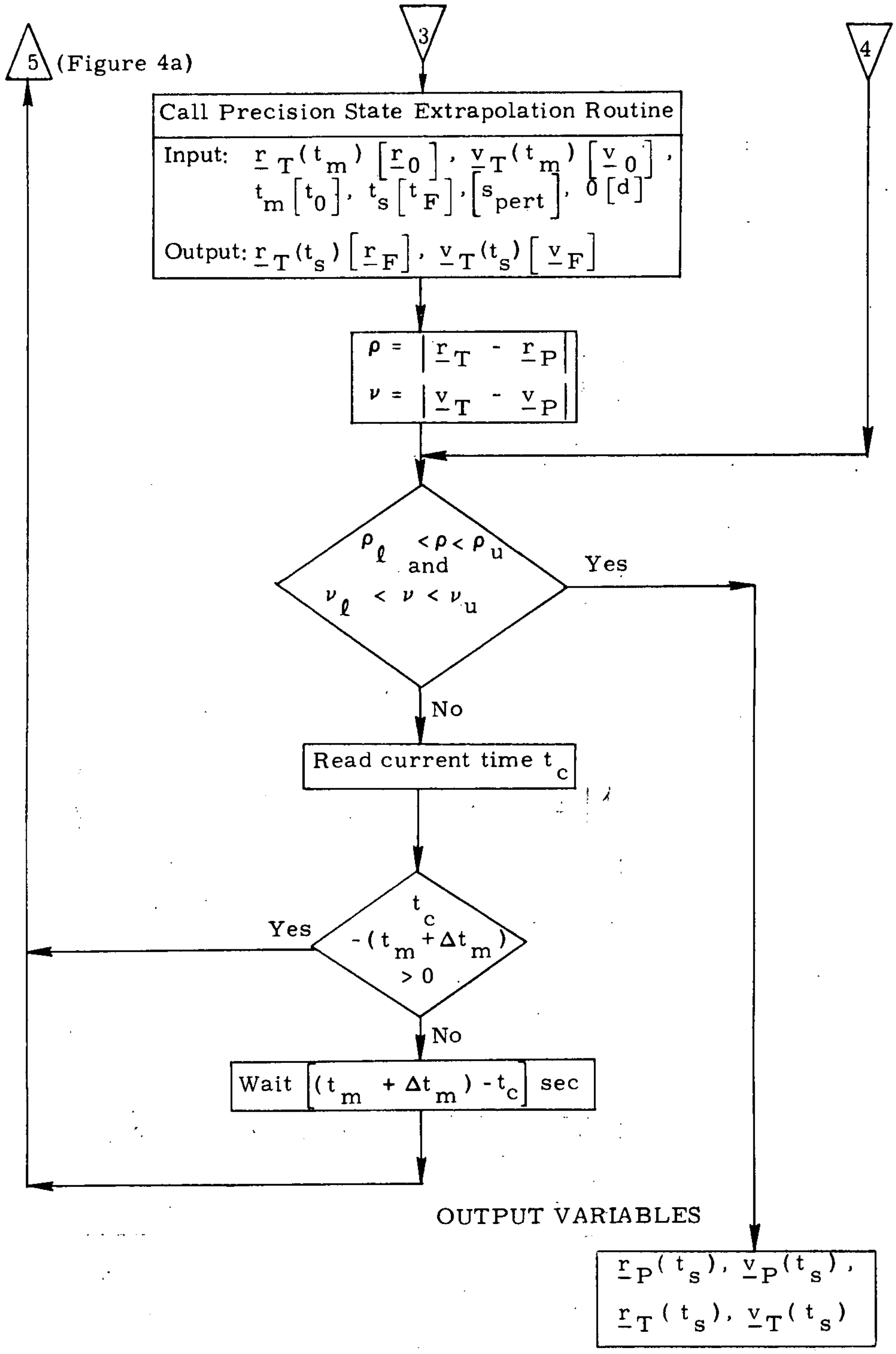

5
(Figure 4a)
3
4
Call Precision State Extrapolation Routine
Input: $\underline{r}_T(t_m)$ $[\underline{r}_0]$, $\underline{v}_T(t_m)$ $[\underline{v}_0]$, t_m $[t_0]$, t_s $[t_F]$, $[s_{pert}]$, 0 [d]
Output: $\underline{r}_T(t_s)$ $[\underline{r}_F]$, $\underline{v}_T(t_s)$ $[\underline{v}_F]$
$\rho = |\underline{r}_T - \underline{r}_P|$
$\nu = |\underline{v}_T - \underline{v}_P|$
$\rho_\ell < \rho < \rho_u$ and $\nu_\ell < \nu < \nu_u$
Yes
No
Read current time t_c
$t_c - (t_m + \Delta t_m) > 0$
Yes
No
Wait $[(t_m + \Delta t_m) - t_c]$ sec
OUTPUT VARIABLES
$\underline{r}_P(t_s)$, $\underline{v}_P(t_s)$, $\underline{r}_T(t_s)$, $\underline{v}_T(t_s)$

Figure 4c. Terminal Phase Braking Sequencing Program, Detailed Flow Diagram

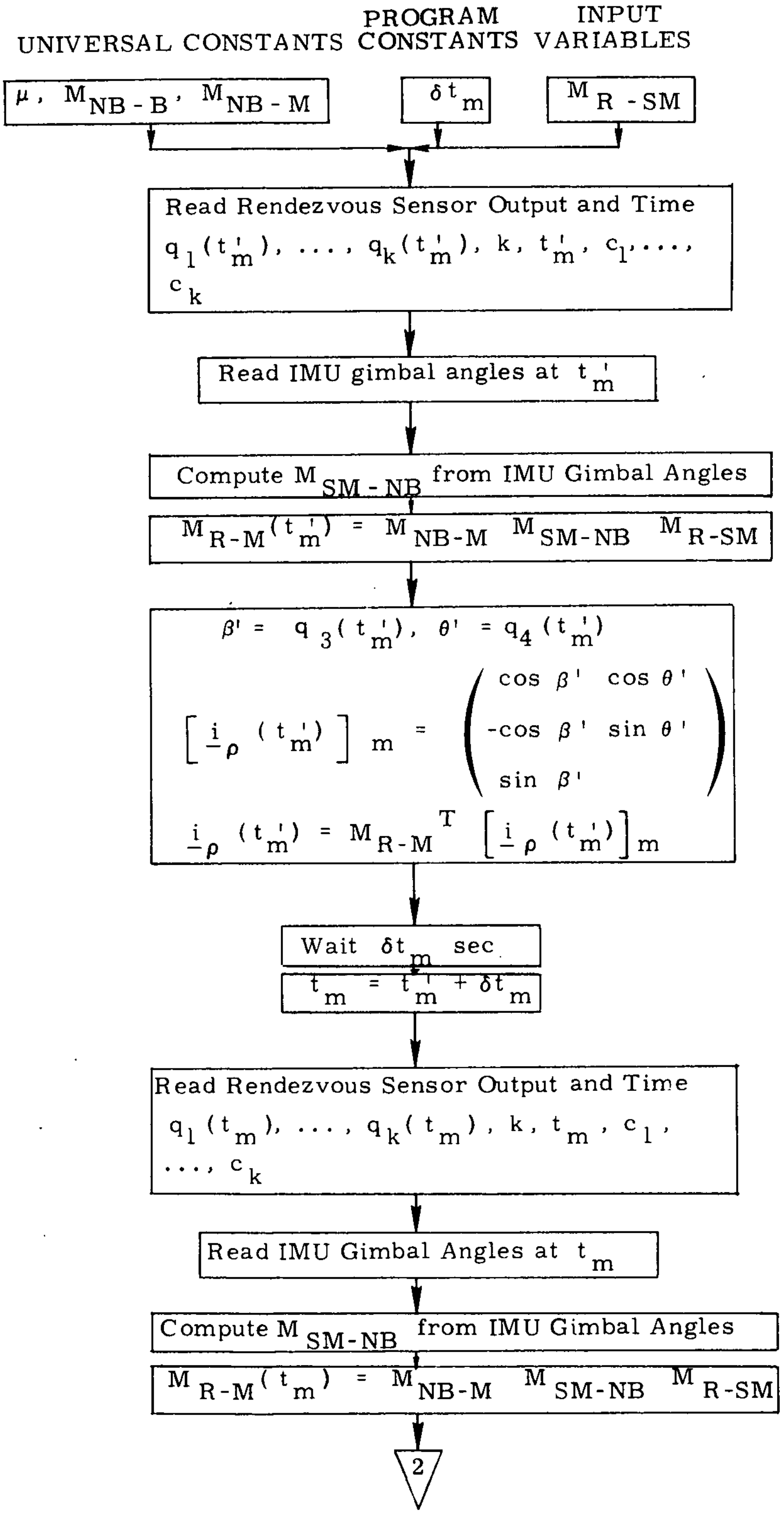

UNIVERSAL CONSTANTS
PROGRAM CONSTANTS
INPUT VARIABLES
μ, M_{NB-B}, M_{NB-M}
δt_m
M_{R-SM}
Read Rendezvous Sensor Output and Time
$q_1(t_m'), \ldots, q_k(t_m'), k, t_m', c_1, \ldots, c_k$
Read IMU gimbal angles at t_m'
Compute M_{SM-NB} from IMU Gimbal Angles
$M_{R-M}(t_m') = M_{NB-M} \; M_{SM-NB} \; M_{R-SM}$
$\beta' = q_3(t_m'), \; \theta' = q_4(t_m')$
$\left[\underline{i}_\rho(t_m')\right]_m = \begin{pmatrix} \cos\beta' \cos\theta' \\ -\cos\beta' \sin\theta' \\ \sin\beta' \end{pmatrix}$
$\underline{i}_\rho(t_m') = M_{R-M}^T \left[\underline{i}_\rho(t_m')\right]_m$
Wait δt_m sec
$t_m = t_m' + \delta t_m$
Read Rendezvous Sensor Output and Time
$q_1(t_m), \ldots, q_k(t_m), k, t_m, c_1, \ldots, c_k$
Read IMU Gimbal Angles at t_m
Compute M_{SM-NB} from IMU Gimbal Angles
$M_{R-M}(t_m) = M_{NB-M} \; M_{SM-NB} \; M_{R-SM}$
2

Figure 5a. Relative State Parameter Updating Routine, Detailed Flow Diagram

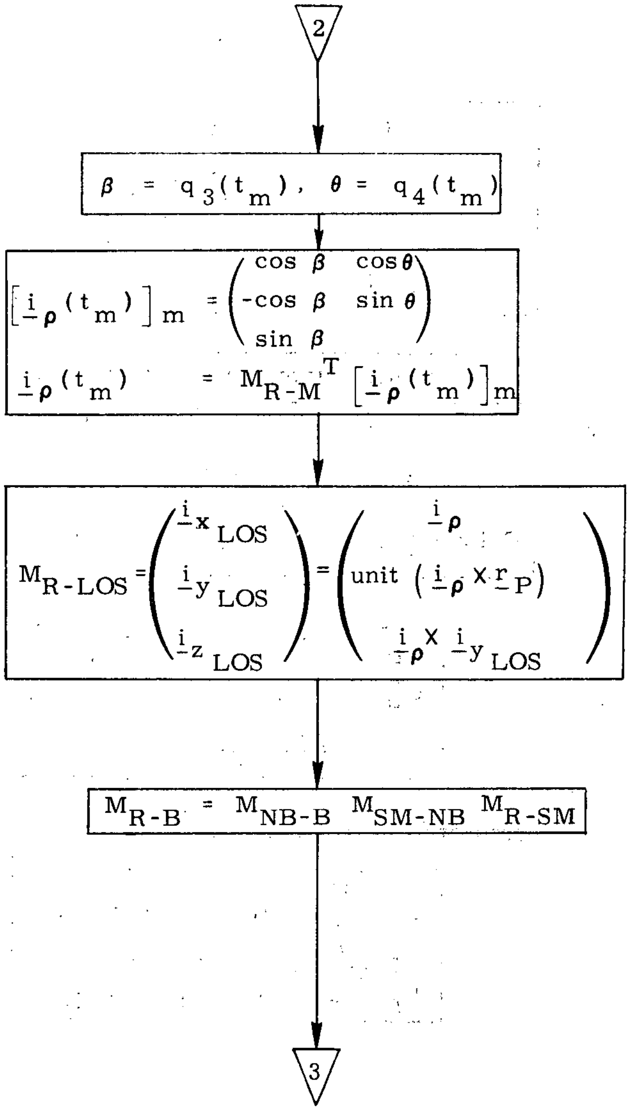
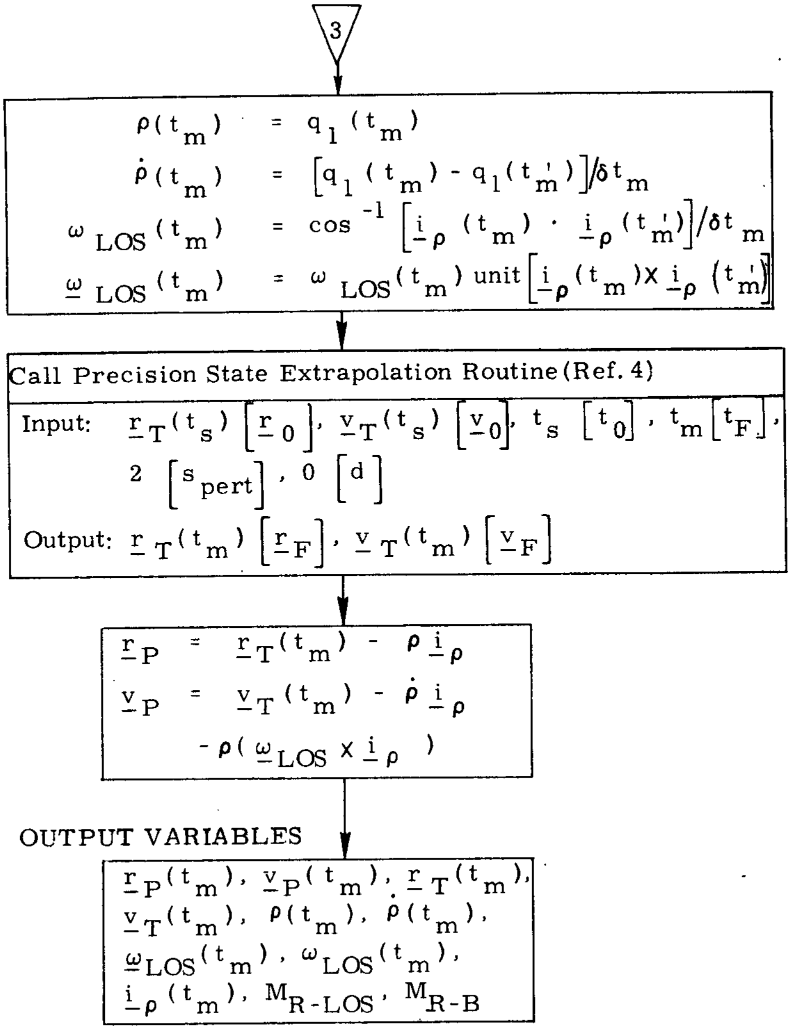

2
$\beta = q_3(t_m)$, $\theta = q_4(t_m)$
$[\underline{i}_\rho(t_m)]_m = \begin{pmatrix}\cos\beta \cos\theta \\ -\cos\beta \sin\theta \\ \sin\beta\end{pmatrix}$
$\underline{i}_\rho(t_m) = M_{R-M}^T [\underline{i}_\rho(t_m)]_m$
$M_{R-LOS} = \begin{pmatrix}\underline{i}_{x\,LOS} \\ \underline{i}_{y\,LOS} \\ \underline{i}_{z\,LOS}\end{pmatrix} = \begin{pmatrix}\underline{i}_\rho \\ \text{unit}\ (\underline{i}_\rho \times \underline{r}_P) \\ \underline{i}_\rho \times \underline{i}_{y\,LOS}\end{pmatrix}$
$M_{R-B} = M_{NB-B}\ M_{SM-NB}\ M_{R-SM}$
3

Figure 5b. Relative State Parameter Updating Routine, Detailed Flow Diagram

9.8.1.2 Braking (continued)



$$\rho(t_m) = q_1(t_m)$$

$$\dot{\rho}(t_m) = \left[q_1(t_m) - q_1(t_m')\right]/\delta t_m$$

$$\omega_{LOS}(t_m) = \cos^{-1}\left[\underline{i}_\rho(t_m) \cdot \underline{i}_\rho(t_m')\right]/\delta t_m$$

$$\underline{\omega}_{LOS}(t_m) = \omega_{LOS}(t_m)\ \mathrm{unit}\left[\underline{i}_\rho(t_m) \times \underline{i}_\rho(t_m')\right]$$

Call Precision State Extrapolation Routine (Ref. 4)

Input: $\underline{r}_T(t_s)\ [\underline{r}_0]$, $\underline{v}_T(t_s)\ [\underline{v}_0]$, $t_s\ [t_0]$, $t_m\ [t_F]$, $2\ [s_{pert}]$, $0\ [d]$

Output: $\underline{r}_T(t_m)\ [\underline{r}_F]$, $\underline{v}_T(t_m)\ [\underline{v}_F]$

$$\underline{r}_P = \underline{r}_T(t_m) - \rho\,\underline{i}_\rho$$

$$\underline{v}_P = \underline{v}_T(t_m) - \dot{\rho}\,\underline{i}_\rho - \rho(\underline{\omega}_{LOS} \times \underline{i}_\rho)$$

OUTPUT VARIABLES

$\underline{r}_P(t_m)$, $\underline{v}_P(t_m)$, $\underline{r}_T(t_m)$, $\underline{v}_T(t_m)$, $\rho(t_m)$, $\dot{\rho}(t_m)$, $\underline{\omega}_{LOS}(t_m)$, $\omega_{LOS}(t_m)$, $\underline{i}_\rho(t_m)$, M_{R-LOS}, M_{R-B}

Figure 5c. Relative State Parameter Updating Routine, Detailed Flow Diagram

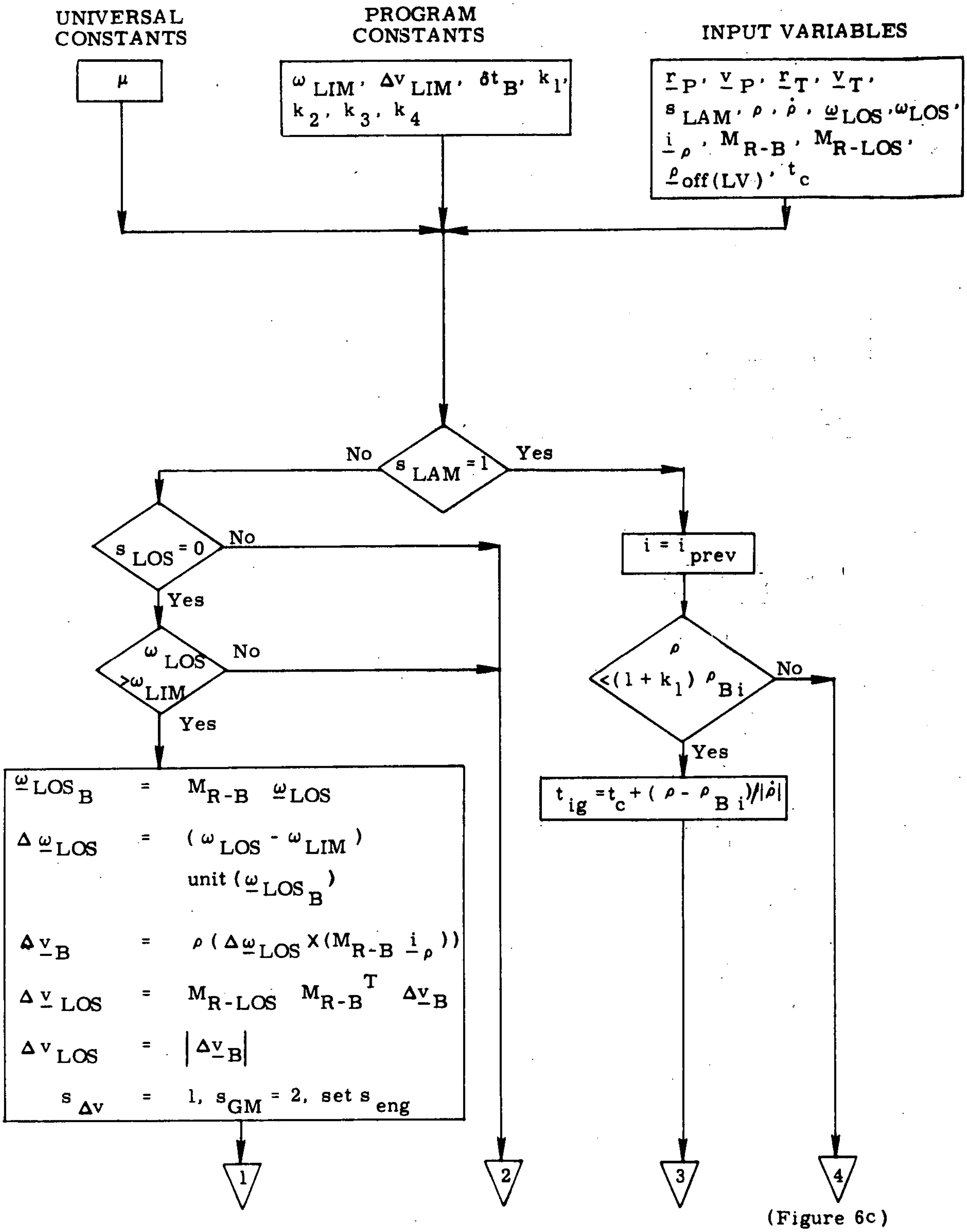

UNIVERSAL CONSTANTS
μ
PROGRAM CONSTANTS
ω_LIM, Δv_LIM, δt_B, k_1, k_2, k_3, k_4
INPUT VARIABLES
r_P, v_P, r_T, v_T, s_LAM, ρ, ρ̇, ω_LOS, ω_LOS, i_ρ, M_R-B, M_R-LOS, ρ_off(LV), t_c
s_LAM = 1
No
Yes
s_LOS = 0
No
Yes
ω_LOS > ω_LIM
No
Yes
ω_LOS_B = M_R-B ω_LOS
Δω_LOS = (ω_LOS - ω_LIM) unit (ω_LOS_B)
Δv_B = ρ (Δω_LOS X (M_R-B i_ρ))
Δv_LOS = M_R-LOS M_R-B^T Δv_B
Δv_LOS = |Δv_B|
s_Δv = 1, s_GM = 2, set s_eng
i = i_prev
ρ < (1 + k_1) ρ_B i
No
Yes
t_ig = t_c + (ρ - ρ_B i)/|ρ̇|
1
2
3
4
(Figure 6c)

Figure 6a. Terminal Phase Targeting Routine, Detailed Flow Diagram

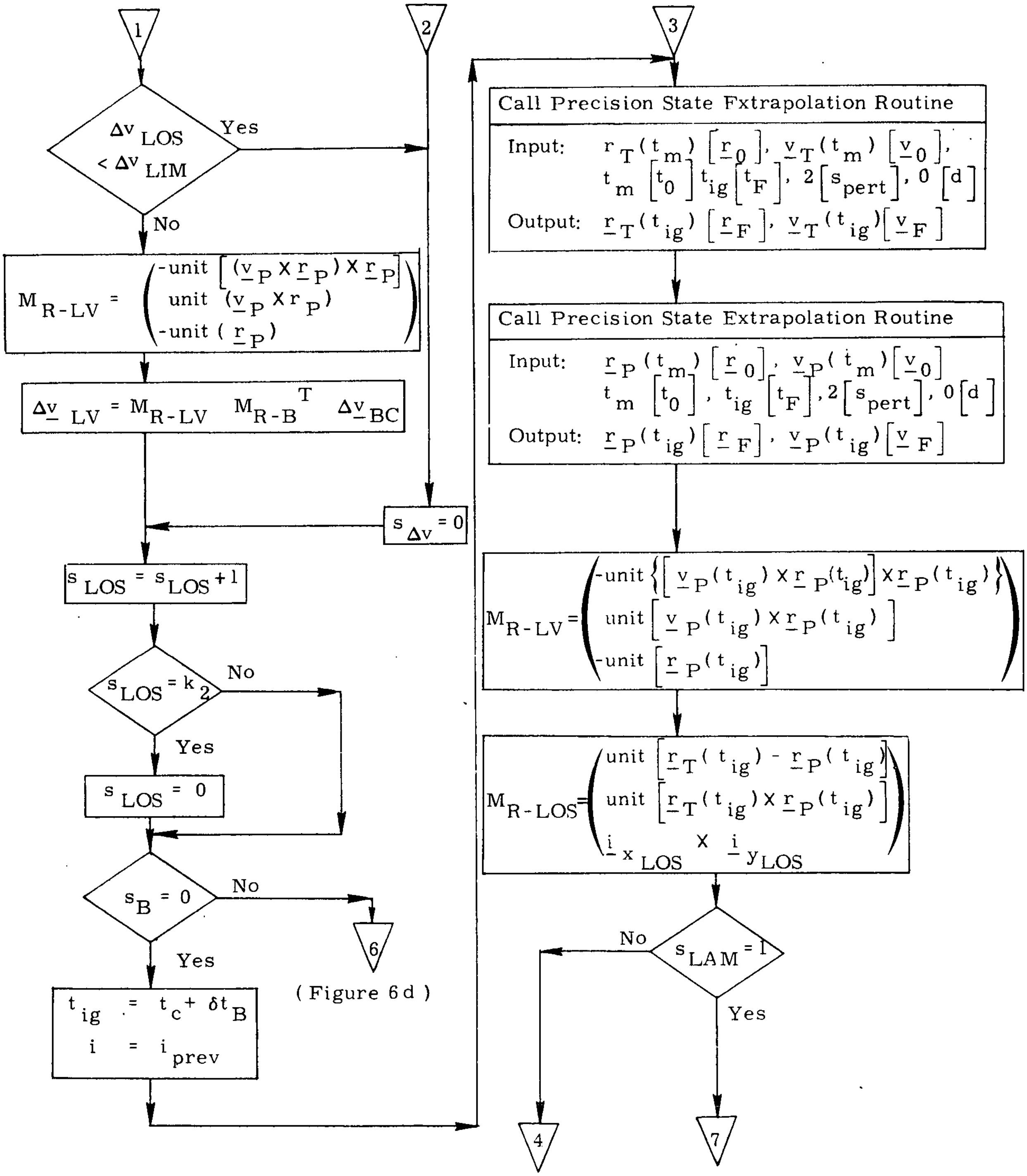

1
2
3
$\Delta v_{LOS} < \Delta v_{LIM}$
Yes
No
$M_{R-LV} = \begin{pmatrix} -\text{unit}\left[(\underline{v}_P \times \underline{r}_P) \times \underline{r}_P\right] \\ \text{unit}\ (\underline{v}_P \times r_P) \\ -\text{unit}\ (\underline{r}_P) \end{pmatrix}$
$\Delta \underline{v}_{LV} = M_{R-LV}\ M_{R-B}^T\ \Delta \underline{v}_{BC}$
$s_{\Delta v} = 0$
$s_{LOS} = s_{LOS} + 1$
$s_{LOS} = k_2$
No
Yes
$s_{LOS} = 0$
$s_B = 0$
No
Yes
6
(Figure 6d)
$t_{ig} = t_c + \delta t_B$
$i = i_{prev}$
Call Precision State Fxtrapolation Routine
Input: $r_T(t_m)\ [\underline{r}_0],\ \underline{v}_T(t_m)\ [\underline{v}_0],$
$t_m\ [t_0]\ t_{ig}\ [t_F],\ 2\ [s_{pert}],\ 0\ [d]$
Output: $\underline{r}_T(t_{ig})\ [\underline{r}_F],\ \underline{v}_T(t_{ig})[\underline{v}_F]$
Call Precision State Extrapolation Routine
Input: $\underline{r}_P(t_m)\ [\underline{r}_0],\ \underline{v}_P(t_m)[\underline{v}_0]$
$t_m\ [t_0],\ t_{ig}\ [t_F], 2\ [s_{pert}],\ 0\ [d]$
Output: $\underline{r}_P(t_{ig})\ [\underline{r}_F],\ \underline{v}_P(t_{ig})[\underline{v}_F]$
$M_{R-LV} = \begin{pmatrix} -\text{unit}\left\{\left[\underline{v}_P(t_{ig}) \times \underline{r}_P(t_{ig})\right] \times \underline{r}_P(t_{ig})\right\} \\ \text{unit}\left[\underline{v}_P(t_{ig}) \times \underline{r}_P(t_{ig})\right] \\ -\text{unit}\left[\underline{r}_P(t_{ig})\right] \end{pmatrix}$
$M_{R-LOS} = \begin{pmatrix} \text{unit}\left[\underline{r}_T(t_{ig}) - \underline{r}_P(t_{ig})\right] \\ \text{unit}\left[\underline{r}_T(t_{ig}) \times \underline{r}_P(t_{ig})\right] \\ \underline{i}_{x\ LOS} \times \underline{i}_{y\ LOS} \end{pmatrix}$
No
$s_{LAM} = 1$
Yes
4
7

Figure 6b. Terminal Phase Targeting Routine, Detailed Flow Diagram

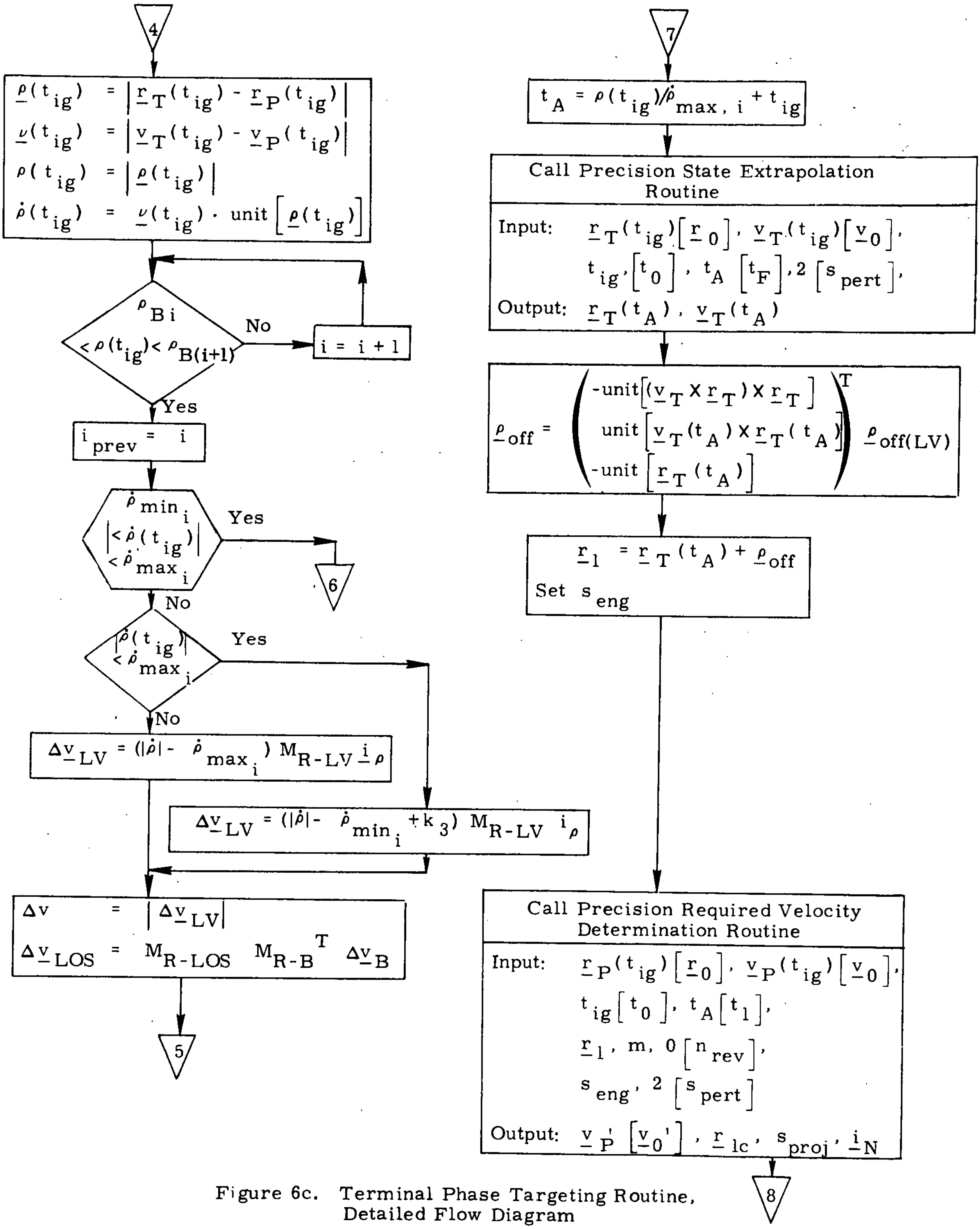

4
$\underline{\rho}(t_{ig}) = \left|\underline{r}_T(t_{ig}) - \underline{r}_P(t_{ig})\right|$
$\underline{\nu}(t_{ig}) = \left|\underline{v}_T(t_{ig}) - \underline{v}_P(t_{ig})\right|$
$\rho(t_{ig}) = \left|\underline{\rho}(t_{ig})\right|$
$\dot{\rho}(t_{ig}) = \underline{\nu}(t_{ig}) \cdot \text{unit}\left[\underline{\rho}(t_{ig})\right]$
$\rho_{Bi} < \rho(t_{ig}) < \rho_{B(i+1)}$
No
$i = i + 1$
Yes
$i_{prev} = i$
$\dot{\rho}_{min_i} < \left|\dot{\rho}(t_{ig})\right| < \dot{\rho}_{max_i}$
Yes
6
No
$\left|\dot{\rho}(t_{ig})\right| < \dot{\rho}_{max_i}$
Yes
No
$\Delta\underline{v}_{LV} = (|\dot{\rho}| - \dot{\rho}_{max_i})\, M_{R-LV}\, \underline{i}_\rho$
$\Delta\underline{v}_{LV} = (|\dot{\rho}| - \dot{\rho}_{min_i} + k_3)\, M_{R-LV}\, \underline{i}_\rho$
$\Delta v = \left|\Delta\underline{v}_{LV}\right|$
$\Delta\underline{v}_{LOS} = M_{R-LOS}\, M_{R-B}^T\, \Delta\underline{v}_B$
5
7
$t_A = \rho(t_{ig})/\dot{\rho}_{max,\,i} + t_{ig}$
Call Precision State Extrapolation Routine
Input: $\underline{r}_T(t_{ig})[\underline{r}_0]$, $\underline{v}_T(t_{ig})[\underline{v}_0]$, $t_{ig}, [t_0]$, $t_A\ [t_F]$, $2\ [s_{pert}]$,
Output: $\underline{r}_T(t_A)$, $\underline{v}_T(t_A)$
$\underline{\rho}_{off} = \begin{pmatrix} -\text{unit}[(\underline{v}_T \times \underline{r}_T) \times \underline{r}_T] \\ \text{unit}[\underline{v}_T(t_A) \times \underline{r}_T(t_A)] \\ -\text{unit}[\underline{r}_T(t_A)] \end{pmatrix}^T \underline{\rho}_{off(LV)}$
$\underline{r}_1 = \underline{r}_T(t_A) + \underline{\rho}_{off}$
Set s_{eng}
Call Precision Required Velocity Determination Routine
Input: $\underline{r}_P(t_{ig})[\underline{r}_0]$, $\underline{v}_P(t_{ig})[\underline{v}_0]$, $t_{ig}[t_0]$, $t_A[t_1]$, $\underline{r}_1$, m, $0\,[n_{rev}]$, s_{eng}, $2\ [s_{pert}]$
Output: $\underline{v}_P'\ [\underline{v}_0']$, $\underline{r}_{lc}$, s_{proj}, $\underline{i}_N$
8

Figure 6c. Terminal Phase Targeting Routine, Detailed Flow Diagram

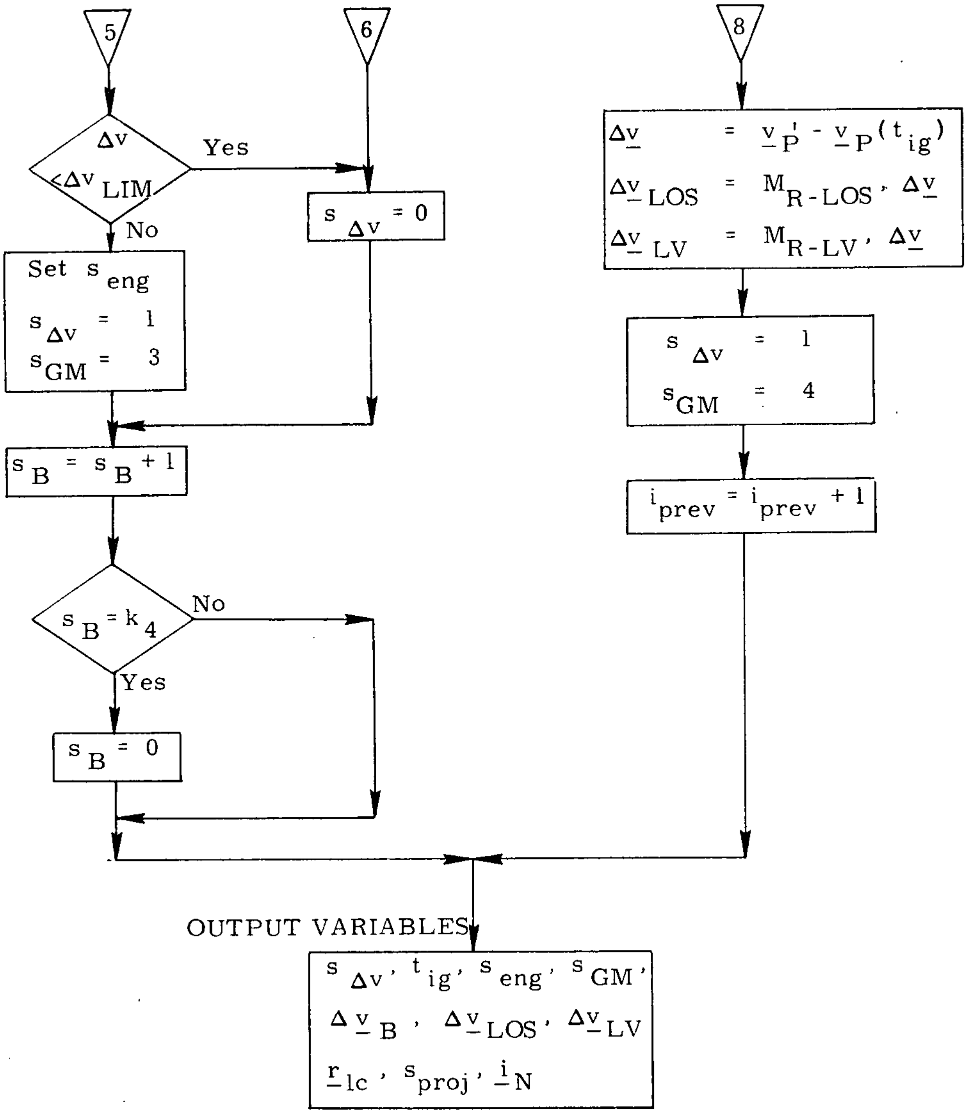

5
6
8
$\Delta v < \Delta v_{LIM}$
Yes
No
Set s_{eng}
$s_{\Delta v} = 1$
$s_{GM} = 3$
$s_{\Delta v} = 0$
$s_B = s_B + 1$
$s_B = k_4$
No
Yes
$s_B = 0$
$\Delta\underline{v} = \underline{v}_P' - \underline{v}_P(t_{ig})$
$\Delta\underline{v}_{LOS} = M_{R-LOS} \cdot \Delta\underline{v}$
$\Delta\underline{v}_{LV} = M_{R-LV} \cdot \Delta\underline{v}$
$s_{\Delta v} = 1$
$s_{GM} = 4$
$i_{prev} = i_{prev} + 1$
OUTPUT VARIABLES
$s_{\Delta v}$, t_{ig}, s_{eng}, s_{GM},
$\Delta\underline{v}_B$, $\Delta\underline{v}_{LOS}$, $\Delta\underline{v}_{LV}$
$\underline{r}_{lc}$, s_{proj}, $\underline{i}_N$

Figure 6d. Terminal Phase Targeting Routine, Detailed Flow Diagram

6. SUPPLEMENTARY INFORMATION

The design of the Rendezvous Terminal Phase Braking Program described in the preceeding sections has been based on the work reported in Reference 1.

The design is such as to incorporate present thinking on several desired options. The first is the method of processing rendezvous sensor data either with the Relative State Updating Function (with the appropriate filter modifications) or using the raw sensor data to determine the necessary relative state parameters.

The second option is to use either the automatic line-of-sight braking scheme or the automatic Lambert braking mode.

These options are indicated in the flow diagrams by the use of appropriate switches (for example s_{LAM}). It is conceivable that these switches may not appear in the final equations if analysis shows that one option is clearly the best for all rendezvous situations.

Studies are presently under way to determine the following:

1. For given baseline sensors, what is the best way of utilizing this sensor data to obtain the necessary relative state parameters ?

2. If it is desired to filter the sensor data, what modification to the Relative State Updating Function is necessary to obtain the desired filter performance; or is the Station-keeping Updating Function satisfactory in this phase ?

3. Which of the two targeting modes is best suited for a particular sensor processing scheme or is a combination of the two modes the best ?

4. What modification of the Powered Flight Guidance Routine and targeting routine are necessary to obtain the desired terminal phase profile and performance ?

REFERENCES

1. Design and Development of Guidance, Navigation and Control Software for the Apollo Applications Program (AAP). "Wet-Workshop" Missions, E-2469, April 1970, M.I.T./DL

2. Muller, E. S., Phillips, R. E., "Relative State Updating", Space Shuttle GN&C Equation Document, No. 6-71, M.I.T./DL

3. Brand, T. J., et al, "Powered Flight Guidance", Space Shuttle GN&C Equation Document, No. 11, (Rev. 1), September 1971, M.I.T. /DL

4. Robertson, W. M., "Precision State and Filter Weighting Matrix Extrapolation," Space Shuttle GN&C Equation Document, No. 4, (Rev. 1), October 1971, M.I.T./DL

5. Brand, T. J., "Precision Required Velocity Determination,", Space Shuttle GN&C Equation Document, No. 13, (Rev. 1) September 1971, M.I.T./DL

C.8

SPACE SHUTTLE

GN&C SOFTWARE EQUATION SUBMITTAL

Software Equation Section: Relative State Updating Submittal No. 20

Function: Compute Relative State of Orbiter with another Satellite

Module No. OG1 Function No. 2,5 (MSC03690)

Submitted by: E. S. Muller, R. E. Phillips Co. MIT No. 6-71

Date: Feb 1971

NASA Contact: J. Suddath Organization: GCD

Approved by Panel III: K. J. Cox Date: 3/10/71

Summary Description: Provides a means of automatically and autonomously improving the estimate of relative state between the Orbiter and another orbiting vehicle. This function may be required in a rendezvous mission or as an orbit navigation mode which utilizes the tracking of satellites.

Shuttle Configuration: Sensors are not specified.

Comments:

(Design Status) ________________

(Verification Status) ________________

Panel Comments:

9.8.2 Relative State Updating

1. INTRODUCTION AND FUNCTIONAL FLOW DIAGRAM

The purpose of the Relative State Updating function is to provide a means of automatically and autonomously improving on-board knowledge of the relative state between the SSV (primary vehicle) and another orbiting vehicle (target vehicle). This knowledge would be required in (a) rendezvous missions as inputs to rendezvous targeting programs to compute maneuvers which effect rendezvous between the primary and target vehicles or (b) orbit navigation modes which utilize tracking of navigation satellites or satellites ejected from the primary vehicle.

Rendezvous navigation sensor data, consisting of measurements of some portion of the relative state, are accepted at discrete "measurement incorporation times". Relative state updating is accomplished at each of these times by sequentially processing the components of the relative state measured by the sensor. A precision extrapolation routine extrapolates the primary and target vehicle state vectors and the filter weighting matrix from one "measurement incorporation time" to the next. A typical measurement incorporation sequence is thus:

(a) Extrapolate primary and target vehicle state vectors and filter weighting matrix to measurement incorporation time (t_m)

(b) Accept set of rendezvous navigation sensor data taken at time = t_m. This will consist of k components of the relative state at t_m given by Q_i (i = 1, 2, ..., k). A measurement code (c_i) is associated with each Q_i to identify the type of measurement taken.

(c) Process Q_1 in the Measurement Incorporation Routine. If more than one component of the relative state is being sensed, process Q_2, Q_3, ..., Q_k sequentially in the Measurement Incorporation Routine.

(a) thru (c) are then repeated for the next measurement incorporation time.

A general flow diagram of this function is presented in Fig. 1. The inputs required by this function are:

1. On-board estimate of primary vehicle state ($\underline{x}_P$) with time tag.
2. On-board estimate of target vehicle state ($\underline{x}_T$) with time tag.
3. Initial filter weighting matrix (W) (not required if computed using Automatic Initialization Routine).
4. A priori sensor measurement variances.
5. Rendezvous sensor measurements.

The output of this function is an updated n-dimensional state ($\underline{x}$) which minimizes the mean squared uncertainty in the estimate of the relative state. This output is available after each measurement incorporation.

The operations shown in Fig. 1, with the exception of precision extrapolation, belong in this function. The Precision Extrapolation Routine is described in another report. The bulk of the equations involved in this function are associated with the Measurement Incorporation Routine. The equations involved in an optional Automatic Initialization Routine (initializes the filter weighting matrix) will also be described. The equations associated with reading the rendezvous navigation sensor will be described in a later report.

9.8.2 Relative State Updating (Con't)

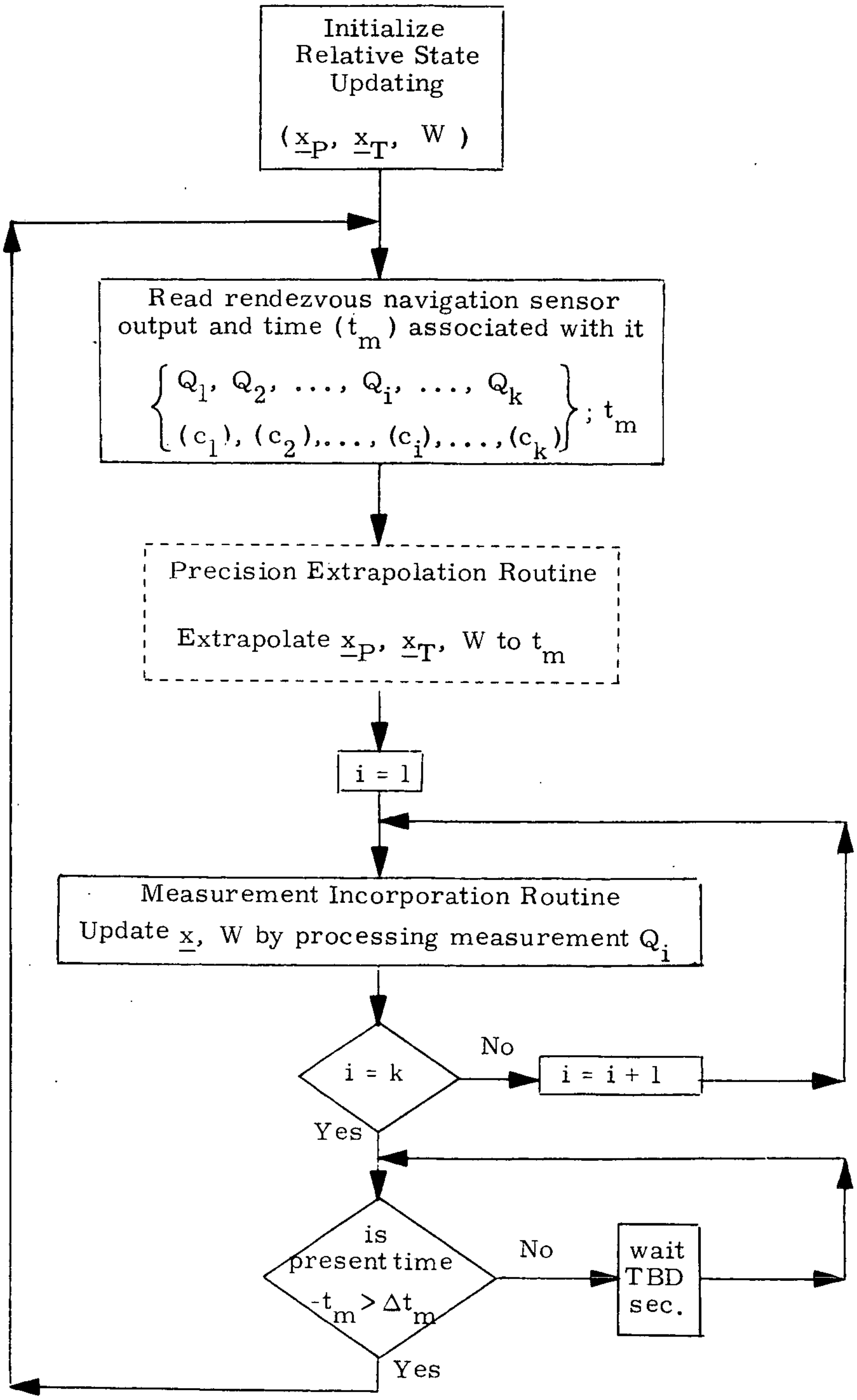

Initialize Relative State Updating
$(\underline{x}_P, \underline{x}_T, W)$
Read rendezvous navigation sensor output and time (t_m) associated with it
$\left\{ \begin{matrix} Q_1, Q_2, \ldots, Q_i, \ldots, Q_k \\ (c_1), (c_2), \ldots, (c_i), \ldots, (c_k) \end{matrix} \right\}; t_m$
Precision Extrapolation Routine
Extrapolate $\underline{x}_P, \underline{x}_T$, W to t_m
i = 1
Measurement Incorporation Routine
Update $\underline{x}$, W by processing measurement Q_i
i = k
No
i = i + 1
Yes
is present time $-t_m > \Delta t_m$
No
wait TBD sec.
Yes

Figure 1 RELATIVE STATE UPDATING FLOW DIAGRAM

NOMENCLATURE

$\underline{b}$ n-dimensional measurement geometry

$\underline{b}_{P0}$, $\underline{b}_{P3}$ 3 dimensional measurement geometry vectors associated with $\underline{r}_P$, $\underline{v}_P$

$\underline{b}_\gamma$ J-dimensional measurement geometry vector associated with $\underline{\gamma}_B$

c_i Measurement code identifying i th measurement at t_m

FIRSTMEAS Initially set to "1" and reset to "0" after first entrance into Measurement Incorporation Routine

FULLTRACK "1" if angles and relative range measurements have been taken prior to final intercept maneuver.

"0" if angles or relative range measurements only have been taken prior to final intercept maneuver.

MANEUVER Initially set to "0". Set to "1" at completion of maneuver

MANNOTRK1 Assumed "no track" time immediately prior to maneuver

MANNOTRK2 Assumed "no track" time immediately following maneuver

MANTM Predicted time of next maneuver (either from preloaded input or previous targeting routine)

$M_{NB\text{-}m}$ Transformation matrix from navigation base axes to rendezvous sensor axes. $M_{NB\text{-}m}$ is fixed according to spacecraft configuration.

$M_{R\text{-}SM}$ Transformation matrix from reference coordinate frame (in which initial state is expressed and computations are performed) to stable member axes. $M_{R\text{-}SM}$ is given from specified platform alignment

$M_{SM\text{-}NB}$ Transformation matrix from stable member axes to navigation base axes on which IMU is mounted. $M_{SM\text{-}NB}$ is determined from IMU gimbal angles

NOTRACKTM Maximum break in tracking threshold - if time of "no track" period exceeds this, W reinitialization is inhibited until after 3 measurement incorporation times

POSTMANWR Initially set to "0". If set to "1", forces W reinitialization prior to first mark after maneuver.

Q_{EST} On-board estimate of measured parameter

Q_i i th measured parameter at t_m

r_P Magnitude of vector $\underline{r}_P$

$\underline{r}_P$ Primary vehicle position vector

$\underline{r}_T$ Target vehicle position vector

$\underline{r}_{TP}$ Relative position vector

$\underline{r}_U$ Position vector found in Automatic Initialization Routine (A.I.R)

RENDWFLAG "0" - W is left as extrapolated value from Precision Integration routine (initially set to "0")

"1" - W is set to pre-loaded value given by W_R

t_m Measurement Incorporation Time

TBEFCOMP Minimum time required prior to a final targeting computation to allow requested W reinitialization to be performed

TPIMAN — Initially set to "0". Set to "1" at completion of final intercept maneuver (TPI)

UNIT ($\underline{r}_P$) — Unit vector ($\underline{r}_P / r_P$)

UPD — = 1 find state of primary vehicle
= 2 find state of target vehicle

$\underline{v}_P$ — Primary vehicle velocity vector

$\underline{v}_T$ — Target vehicle velocity vector

$\underline{v}_{TP}$ — relative velocity vector

VAR — A priori random measurement error variance

$\underline{v}_U$ — Velocity vector found in A.I.R.

W — n x n filter weighting matrix associated with $\underline{x}$

WAIT3TM — Initially set to "0" and reset to "1" in order to inhibit W reinitialization until after 3 measurement inforporation times

W_I — Pre-loaded value of initial filter weighting matrix

W_F — Pre-loaded value to which W is reinitialized

WMAXTM — Maximum threshold value - if time since last W reinitialization exceeds this, a W reinitialization is forced to occur prior to the first mark after the next maneuver

WRTM — Normal threshold value - if time since last W reinitialization exceeds this, a W reinitialization is requested

9.8.2 Relative State Updating (Con't)

$\underline{x}$ n-dimensional state vector

$\underline{x}_P = \begin{pmatrix} \underline{r}_P \\ \underline{v}_P \end{pmatrix}$ 6 dimensional primary vehicle state vector

$\underline{x}_T = \begin{pmatrix} \underline{r}_T \\ \underline{v}_T \end{pmatrix}$ 6 dimensional target vehicle state vector

$\underline{x}_U = \begin{pmatrix} \underline{r}_U \\ \underline{v}_U \end{pmatrix}$ 6 dimensional state vector found in A.I.R.

Δt_m Time increment between measurement incorporation times

$\delta \underline{x}$ n-dimensional navigation update of $\underline{x}$

$\underline{\epsilon}_P$ A priori standard deviation of stable member misalignment

$\underline{\epsilon}_T$ A priori standard deviation of misalignment between sensor measurement frame and navigation base

$\eta_r, \eta_\beta, \eta_\theta$ A priori standard deviation of sensor bias errors in range, gimbal angles β and θ (Fig. 2)

$\underline{\Upsilon}_B$ j dimensional sensor bias vector

$\sigma_r, \sigma_\beta, \sigma_\theta$ A priori standard deviation of sensor random errors in range, β, θ

μ Gravitational constant

2. DESCRIPTION OF EQUATIONS

The recursive navigation equations presented in the Measurement Incorporation section are general with respect to the dimension of the state vector to be updated. These equations are therefore applicable to any one of the following navigated state vectors which is selected for the shuttle relative state updating function. (This selection will ultimately be based on shuttle G& N computer capacity, expected target vehicle state uncertainties, and the error characteristics of shuttle navigation sensors.)

Table I

Possible Navigated States

Navigated State ($\underline{x}$)	Parameters Updated	State dimensions (n)
A. $\underline{x} = \underline{x}_P$ or $\underline{x}_T$	primary vehicle state or target vehicle state	6
B. $\underline{x} = \begin{pmatrix} \underline{x}_P \text{ or } \underline{x}_T \\ \underline{\gamma}_B \end{pmatrix}$	primary or target vehicle state plus j components of sensor bias	6+j
C. $\underline{x} = \begin{pmatrix} \underline{x}_P \\ \underline{x}_T \end{pmatrix}$	primary and target vehicle states	12
D. $\underline{x} = \begin{pmatrix} \underline{x}_P \\ \underline{x}_T \\ \underline{\gamma}_B \end{pmatrix}$	primary and target vehicle states plus j components of sensor bias	12+j

For any of these navigated state vectors, the relative state does not appear directly, but is updated implicitly as a result of the update of either or both vehicle inertial states. Utilizing the state vectors (A or B) results in the Apollo rendezvous navigation filter, whereas either state vector (C or D) results in an optimum rendezvous navigation filter. Specifying the dimension (n) of the navigated state vector automatically specifies the dimension of the measurement geometry vector $\underline{b}$ to be (n), and the filter weighting matrix W to be n x n.

2.1 Measurement Incorporation Routine

As discussed above, this routine is entered k (number of measured components from sensor) times at each measurement incorporation time (t_m). The equations presented below are identical for incorporation of each of these components with the exception of equations for $\underline{b}$, Q_{EST} and VAR which depend on the component incorporated. Equations for $\underline{b}$, Q_{EST}, and VAR are given for typical relative measurement parameters and bias estimation, since the precise parameters will not be known until the rendezvous sensor (s) are selected. The assumed sensor coordinate frame geometry is shown in Fig. 2. (Gimbal limits are assumed to be between ± 90°).

The precedure for computing $\underline{b}$, Q_{EST} and VAR is as follows:

① Compute the relative state ($\underline{x}_R$) from:

$$\underline{x}_R = \begin{pmatrix} \underline{R}_{TP} \\ \underline{V}_{TP} \end{pmatrix} = \underline{x}_T - \underline{x}_P$$

and

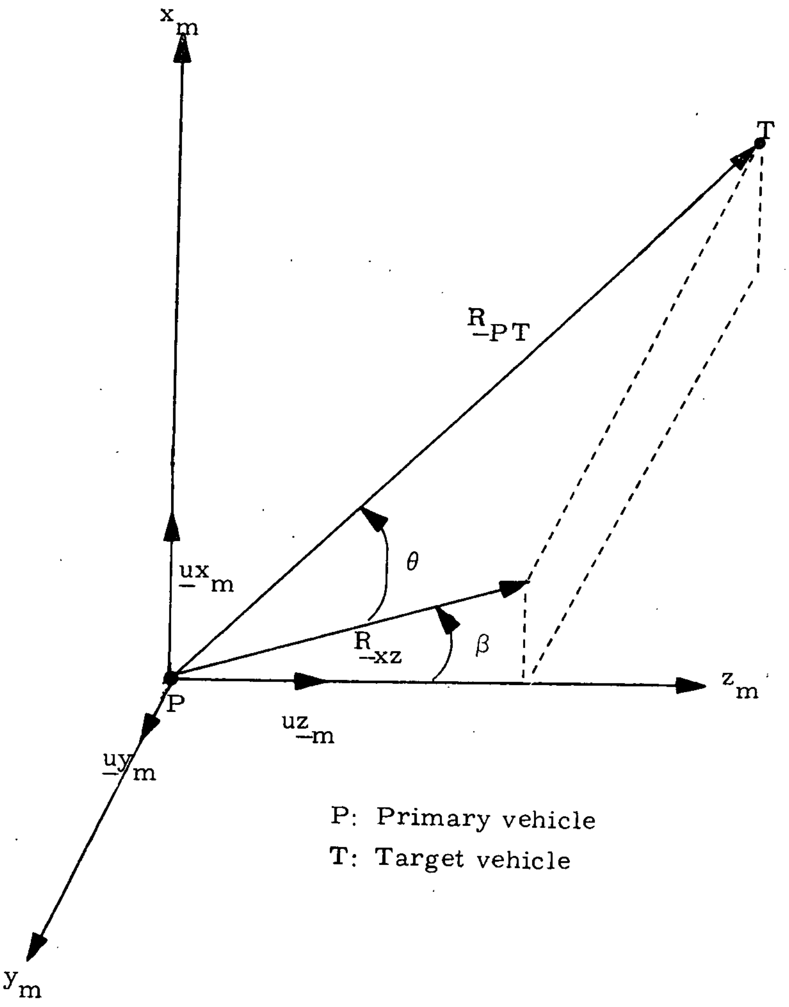


Figure 2 RENDEZVOUS SENSOR COORDINATE FRAME GEOMETRY

$$u\underline{R}_{TP} = \text{UNIT}\,(\underline{R}_{TP})$$

From the measurement code (c_i), compute $\underline{b}$, Q_{EST} and VAR appropriate to this measurement.

For sensor gimbal angle (β, θ) measurements, make the following preliminary computations:

1a Compute the unit vectors of the sensor coordinate frame $u\underline{x}_m$, $u\underline{y}_m$, $u\underline{z}_m$ from:

$$\begin{pmatrix} u\underline{x}_m^{\ T} \\ u\underline{y}_m^{\ T} \\ u\underline{z}_m^{\ T} \end{pmatrix} = M_{NB\text{-}m}\, M_{SM\text{-}NB}\, M_{R\text{-}SM}$$

1b Compute sin (θ), R_{xz} (Fig. 1) from:

$$S = -\,u\underline{R}_{TP} \cdot \underline{uy}_m$$

and

$$R_{xz} = R_{PT}\sqrt{1 - S^2}$$

Computation of $\underline{b}$

Depending on the ultimate selection of the navigated state ($\underline{x}$ of Table I), the vector $\underline{b}$ will take on the following definitions:

If:

$$\underline{x} = \underline{x}_P \ , \ \underline{b} = \underline{b}_P = \begin{pmatrix} \underline{b}_{P0} \\ \underline{b}_{P3} \end{pmatrix}$$

$$\underline{x} = \underline{x}_T \ , \ \underline{b} = -\underline{b}_P$$

$$\underline{x} = \begin{pmatrix} \underline{x}_P \\ \underline{\gamma}_B \end{pmatrix}, \ \underline{b} = \begin{pmatrix} \underline{b}_{P0} \\ \underline{b}_{P3} \\ \underline{b}_{\gamma} \end{pmatrix}$$

$$\underline{x} = \begin{pmatrix} \underline{x}_T \\ \underline{\gamma}_B \end{pmatrix}, \ \underline{b} = \begin{pmatrix} -\underline{b}_{P0} \\ -\underline{b}_{P3} \\ \underline{b}_{\gamma} \end{pmatrix}$$

$$\underline{x} = \begin{pmatrix} \underline{x}_P \\ \underline{x}_T \end{pmatrix}, \ \underline{b} = \begin{pmatrix} \underline{b}_P \\ -\underline{b}_P \end{pmatrix}$$

$$\underline{x} = \begin{pmatrix} \underline{x}_P \\ \underline{x}_T \\ \gamma_B \end{pmatrix}, \ \underline{b} = \begin{pmatrix} \underline{b}_P \\ -\underline{b}_P \\ \underline{b}_{\gamma} \end{pmatrix}$$

2. Compute $\underline{b}_{P0}$ and $\underline{b}_{P3}$ from the appropriate equations in the following table using c_i to identify the type of measurement

Measurement	$\underline{b}_{P0}$	$\underline{b}_{P3}$
Relative Range	$-u\underline{R}_{TP}$	$\underline{0}$
Range Rate	$u\underline{R}_{TP}\times(u\underline{R}_{TP}\times\underline{V}_{TP})/R_{TP}$	$-u\underline{R}_{TP}$
Sensor Angle (β)	$\text{UNIT}(u\underline{R}_{TP}\times u\underline{y}_m)/R_{xz}$	$\underline{0}$
Sensor Angle (θ)	$(u\underline{R}_{TP}\times u\underline{y}_m)\times u\underline{R}_{TP}/R_{xz}$	$\underline{0}$

3. Compute $\underline{b}_{\gamma}$

If $\underline{\gamma}_B$ is included in the navigated state, $\underline{b}$ will be computed based on the selection of bias parameters to be estimated. The following are equations for some possible $\underline{b}_{\gamma}$'s, with c_i used to identify the measurement type

(a) Estimating a single bias (γ_B) in measurement code = j): For measurement (code = c_i)

$$b_\gamma\text{(scalar)} = \begin{cases} 1 & c_i = j \\ 0 & c_i \neq j \end{cases}$$

(b) Estimating bias (γ_B) in m of the total of k measurements, the measurement codes of the m measurements being: 1, 2, . . . , m:

$$\underline{b}\text{ (m-vector)} = \underset{c_i=1}{\begin{pmatrix} 1 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \end{pmatrix}} ; \underset{c_i=2}{\begin{pmatrix} 0 \\ 1 \\ \cdot \\ \cdot \\ \cdot \\ 0 \end{pmatrix}} \cdots \underset{c_i=m}{\begin{pmatrix} 0 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 1 \end{pmatrix}}$$

$$\underline{b}_\gamma = \underline{0} \text{ for } c_i > m$$

(c) Estimating three angles (α_x, α_y, α_z) of the stable member misalignment about x, y, z axes of stable member, i.e.

$$\underline{\gamma}_B = \begin{pmatrix} \alpha_x \\ \alpha_y \\ \alpha_z \end{pmatrix}$$

Measurement	$\underline{b}_\gamma$ (3 dimensional)
Relative Range	$\underline{0}$
Range Rate	$\underline{0}$
Sensor Angle (β)	$R_{PT}/R_{xz}\left[M_{R\text{-}SM}(u\underline{R}_{TP} \times (u\underline{R}_{TP} \times u\underline{y}_m))\right]$
Sensor Angle (θ)	$M_{R\text{-}SM}(\text{UNIT}(u\underline{R}_{TP} \times u\underline{y}_m))$

Computation of Q_{EST}

4 Compute estimate of bias ($\hat{\gamma}$) in measurement form:

$$\hat{\gamma} = \underline{b}_{\gamma} \cdot \hat{\underline{\gamma}}_B$$

($\underline{\gamma}_B$ is initially set to $\underline{0}$ and attains a value after measurement incorporation for a state ($\underline{x}$) which contains $\underline{\gamma}_B$· $\hat{\underline{\gamma}}_B$ will be the corresponding components of $\delta\underline{x}$ in Eq. (11)).

5 Compute Q_{EST} from the appropriate equation identified by the measurement code (c_i):

Measurement	Q_{EST}
Relative Range	$R_{TP} + \hat{\gamma}$
Range Rate	$\underline{V}_{TP} \cdot u\underline{R}_{TP} + \hat{\gamma}$
Sensor Angle (β)	$\text{Tan}^{-1}\left(\dfrac{u\underline{R}_{TP} \cdot u\underline{x}_m}{u\underline{R}_{TP} \cdot u\underline{z}_m}\right) + \hat{\gamma}$
Sensor Angle (θ)	$\text{Sin}^{-1}(s) + \hat{\gamma}$

Computation of VAR

Equations for VAR can not be anticipated as easily as was done for $\underline{b}$ and Q_{EST} since it is so strongly a function of the error model for the particular rendezvous sensor selected for the final configuration. The measurement variance can be a constant or some function of relative range, range rate, etc and it may have a minimum threshold. Consequently, equations for VAR will not be given at this time.

State Vector and Filter Update at Measurement Incorporation Time

The n × n filter weighting matrix (W) is available from one of the following sources:

At the first measurement incorporation:

1. Pre-loaded values based on mission simulations
2. As an output of the Automatic Initialization Routine

Between measurement incorporations at a given t_m:

3. From the computation (below) after a measurement incorporation

At the first measurement incorporation of new t_m:

4. From the Precision Extrapolation Routine*
5. From the Automatic Reinitialization Routine

6 Compute n-dimensional $\underline{z}$ vector for measurement (c_i) from:

$$\underline{z} = W^T \underline{b}$$

7 Compute n-dimensional weighting vector $\underline{\omega}$, from:

$$\underline{\omega} = \frac{1}{\underline{z} \cdot \underline{z} + VAR} W \underline{z}$$

*This routine provides the state and bias portions of W when time invariant biases are modeled. For estimation of biases modeled as time variant, appropriate equations in the Precision Extrapolation Routine will be formulated.

⑧ Compute n-dimensional navigation update of $\underline{x}$ for measured parameter Q_i (code = c_i) from:

$$\delta\underline{x} = \underline{\omega}(Q_i - Q_{EST})$$

⑨ Update $\underline{x}$ by:

$$\underline{x} = \underline{x} + \delta\underline{x}$$

⑩ Update W by:

$$W = W - \underline{\omega}\,\underline{z}^T / \left(1 + \sqrt{\frac{VAR}{\underline{z} \cdot \underline{z} + VAR}}\right)$$

2.2 Automatic Filter Weighting Matrix (W) Reinitialization

If reinitialization of the filter weighting matrix is required (e.g. if navigated states A or B of Table I are utilized), this operation may be accomplished automatically by the Automatic Reinitialization Routine. This routine consists almost entirely of logic statements so that there is no real need to present a description of equations here. Instead, the detailed description of the routine will be provided by the detailed flow diagrams, and a brief description of the approach will be given in this section.

A conservative approach is taken in that W is reinitialized to pre-stored values more often than actually required but not at a time which would violate accepted W matrix reinitialization ground rules. The only exception to this is the case in which not reinitializing will most probably produce a greater performance degradation than a reinitialization. The ground rules which prohibit reinitialization are:

1. No reinitialization unless a minimum time (TBEFCOMP) exists prior to the final targeting computation for a rendezvous maneuver.

2. No reinitialization following a "no tracking" interval greater than NO TRACKTM seconds, until after 3 measurement incorporation times.

The only exception occurs when a maximum time has passed without a reinitialization (WMAXTM) because of (1) or (2). In this case a reinitialization is forced to occur immediately following a rendezvous maneuver (representing a "no track" interval) instead of waiting the required 3 measurement incorporation times as specified by (2).

2.3 Automatic Initialization Routine

2.3.1 Introduction

This routine provides a means for computing an initial filter weighting matrix for recursive navigation which is closely related to the actual errors in the computed relative state. Two position fixes are required. The equations described relate to the problem of finding the inertial state of one vehicle given in the inertial position of the other and the relative position of the two.

The routine might be used if the inertial state of the primary vehicle is poorly known. That is, the estimate of the relative state is so bad that the (linear) recursive navigation filter does not converge. This situation might arise, for instance, when (sensor) acquisition does not occur until the range between the vehicles is of the same order of magnitude as the relative error between them.

2.3.2 Program Input-Output

The required inputs to the routine are two sets of sensor measurements at t_1 and t_2, and two inertial positions at t_1 and t_2. Also required are various assumed values for instrument performance to be used in forming the W matrix.

r_1 β_1 θ_1	sensor measurements at t_1
r_2 β_2 θ_2	sensor measurements at t_2
M_{R-SM} M_{SM-NB} (at t_1 and t_2) M_{NB-m}	rotation matrices
$\underline{r}_{T1}$	known inertial position of target at t_1
$\underline{r}_{T2}$	known inertial position of target at t_2
σ_r σ_β σ_θ	a priori standard deviation of sensor random measurement errors
η_r η_β η_θ	a priori standard deviation of bias in sensor measurement

$$\begin{bmatrix} \alpha_T \\ \beta_T \\ \gamma_T \end{bmatrix} = \underline{\epsilon}_T$$ a priori standard deviation of misalignment between sensor measurement frame and navigation base

$$\begin{bmatrix} \alpha_P \\ \beta_P \\ \gamma_P \end{bmatrix} = \underline{\epsilon}_P$$ a priori standard deviation of stable member misalignment

The output of the program is $\underline{x}_P$ (or $\underline{x}_T$) at t_2 and an n x n W matrix to use in relative state updating.

2.3.3 Description of Equations

The following equations are in two parts, computing the state of the unknown vehicle and computing the related covariance and W matrix. The first set of equations uses two position "fixes" to solve Lambert's problem for the velocity connecting the positions.

Calculation of the State

Let r_1, β_1, θ_1 and r_2, β_2, θ_2 be the measurements made by the sensor at the times t_1 and t_2. Find the cartesian vector $\underline{r}_{TPS\ 1}$ in the sensor frame shown in Fig. 2.

$$r_{TPS\ 1,0} = r_1 \cos \theta_1 \sin \beta_1$$
$$r_{TPS\ 1,1} = - r_1 \sin \theta_1$$
$$r_{TPS\ 1,2} = r_1 \cos \theta_1 \cos \beta_1$$

Using the same relations define $\underline{r}_{TPS\ 2}$. Transform the vector $\underline{r}_{TPS\ 1}$ from sensor frame to reference frame.

$$\underline{R}_{TP\ 1} = M^T_{R\text{-}SM},\ M^T_{SM\text{-}NB\ 1}\ M^T_{NB\text{-}m}\ \underline{r}_{TPS\ 1}$$

Similarly define the vector $\underline{R}_{TP\ 2}$ from (r_2, β_2, θ_2). Depending on the value of logic switch, UPD, extrapolate either the primary or target vehicle to the times of the two fixes t_1 and t_2. Using these two inertial positions the two relative positions, and the time interval $\Delta t = t_2 - t_1$ find the velocity $\underline{v}_{U\ 2}$ at t_2 via the Lambert Routine. The six-dimensional state vector $\underline{x}_{U\ 2}$ at t_2 is:

$$\underline{x}_{U\ 2} = \begin{bmatrix} \underline{r}_{U\ 2} \\ \\ \underline{v}_{U\ 2} \end{bmatrix}$$

From two position measurements it is impossible to estimate any bias, so those components, if included in the state, are set to zero.

Calculation of the W-Matrix

In rendezvous navigation it is the relative state which is measured and used to update either (or both) of the inertial vectors. Associated with the relative state is the relative covariance matrix. As an example the W matrix for a 9-dimensional state including constant sensor bias is computed.

The error in the relative state is due to errors in the sensor measurements r, β, and θ and to errors in the transformation matrices, $M_{R\text{-}SM}$, $M_{SM\text{-}NB}$, $M_{NB\text{-}m}$. The measured quantities r_m, β_m, and θ_m include noise $\underline{\sigma}$ and bias $\underline{\eta}$.

$$\underline{\sigma} \equiv (\sigma_r, \sigma_\beta, \sigma_\theta)$$
$$\underline{\eta} \equiv (\eta_r, \eta_\beta, \eta_\theta)$$

Errors in the transformation matrices are due to stable member misalignment and structural deformation between the sensor and the navigation base. These errors in the two matrices $M_{R\text{-}SM}$ and $M_{NB\text{-}m}$ are expressed as $\underline{\epsilon}_P$ and $\underline{\epsilon}_T$. These pseudo-vectors represent one standard deviation small rotations about three othogonal axes. From the values of $\underline{\sigma}$, $\underline{\eta}$, $\underline{\epsilon}_T$ and $\underline{\epsilon}_P$ two diagonal 9 x 9 matrices SIG and GAM are formed. It should be noted that η_r^2, η_β^2 and η^2_θ appear in SIG if each of those components of bias is to be estimated (as is done here), otherwise they appear in GAM.

Combinations of several 3 x 3 partial derivative matrices make up a 9 x 9 matrix relating state error to the matrix SIG. Those component matrices will now be computed.

The partial derivative matrix of relative position error in the sensor frame due to error in r, β and θ, $DRDM_S$, is computed by simply taking the necessary derivatives of the geometric relations:

$$r_{k,0} = r_k \cos\theta_k \sin\beta_k$$
$$r_{k,1} = - r_k \sin\theta_k$$
$$r_{k,2} = r_k \cos\theta_k \cos\beta_k$$

Combined with transformation matrices the partial derivative matrices allow the partial derivative matrix of relative position error in the inertial frame to be written:

$$DRDM_k = M^T_{R\text{-}SM} \, M^T_{SM\text{-}NB\,k} \, M^T_{NB\text{-}m} \, DRDM_{S\,k}$$

The dependence of the velocity deviations $\delta \underline{v}_{U\,2}$ on the two position errors must be computed. The two matrices $DVDR_2$ and $DVDR_1$ are derivatives of Lambert's solution for the velocity at the second point. They may be computed from values of semi-major axis, $1/\alpha$, eccentric anomalies, E_k, and S and C (Battin's special transcendental functions) found in the Lambert Routine.

α = reciprocal of semi-major axis

x = $(E_1 - E_2)/\sqrt{\alpha}$ (E = eccentric anomaly)

y = x^2/c

$$\left.\begin{array}{l} S(\alpha x^2) \\ C(\alpha x^2) \end{array}\right\}$$ Battin's special transcendental functions

Using these variables and the following definitions proceed:

r_1 = $|\underline{r}_1|$

r_2 = $|\underline{r}_2|$

$\hat{r}_1$ = UNIT $(\underline{r}_1)$

$\hat{r}_2$ = UNIT $(\underline{r}_2)$

q = $\sqrt{2 - \alpha x^2 c}$

D_S = $(c - 3S)/(2\alpha x^2)$

D_C = $(1 - \alpha x^2 S - 2C)/(2\alpha x^2)$

D_Q = $-\sin(\sqrt{\alpha x^2})/(4 q \sqrt{\alpha x^2})$

Q_d = $(3 x S y D_C)/(2C^2) - x^3 D_S$

F = $\sqrt{\mu / y}/\alpha$

H = $y/r_1 - 1$

$$
\begin{array}{lcl}
G & = & F\ H \\
Y_c & = & F / r_1 - G / 2y \\
A_c & = & - G / \alpha \\
Q_y & = & \alpha / (2\sqrt{y}) + 3\ S\ x / (2C) \\
D & = & Q_d + Q_y\ \alpha\ D_Q \\
\nabla_{\underline{1}}\alpha & = & (\hat{r}_1 + \hat{r}_2)\ r_2 / 2\alpha \\
\nabla_{\underline{2}}\alpha & = & (\hat{r}_1 + \hat{r}_2)\ r_1 / 2\alpha \\
\nabla_{\underline{1}}\alpha x^2 & = & ((\sqrt{y} - Q_y q)\ \nabla_{\underline{1}}\alpha + Q_y\ \hat{r}_1) / D \\
\nabla_{\underline{2}}\alpha x^2 & = & ((\sqrt{y} - Q_y q)\ \nabla_2\alpha + Q_y\ \hat{r}_2) / D \\
\nabla_{\underline{1}} y & = & \hat{r}_1 - q\ \nabla_{\underline{1}}\alpha - \alpha\ D_Q\ \nabla_{\underline{1}}\alpha x^2 \\
\nabla_2 y & = & \hat{r}_2 - q\ \nabla_{\underline{2}}\alpha - \alpha\ D_Q\ \nabla_2\alpha x^2 \\
\nabla_{\underline{1}} F & = & - F\ (\nabla_{\underline{1}}\alpha / \alpha + \nabla_{\underline{1}} y / 2y) \\
\nabla_{\underline{2}} F & = & - F\ (\nabla_{\underline{2}}\alpha / \alpha + \nabla_{\underline{2}} y / 2y) \\
\nabla_{\underline{2}} G & = & Y_c\ \nabla_{\underline{2}} y + A_c\ \nabla_{\underline{2}}\ \alpha \\
\nabla_{\underline{1}} G & = & Y_c\ \nabla_{\underline{1}} y + A_c\ \nabla_{\underline{1}}\alpha - F\ y\ \hat{r}_1 / r_1^2 \\
DVDR_2 & = & G\ I + \hat{r}_2\ \nabla_{\underline{1}} F^T + \hat{r}_1\ \nabla_{\underline{1}} G^T \\
DVDR_1 & = & F\ I + \hat{r}_2\ \nabla_{\underline{2}} F^T + \hat{r}_1\ \nabla_{\underline{2}} G^T
\end{array}
$$

(I is the 3 dimensional identity matrix)

Combined with $DRDM_1$ and $DRDM_2$ the above matrices yield the 9 x 9 partial derivative matrix relating the state to the matrix of sensor random and bias errors.

$$\begin{aligned} DVDM_1 &= DVDR_1 \cdot DRDM_1 \\ DVDM_2 &= DVDR_2 \, DRDM_2 \\ DVDE &= DVDM_1 + DVDM_2 \end{aligned}$$

$$\underset{(9 \times 9)}{DSDS} = \begin{bmatrix} DVDM_2 & 0 & DRDM_2 \\ DVDM_2 & DVDM_1 & DVDE \\ 0 & 0 & I \end{bmatrix}$$

A second 9 x 9 partial matrix relates state errors to unestimated sensor bias and the two misalignments. The additional needed 3 x 3 component matrices are computed now.

The matrix $DRDE_{TS\,1}$ relates position error at the first "fix" in the sensor frame to misalignment between sensor and navigation base

$$DRDE_{TS\,1} = \begin{bmatrix} 0 & r_{TPS\,k,2} - r_{TPS\,k,1} \\ -r_{TPS\,k,2} & 0 & r_{TPS\,k,0} \\ r_{TPS\,k,1} & -r_{TPS\,k,0} & 0 \end{bmatrix}$$

This matrix rotated into the reference frame is:

$$DRDE_{T\,1} = M^T_{R\text{-}SM} \, M^T_{SM\text{-}NB\,1} \, M^T_{NB\text{-}m} \, DRDE_{TS1}$$

In the same way $DRDE_{T\,2}$ is computed.

The matrices relating stable member misalignment to error in position in the reference frame $DRDE_P$ is computed in the same way. It consists of a matrix composed of elements of the relative position vector in the reference frame $\underline{r}_{TP}$.

Using the chain rule allows the computation of the matrices relating velocity to the two misalignments:

$$DVDE_T = DVDR_2 \, DRDE_{T\,2} + DVDR_1 \, DRDE_{T\,1}$$

$$DVDE_P = DVDR_2 \, DRDE_{P\,2} + DVDR_1 \, DRDE_{P\,1}$$

The dependence of estimated bias in r, β and θ on the two misalignments is given by the following two matrices.

$$DBDE_T = DRDM_{S\,1}^{-1} \, DRDE_{TS\,1}$$

$$DBDE_P = DRDM_{S\,1}^{-1} \, M_{NB\text{-}m} \, M_{SM\text{-}NB\,1} \, M_{R\text{-}SM} \, DRDE_{TS\,1}$$

The complete 9 x 9 partial matrix is thus:

$$DSDG = \begin{bmatrix} DRDM_2 & DRDE_{T\,2} & DRDE_{P\,2} \\ DVDE & DVDE_T & DVDE_P \\ I & DBDE_T & DBDE_P \end{bmatrix}$$

The covariance matrix of errors in the relative state in the reference frame is:

$$\overline{\begin{bmatrix}\delta r\\ \delta v\\ \delta b\end{bmatrix}\begin{bmatrix}\delta r & \delta v & \delta b\end{bmatrix}} = \text{DSDS SIG DSDS}^T + \text{DSDG GAM DSDG}^T$$

The W matrix can be found from the above covariance matrix by forming a diagonal matrix E_c consisting of the square roots of the diagonalized covariance matrix. If the rows of the matrix RV are eigenvectors of COV; that is RV is defined to be:

$$\begin{bmatrix} E^2_{c_{11}} & & \\ & E^2_{c_{22}} & \\ & & \text{etc} \end{bmatrix} = \text{RV COV RV}^T$$

The W matrix is then:

$$W = \text{RV}^T \begin{bmatrix} E_{c_{11}} & & \\ & E_{c_{22}} & \\ & & \text{etc} \end{bmatrix}$$

(Note: The above indicates symbolically the definition of W but the actual routine to compute W may or may not use the above steps). The vector state $\left[\underline{r}_{U2}, \underline{v}_{U2}\right]$, the time t_2 and the relative W matrix are returned to the calling program.

3. DETAILED FLOW DIAGRAMS

This section contains detailed flow diagrams for the Automatic Initialization Routine and Measurement Incorporation Routine of the Co-orbiting Vehicle Navigation Module. A nine dimensional W-matrix is computed. The three adjoined elements are for constant sensor bias in r, β, and θ. These particular biases were chosen only as an example.

Two routines used are not yet documented: the Lambert Routine and the Eigenvalue Routine.

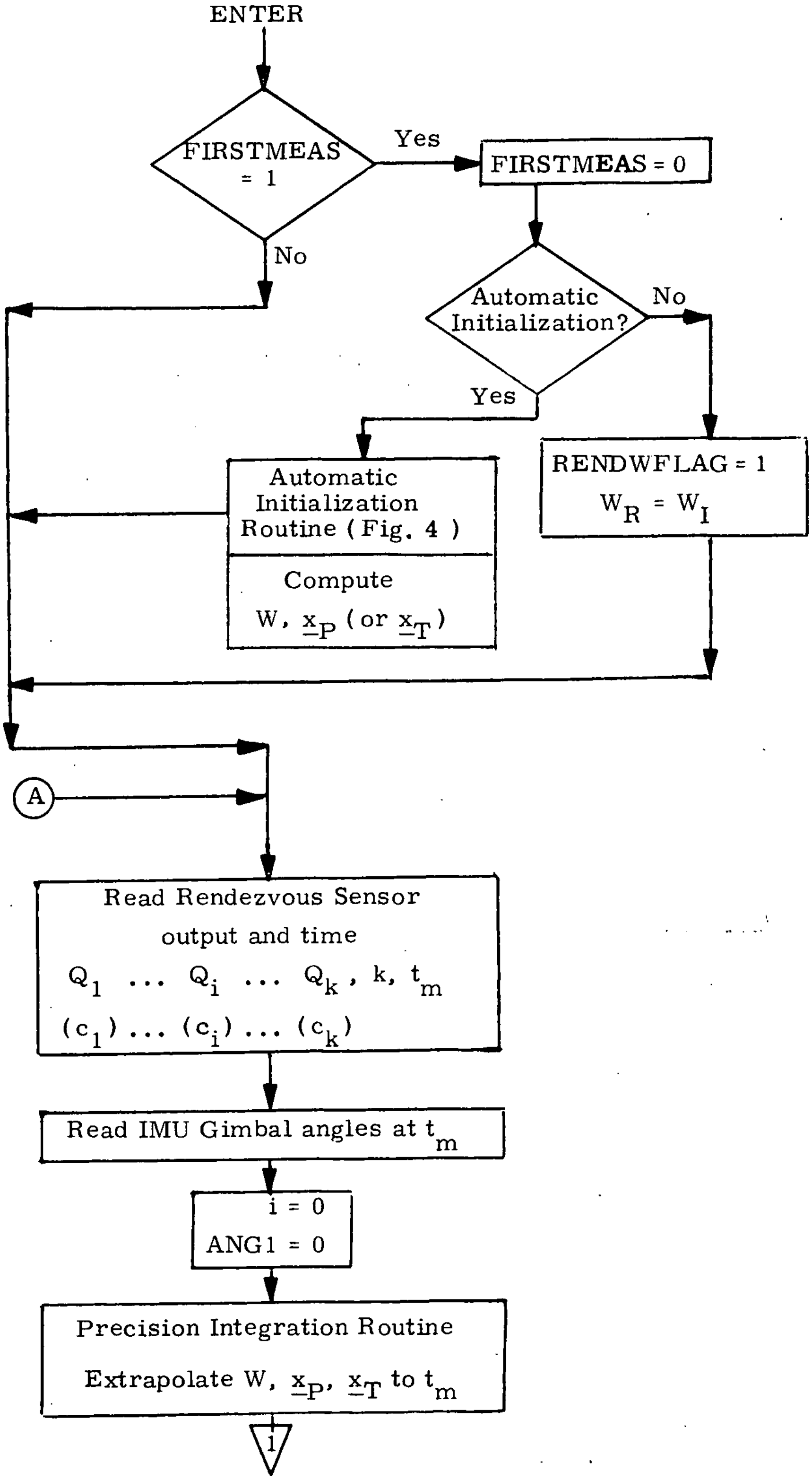

ENTER
FIRSTMEAS = 1
Yes
No
FIRSTMEAS = 0
Automatic Initialization?
No
Yes
Automatic Initialization Routine (Fig. 4)
Compute W, $\underline{x}_P$ (or $\underline{x}_T$)
RENDWFLAG = 1
$W_R = W_I$
A
Read Rendezvous Sensor output and time
Q_1 ... Q_i ... Q_k, k, t_m
(c_1) ... (c_i) ... (c_k)
Read IMU Gimbal angles at t_m
i = 0
ANG1 = 0
Precision Integration Routine
Extrapolate W, $\underline{x}_P$, $\underline{x}_T$ to t_m
1

Figure 3a DETAILED FLOW DIAGRAM, MEASUREMENT INCORPORATION ROUTINE

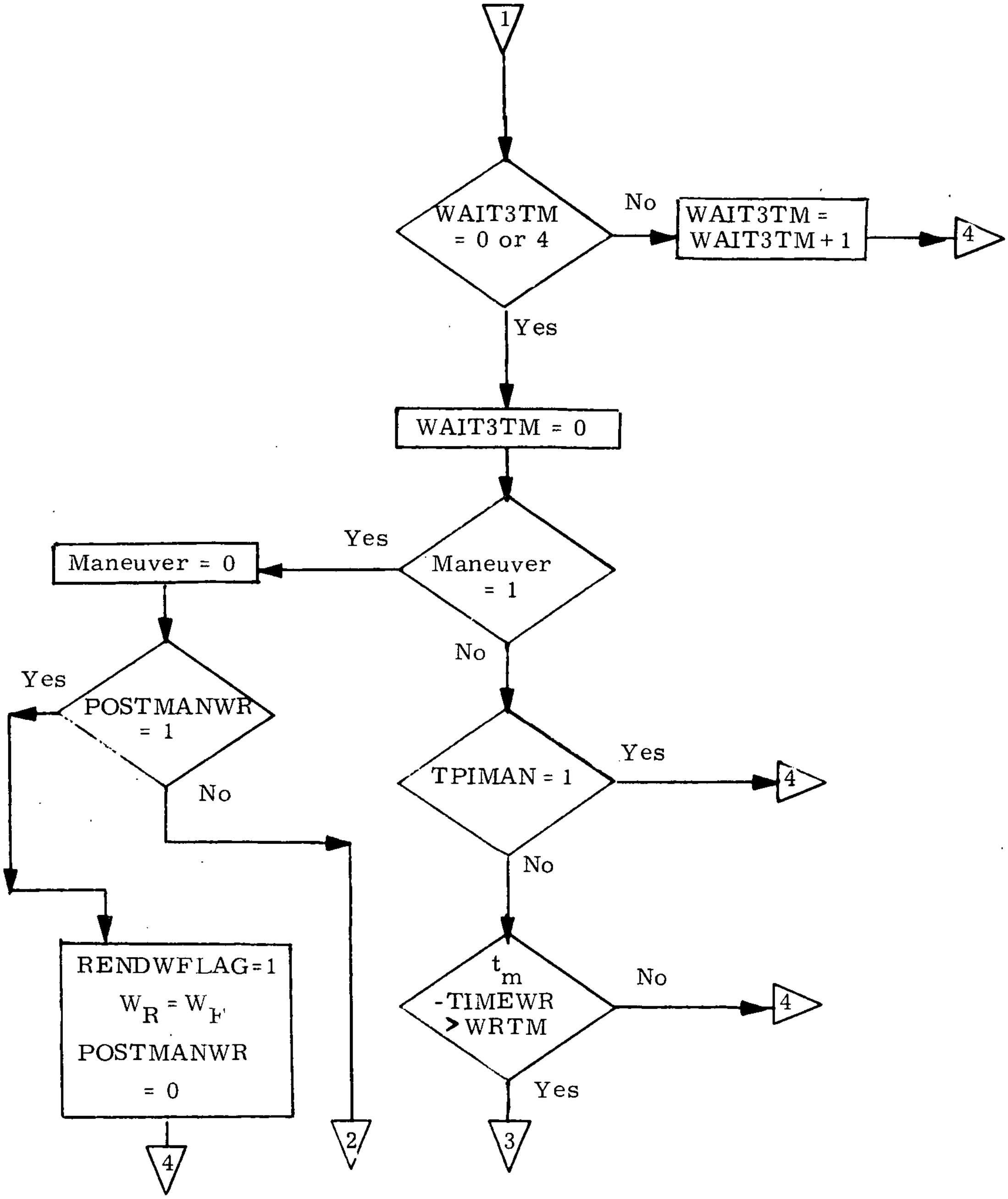


Figure 3b DETAILED FLOW DIAGRAM, MEASUREMENT INCORPORATION ROUTINE

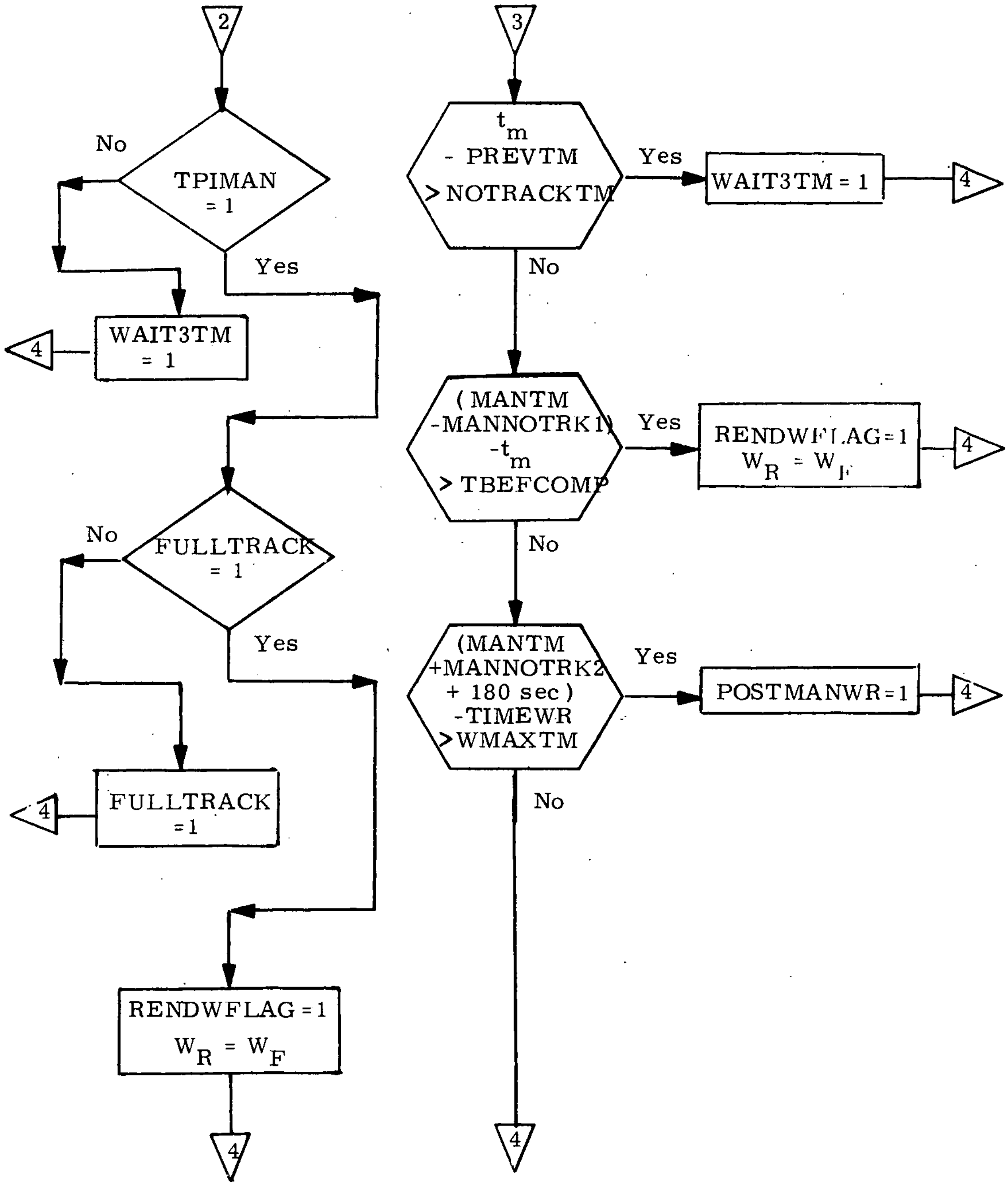


Figure 3c DETAILED FLOW DIAGRAM, MEASUREMENT INCORPORATION ROUTINE

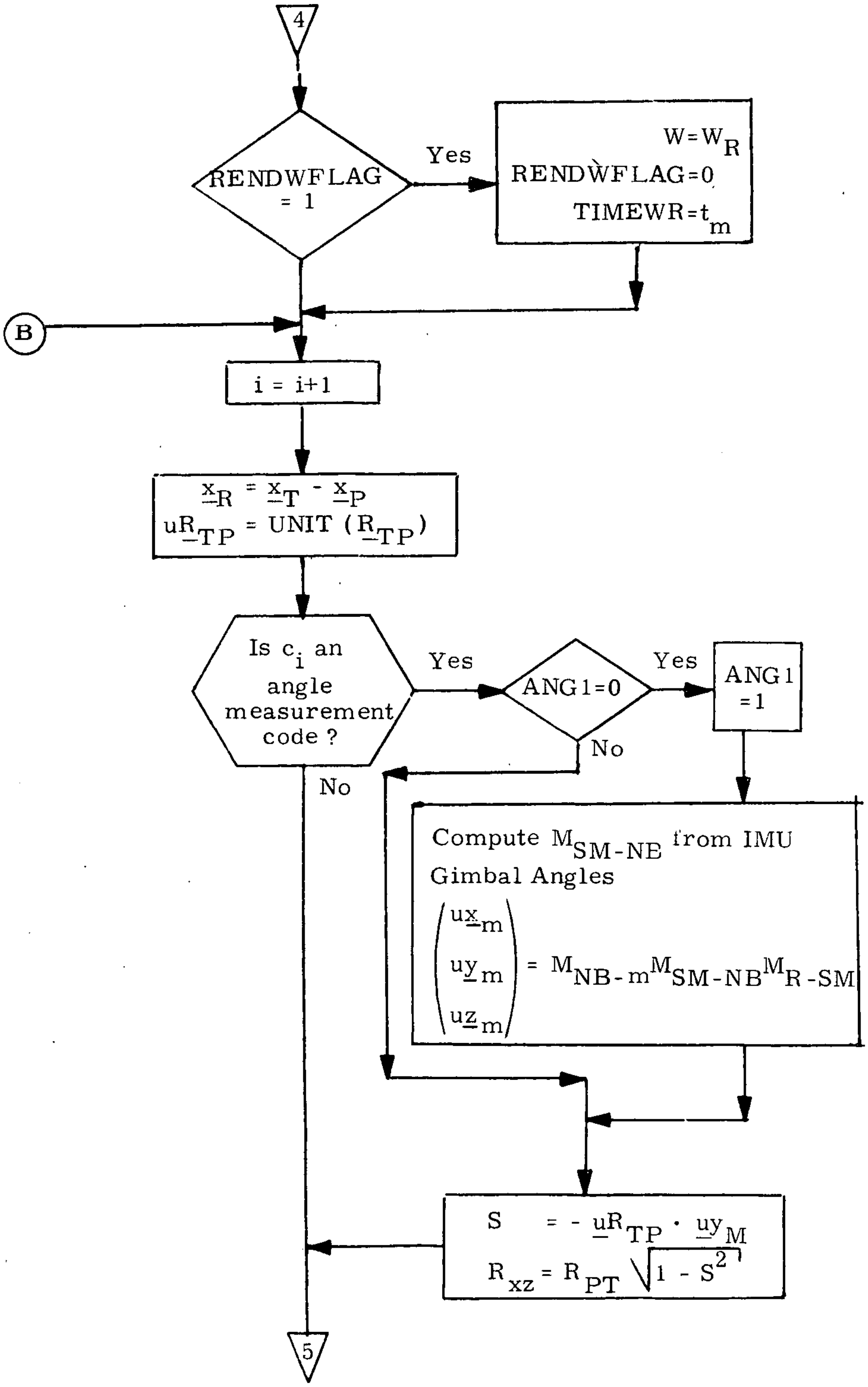


Figure 3d DETAILED FLOW DIAGRAM,MEASUREMENT INCORPORATION ROUTINE

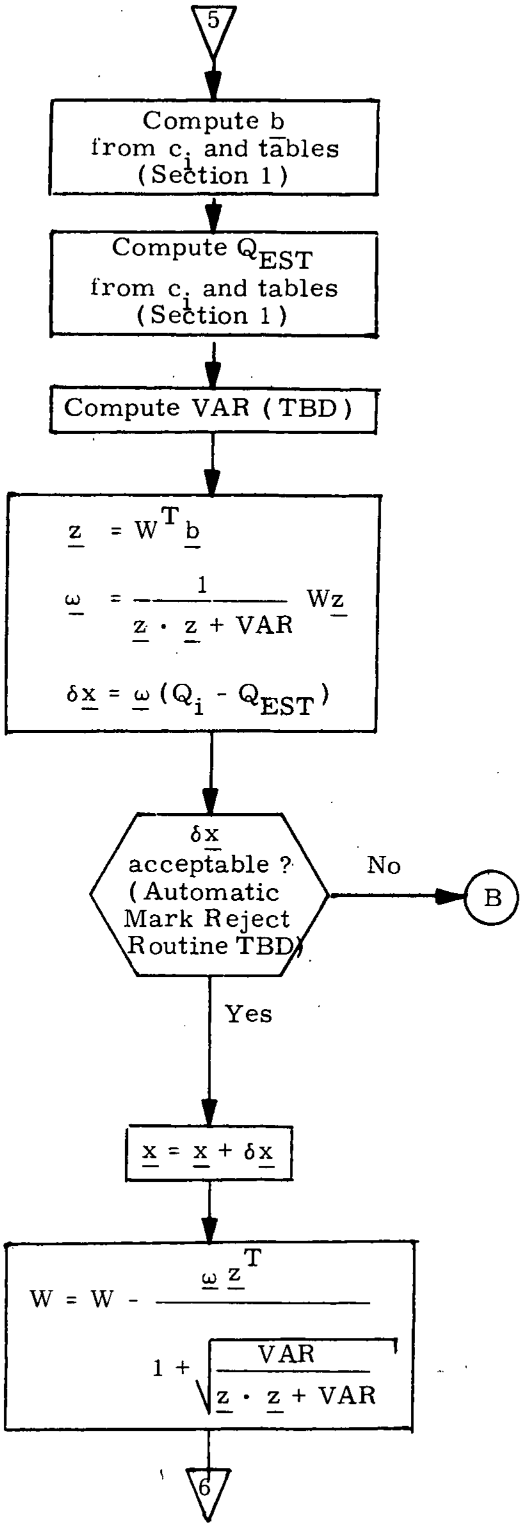

5
Compute b from c_i and tables (Section 1)
Compute Q_{EST} from c_i and tables (Section 1)
Compute VAR (TBD)
$\underline{z} = W^T \underline{b}$
$\underline{\omega} = \frac{1}{\underline{z} \cdot \underline{z} + VAR} W\underline{z}$
$\delta\underline{x} = \underline{\omega} (Q_i - Q_{EST})$
$\delta\underline{x}$ acceptable ? (Automatic Mark Reject Routine TBD)
No
B
Yes
$\underline{x} = \underline{x} + \delta\underline{x}$
$W = W - \frac{\underline{\omega}\,\underline{z}^T}{1 + \sqrt{\frac{VAR}{\underline{z} \cdot \underline{z} + VAR}}}$
6

Figure 3e DETAILED FLOW DIAGRAM, MEASUREMENT INCORPORATION ROUTINE

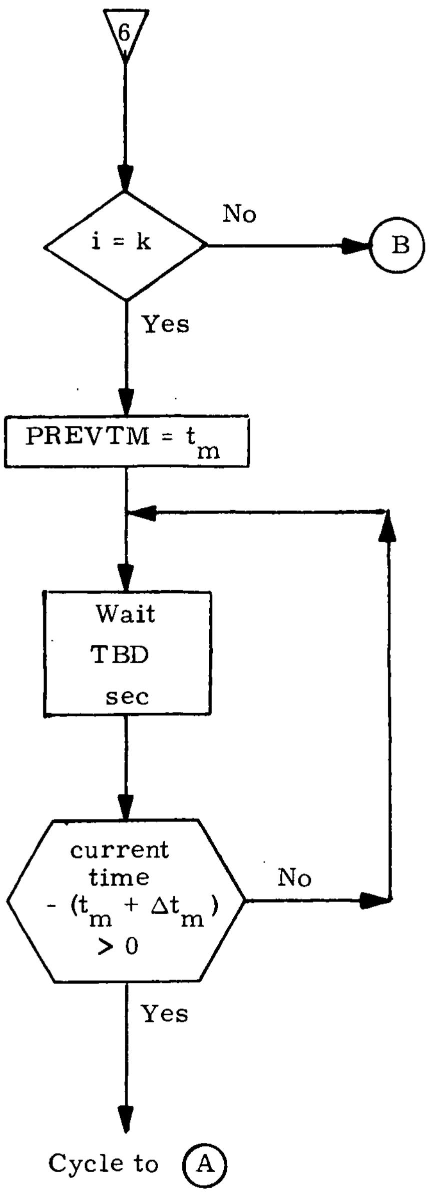


Figure 3f DETAILED FLOW DIAGRAM,MEASUREMENT INCORPORATION ROUTINE

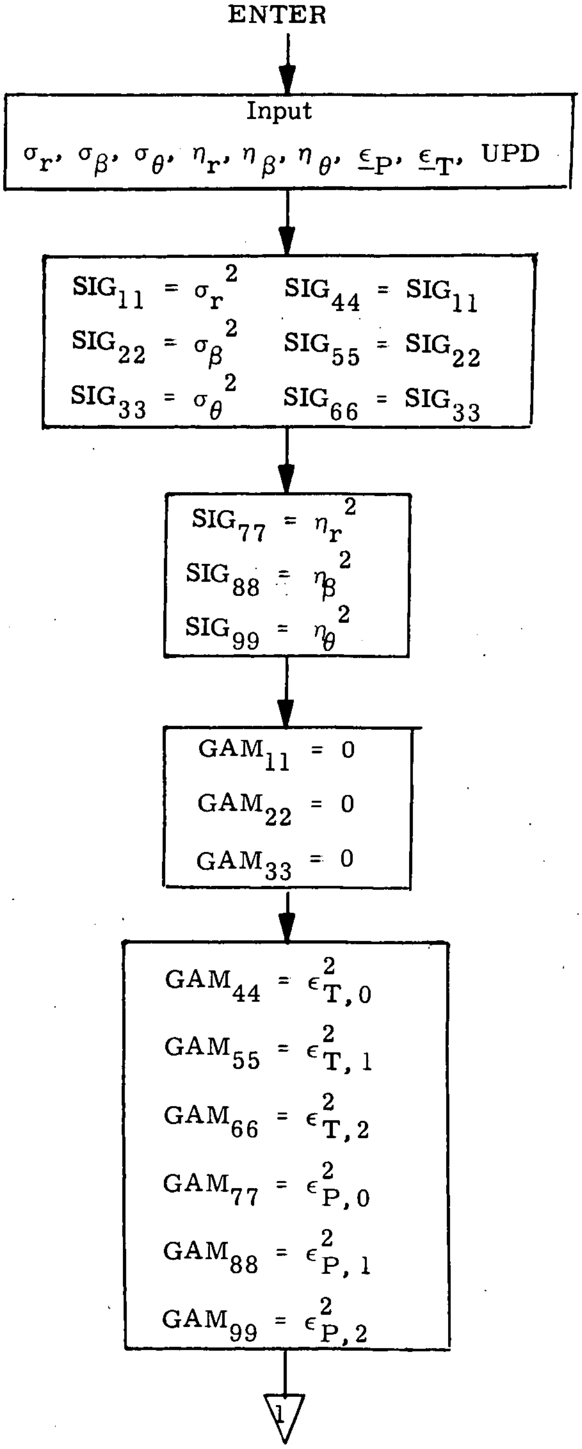

ENTER
Input
$\sigma_r, \sigma_\beta, \sigma_\theta, \eta_r, \eta_\beta, \eta_\theta, \underline{\epsilon}_P, \underline{\epsilon}_T$, UPD
$SIG_{11} = \sigma_r^2$
$SIG_{44} = SIG_{11}$
$SIG_{22} = \sigma_\beta^2$
$SIG_{55} = SIG_{22}$
$SIG_{33} = \sigma_\theta^2$
$SIG_{66} = SIG_{33}$
$SIG_{77} = \eta_r^2$
$SIG_{88} = \eta_\beta^2$
$SIG_{99} = \eta_\theta^2$
$GAM_{11} = 0$
$GAM_{22} = 0$
$GAM_{33} = 0$
$GAM_{44} = \epsilon_{T,0}^2$
$GAM_{55} = \epsilon_{T,1}^2$
$GAM_{66} = \epsilon_{T,2}^2$
$GAM_{77} = \epsilon_{P,0}^2$
$GAM_{88} = \epsilon_{P,1}^2$
$GAM_{99} = \epsilon_{P,2}^2$
1

Figure 4a DETAILED FLOW DIAGRAM,AUTOMATIC INITIALIZATION ROUTINE

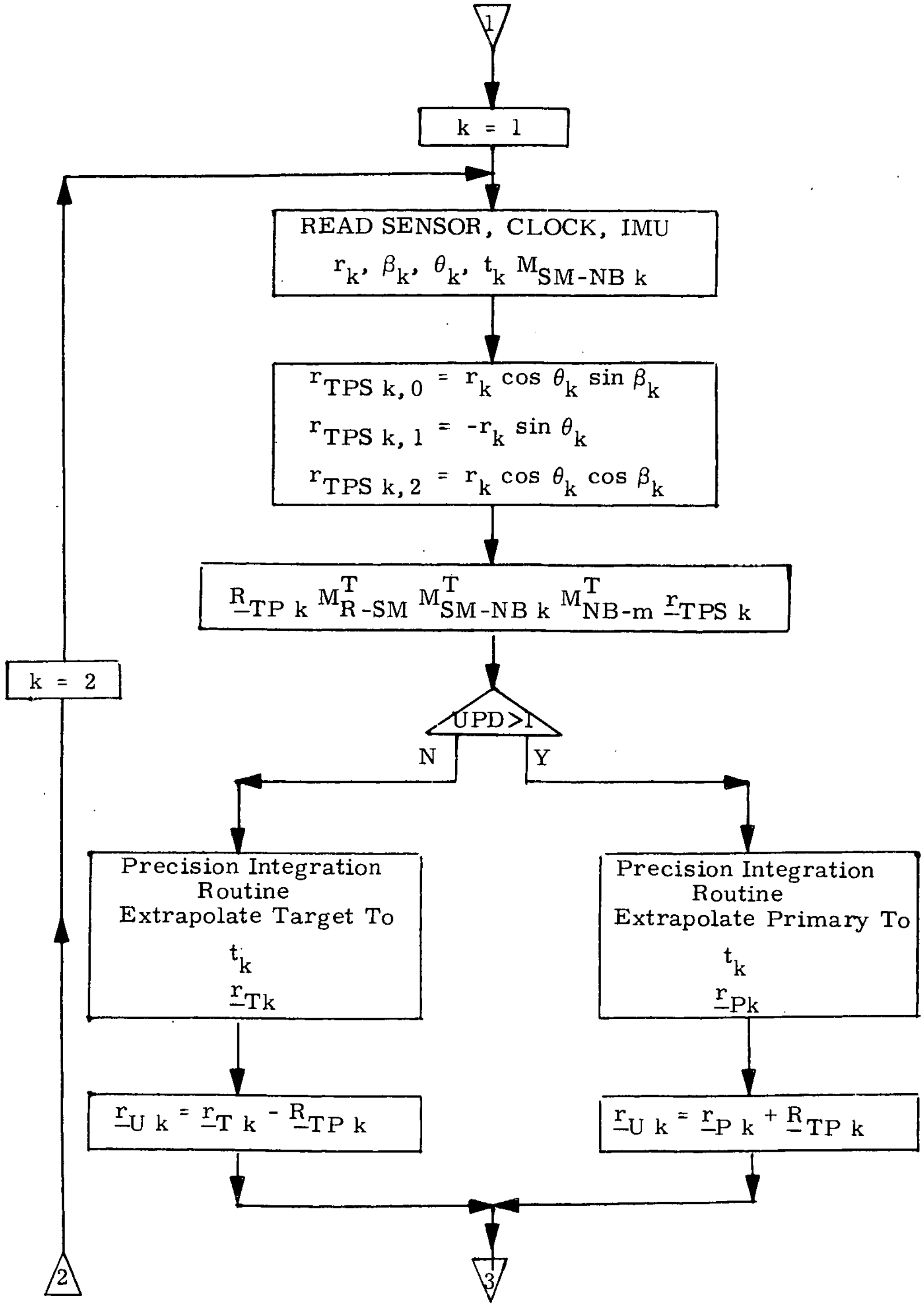

1
k = 1
READ SENSOR, CLOCK, IMU
r_k, β_k, θ_k, t_k $M_{SM-NB\ k}$
$r_{TPS\ k,0} = r_k \cos\theta_k \sin\beta_k$
$r_{TPS\ k,1} = -r_k \sin\theta_k$
$r_{TPS\ k,2} = r_k \cos\theta_k \cos\beta_k$
$\underline{R}_{TP\ k}\ M^T_{R-SM}\ M^T_{SM-NB\ k}\ M^T_{NB-m}\ \underline{r}_{TPS\ k}$
UPD>1
N
Y
Precision Integration Routine Extrapolate Target To
t_k
$\underline{r}_{Tk}$
Precision Integration Routine Extrapolate Primary To
t_k
$\underline{r}_{Pk}$
$\underline{r}_{U\ k} = \underline{r}_{T\ k} - \underline{R}_{TP\ k}$
$\underline{r}_{U\ k} = \underline{r}_{P\ k} + \underline{R}_{TP\ k}$
k = 2
2
3

Figure 4b DETAILED FLOW DIAGRAM,AUTOMATIC INITIALIZATION ROUTINE

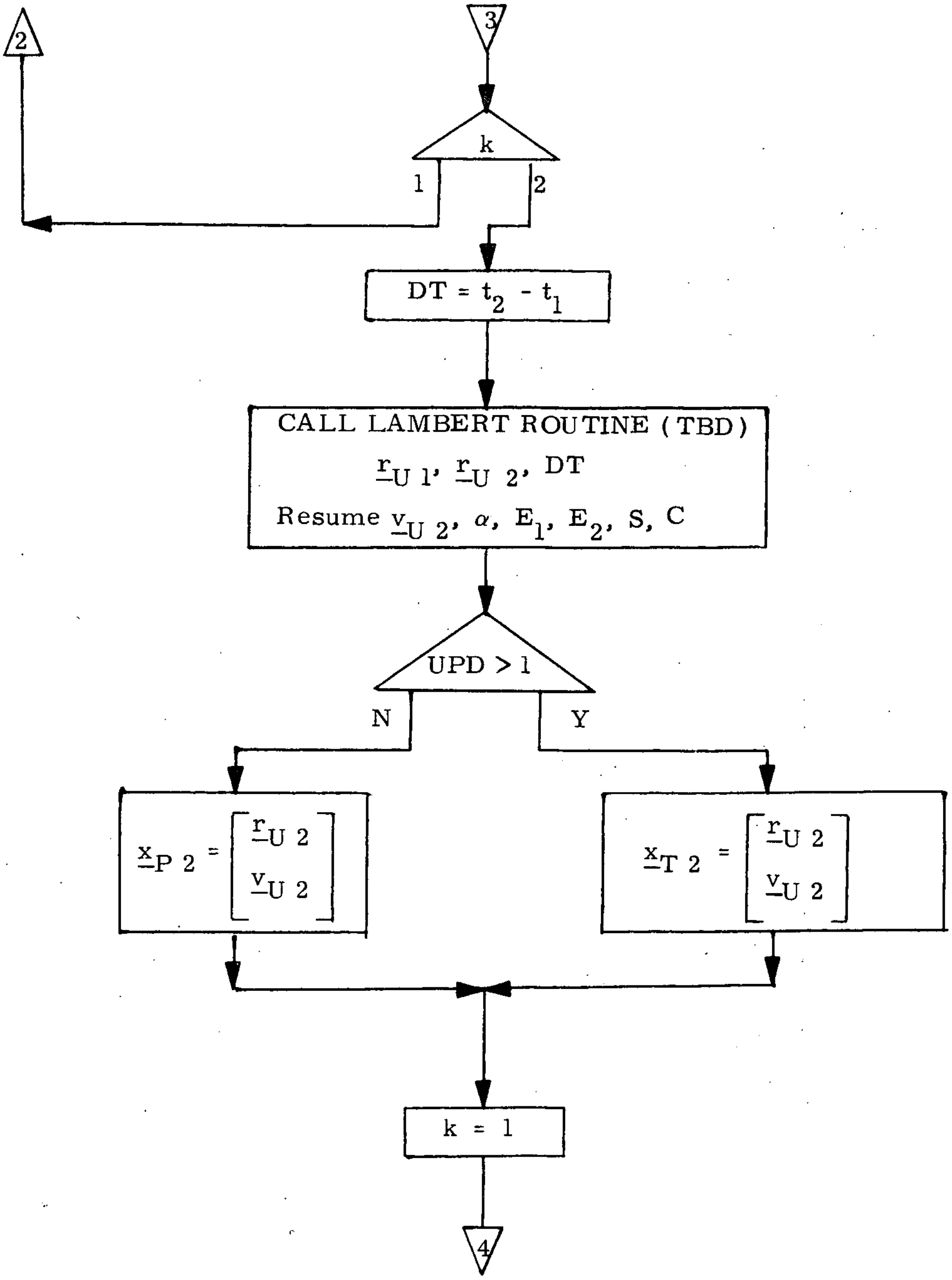
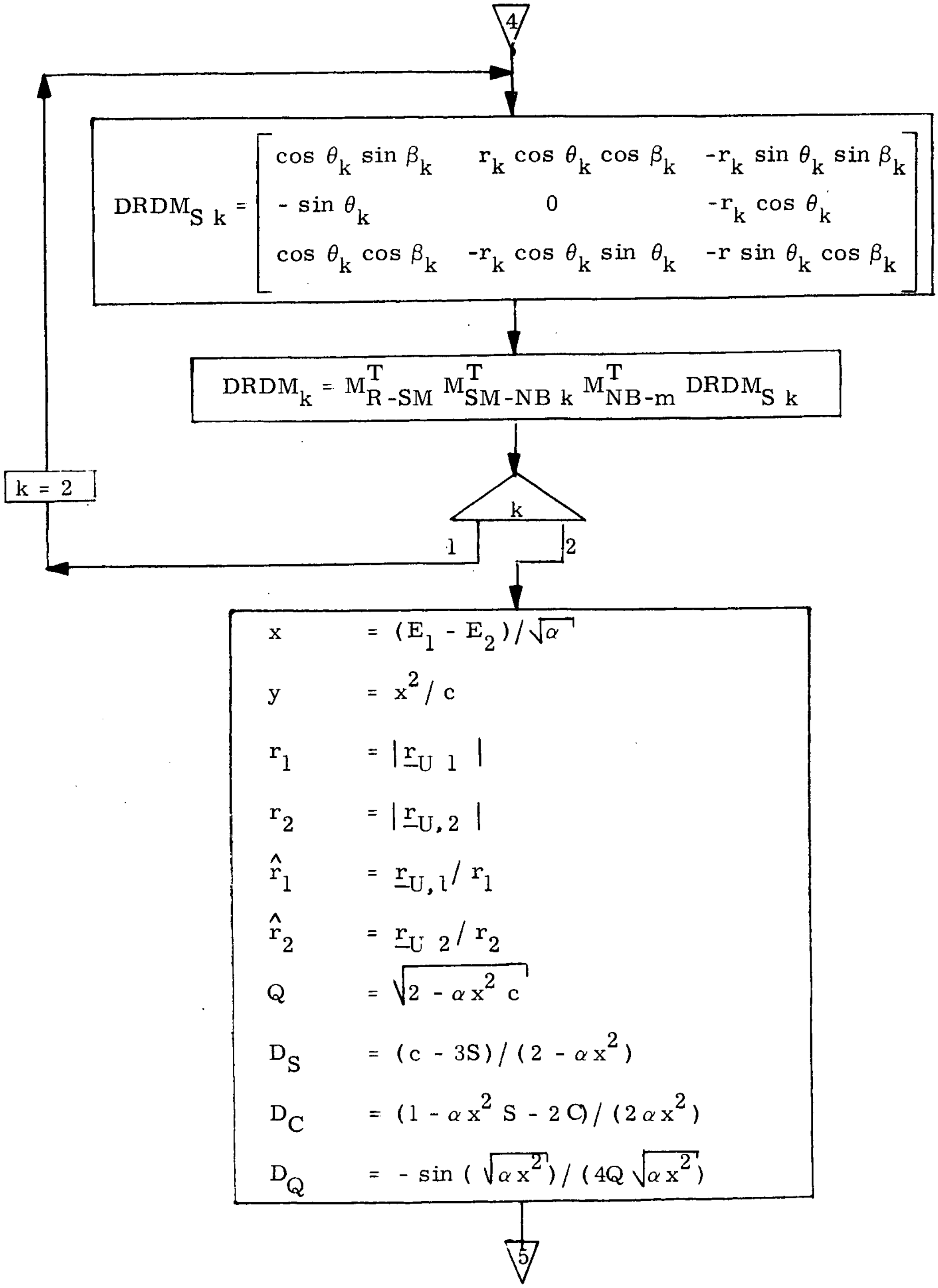

2
3
k
1
2
DT = $t_2 - t_1$
CALL LAMBERT ROUTINE (TBD)
$\underline{r}_{U\,1}$, $\underline{r}_{U\,2}$, DT
Resume $\underline{v}_{U\,2}$, α, E_1, E_2, S, C
UPD > 1
N
Y
$\underline{x}_{P\,2} = \begin{bmatrix} \underline{r}_{U\,2} \\ \underline{v}_{U\,2} \end{bmatrix}$
$\underline{x}_{T\,2} = \begin{bmatrix} \underline{r}_{U\,2} \\ \underline{v}_{U\,2} \end{bmatrix}$
k = 1
4

Figure 4c DETAILED FLOW DIAGRAM,AUTOMATIC INITIALIZATION ROUTINE



$$DRDM_{S\ k} = \begin{bmatrix} \cos\theta_k \sin\beta_k & r_k \cos\theta_k \cos\beta_k & -r_k \sin\theta_k \sin\beta_k \\ -\sin\theta_k & 0 & -r_k \cos\theta_k \\ \cos\theta_k \cos\beta_k & -r_k \cos\theta_k \sin\theta_k & -r \sin\theta_k \cos\beta_k \end{bmatrix}$$

$$DRDM_k = M^T_{R\text{-}SM}\, M^T_{SM\text{-}NB\ k}\, M^T_{NB\text{-}m}\, DRDM_{S\ k}$$

$$x = (E_1 - E_2)/\sqrt{\alpha}$$

$$y = x^2/c$$

$$r_1 = |\underline{r}_{U\ 1}|$$

$$r_2 = |\underline{r}_{U,2}|$$

$$\hat{r}_1 = \underline{r}_{U,1}/r_1$$

$$\hat{r}_2 = \underline{r}_{U\ 2}/r_2$$

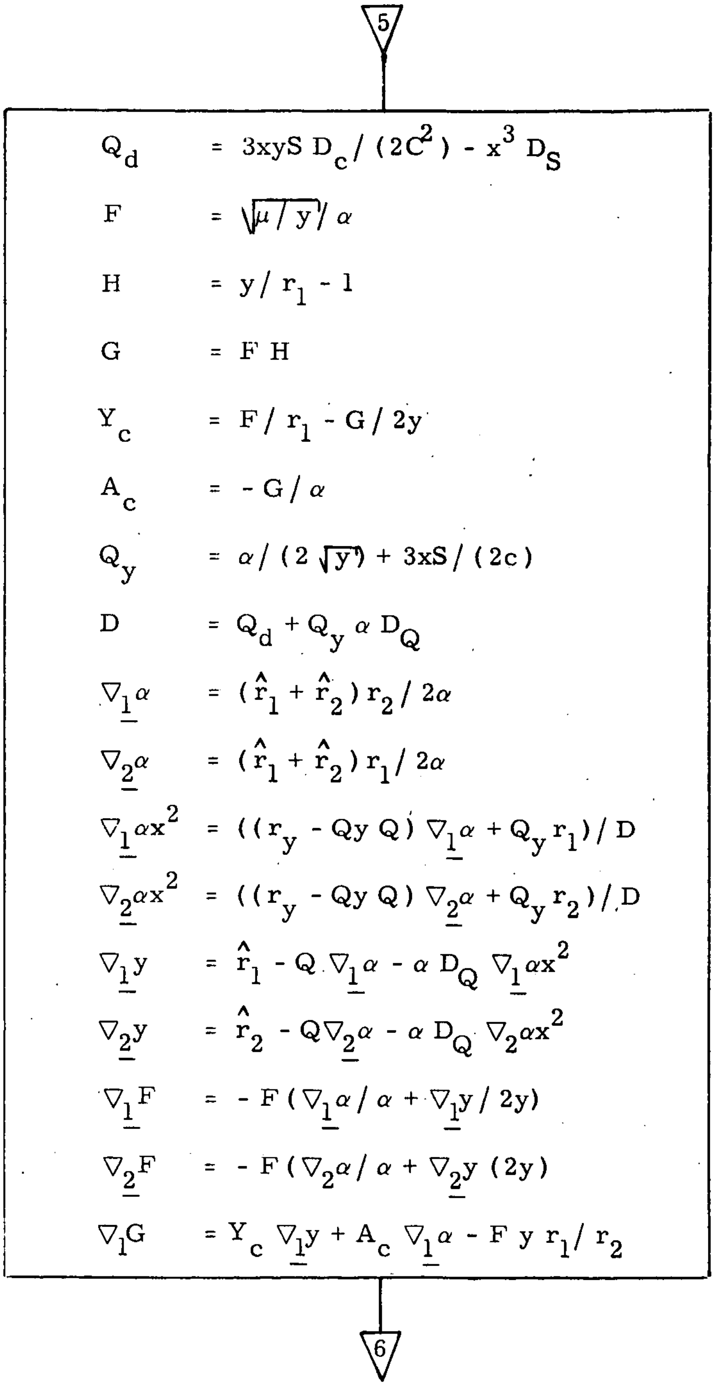
$$Q = \sqrt{2 - \alpha x^2 c}$$

$$D_S = (c - 3S)/(2 - \alpha x^2)$$

$$D_C = (1 - \alpha x^2 S - 2C)/(2\alpha x^2)$$

$$D_Q = -\sin(\sqrt{\alpha x^2})/(4Q\sqrt{\alpha x^2})$$

Figure 4d DETAILED FLOW DIAGRAM, AUTOMATIC INITIALIZATION ROUTINE



$$Q_d = 3xyS\, D_c / (2C^2) - x^3\, D_S$$

$$F = \sqrt{\mu / y} / \alpha$$

$$H = y / r_1 - 1$$

$$G = F\, H$$

$$Y_c = F / r_1 - G / 2y$$

$$A_c = - G / \alpha$$

$$Q_y = \alpha / (2 \sqrt{y}) + 3xS / (2c)$$

$$D = Q_d + Q_y\, \alpha\, D_Q$$

$$\nabla_{\underline{1}}\alpha = (\hat{r}_1 + \hat{r}_2)\, r_2 / 2\alpha$$

$$\nabla_{\underline{2}}\alpha = (\hat{r}_1 + \hat{r}_2)\, r_1 / 2\alpha$$

$$\nabla_{\underline{1}}\alpha x^2 = ((r_y - Qy\, Q)\, \nabla_{\underline{1}}\alpha + Q_y\, r_1) / D$$

$$\nabla_{\underline{2}}\alpha x^2 = ((r_y - Qy\, Q)\, \nabla_{\underline{2}}\alpha + Q_y\, r_2) / D$$

$$\nabla_{\underline{1}}y = \hat{r}_1 - Q\, \nabla_{\underline{1}}\alpha - \alpha\, D_Q\, \nabla_{\underline{1}}\alpha x^2$$

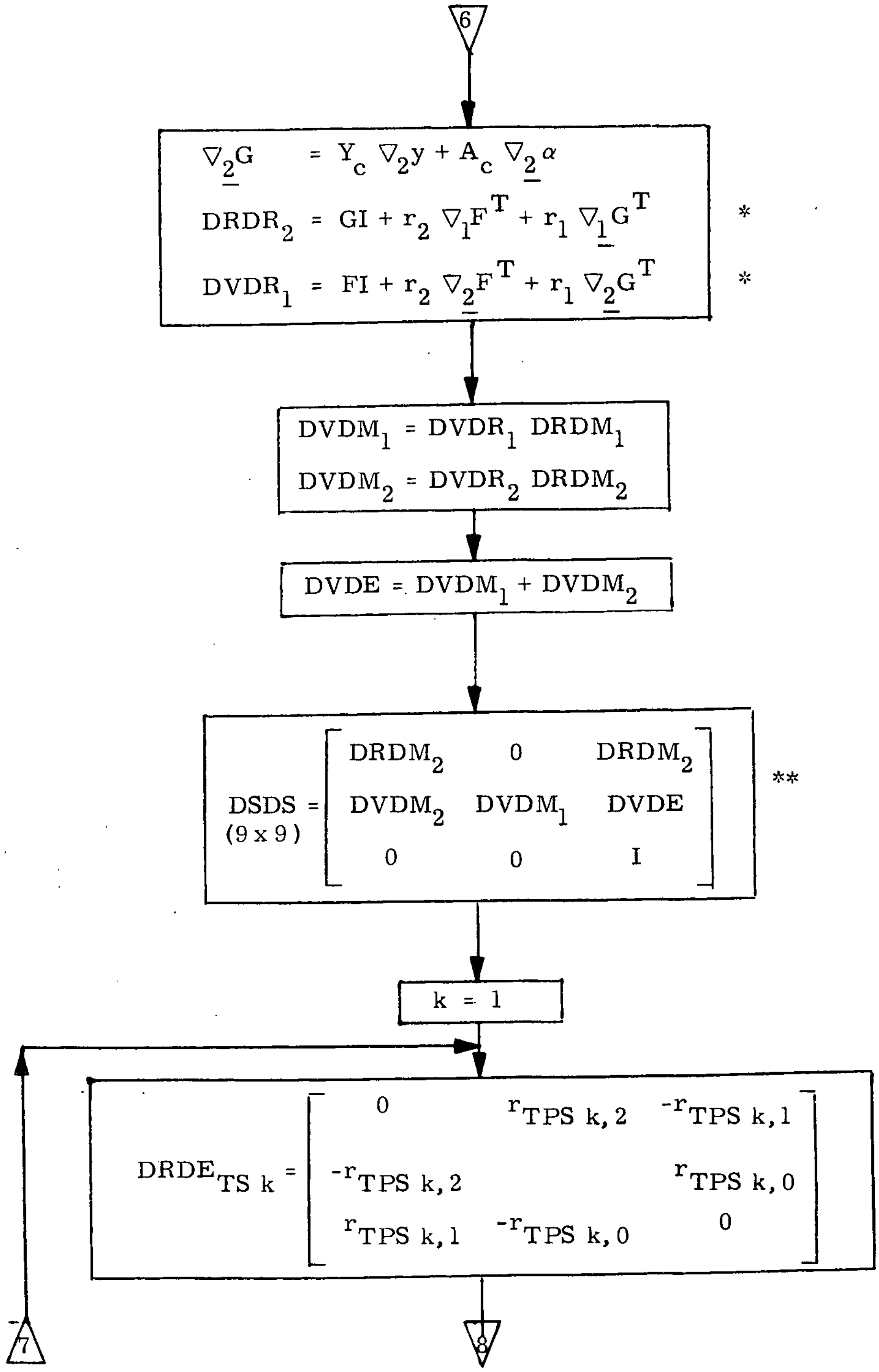
$$\nabla_{\underline{2}}y = \hat{r}_2 - Q \nabla_{\underline{2}}\alpha - \alpha\, D_Q\, \nabla_2 \alpha x^2$$

$$\nabla_{\underline{1}}F = - F(\nabla_{\underline{1}}\alpha / \alpha + \nabla_{\underline{1}}y / 2y)$$

$$\nabla_{\underline{2}}F = - F(\nabla_2\alpha / \alpha + \nabla_{\underline{2}}y\, (2y))$$

$$\nabla_1 G = Y_c\, \nabla_{\underline{1}}y + A_c\, \nabla_{\underline{1}}\alpha - F\, y\, r_1 / r_2$$

Figure 4e DETAILED FLOW DIAGRAM, AUTOMATIC INITIALIZATION ROUTINE



$$\nabla_{\underline{2}} G = Y_c \nabla_2 y + A_c \nabla_{\underline{2}} \alpha$$

$$DRDR_2 = GI + r_2 \nabla_1 F^T + r_1 \nabla_{\underline{1}} G^T \quad *$$

$$DVDR_1 = FI + r_2 \nabla_{\underline{2}} F^T + r_1 \nabla_{\underline{2}} G^T \quad *$$

$$DVDM_1 = DVDR_1 \; DRDM_1$$

$$DVDM_2 = DVDR_2 \; DRDM_2$$

$$DVDE = DVDM_1 + DVDM_2$$

$$\underset{(9 \times 9)}{DSDS} = \begin{bmatrix} DRDM_2 & 0 & DRDM_2 \\ DVDM_2 & DVDM_1 & DVDE \\ 0 & 0 & I \end{bmatrix} \quad **$$

$$k = 1$$

$$DRDE_{TS\ k} = \begin{bmatrix} 0 & r_{TPS\ k,2} & -r_{TPS\ k,1} \\ -r_{TPS\ k,2} & & r_{TPS\ k,0} \\ r_{TPS\ k,1} & -r_{TPS\ k,0} & 0 \end{bmatrix}$$

*I is the 3D identity matrix

**0 is the 3D null matrix

Figure 4f DETAILED FLOW DIAGRAM, AUTOMATIC INITIALIZATION ROUTINE

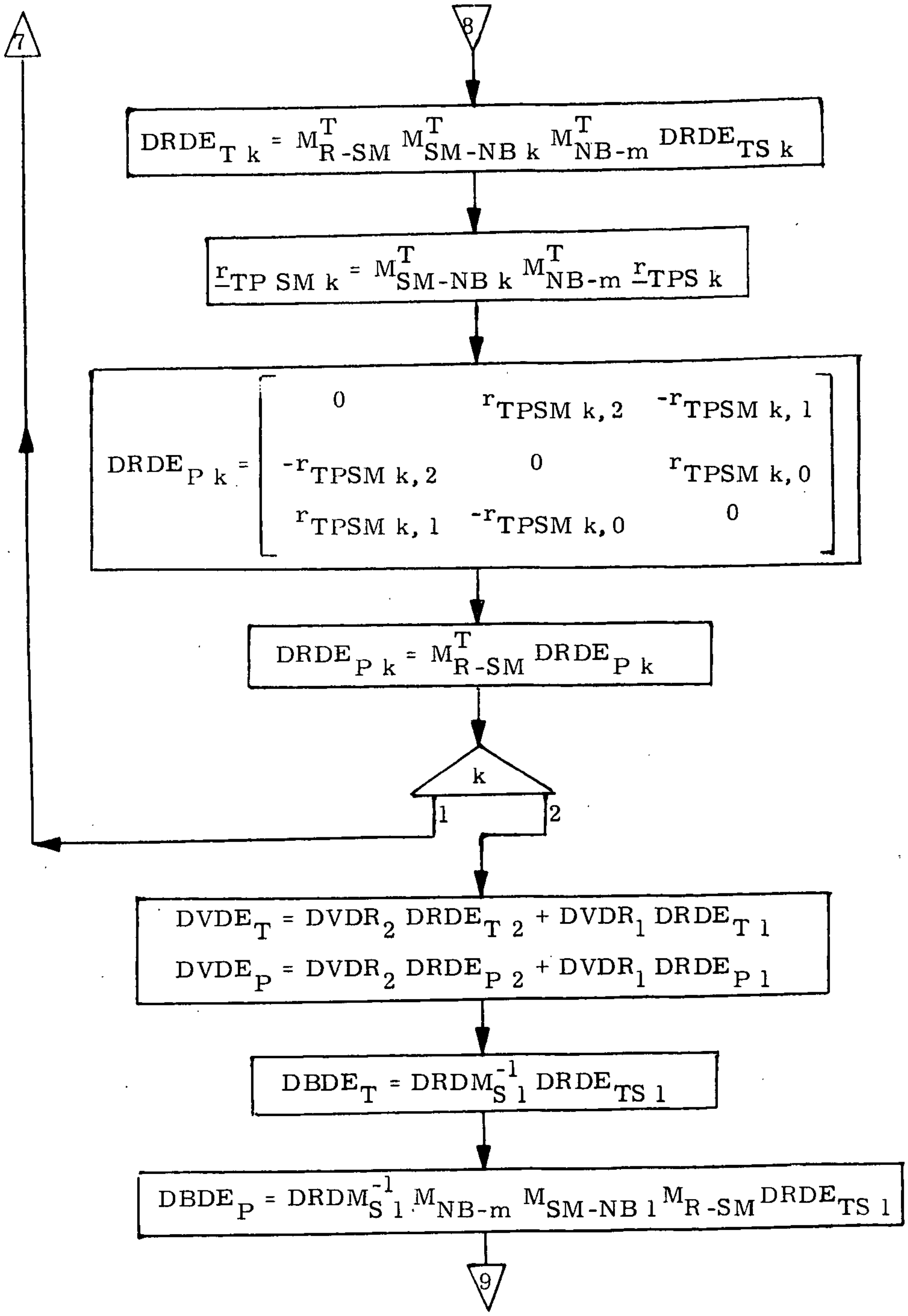

8
7
$DRDE_{T\,k} = M^T_{R\text{-}SM}\ M^T_{SM\text{-}NB\,k}\ M^T_{NB\text{-}m}\ DRDE_{TS\,k}$
$\underline{r}_{TP\,SM\,k} = M^T_{SM\text{-}NB\,k}\ M^T_{NB\text{-}m}\ \underline{r}_{TPS\,k}$
$DRDE_{P\,k} = \begin{bmatrix} 0 & r_{TPSM\,k,2} & -r_{TPSM\,k,1} \\ -r_{TPSM\,k,2} & 0 & r_{TPSM\,k,0} \\ r_{TPSM\,k,1} & -r_{TPSM\,k,0} & 0 \end{bmatrix}$
$DRDE_{P\,k} = M^T_{R\text{-}SM}\ DRDE_{P\,k}$
k
1
2
$DVDE_T = DVDR_2\ DRDE_{T\,2} + DVDR_1\ DRDE_{T\,1}$
$DVDE_P = DVDR_2\ DRDE_{P\,2} + DVDR_1\ DRDE_{P\,1}$
$DBDE_T = DRDM^{-1}_{S\,1}\ DRDE_{TS\,1}$
$DBDE_P = DRDM^{-1}_{S\,1}\ M_{NB\text{-}m}\ M_{SM\text{-}NB\,1}\ M_{R\text{-}SM} DRDE_{TS\,1}$
9

Figure 4g DETAILED FLOW DIAGRAM,AUTOMATIC INITIALIZATION ROUTINE

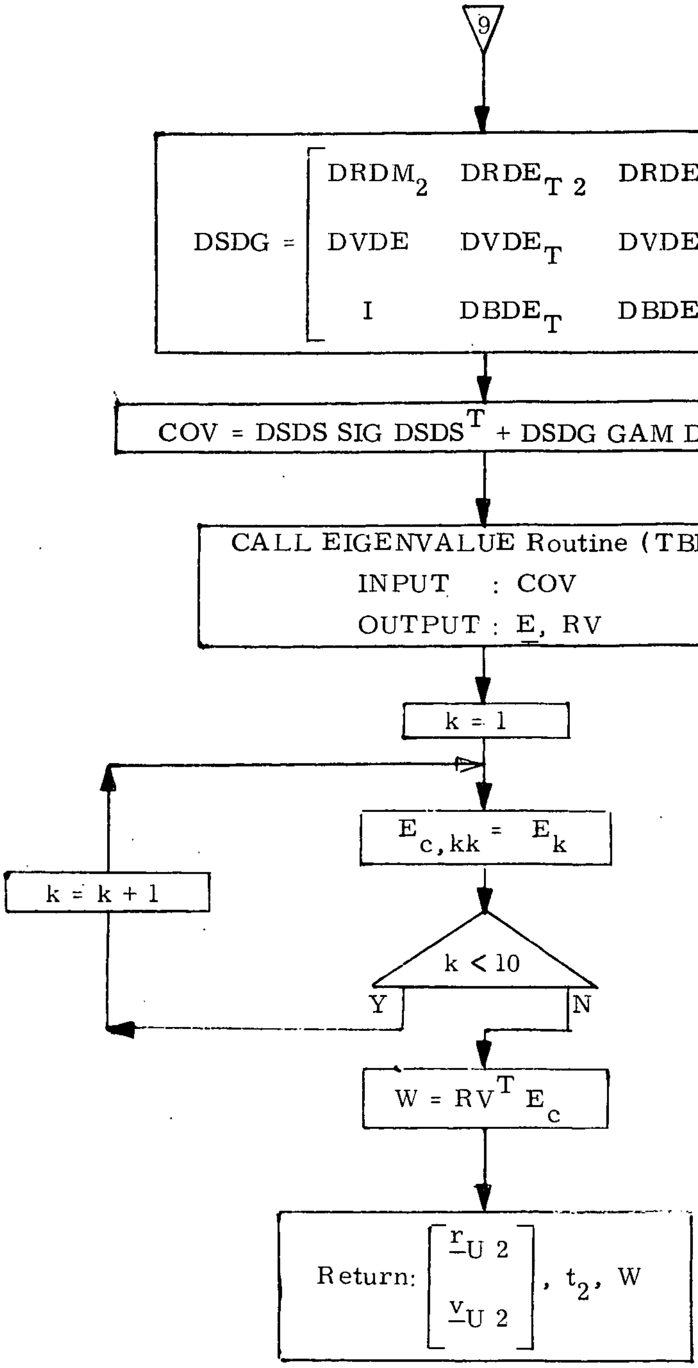

9
DSDG = $\begin{bmatrix} DRDM_2 & DRDE_{T\,2} & DRDE_{P\,2} \\ DVDE & DVDE_T & DVDE_P \\ I & DBDE_T & DBDE_P \end{bmatrix}$
COV = DSDS SIG DSDS^T + DSDG GAM DS DG^T
CALL EIGENVALUE Routine (TBD)
INPUT : COV
OUTPUT : $\underline{E}$, RV
k = 1
$E_{c,kk}$ = E_k
k = k + 1
k < 10
Y
N
W = RV^T E_c
Return: $\begin{bmatrix} \underline{r}_{U\,2} \\ \underline{v}_{U\,2} \end{bmatrix}$, t_2, W

Figure 4h DETAILED FLOW DIAGRAM, AUTOMATIC INITIALIZATION ROUTINE

4. SUPPLEMENTARY INFORMATION

The equations presented in this report are the results to date of studies performed under a G & C shuttle task to develop G & N equations for automatic rendezvous. Two fundamental approaches were taken in these studies: (1) automate proven Apollo rendezvous navigation equations; (2) develop optimum rendezvous navigation equations By presenting the equations in the general form shown, they are made to reflect formulations developed using both approaches (1) and (2). Analyses performed to evaluate the filter equations are reported in the references.

To complete the automation of the Apollo filter, an automatic mark reject routine remains to be formulated.

References

1. Muller, E.S., Kachmar, P.M., The Apollo Rendezvous Navigation Filter-Theory, Description and Performance, Vol. 1 of 2, Draper Lab Report R-649, June 1970, MIT.

2. Muller, E. and Kachmar, P., STS Progress Report - Rendezvous Studies, Memo 23A STS #11-70, May 1970, MIT.

3. Phillips, R., Lambert Determination of the Relative State for Rendezvous, Memo 23A STS #6-70, 28 April 1970, MIT.

4. Tempelman, W., Non-Cooperative Rendezvous Trajectories, 23A STS Memo #4-70, 30 April 1970, MIT.

5. Sears, N.E., et. al., STS Avionic Specification Presentation, C-3397, 12 November 1969, MIT.

6. Users Guide to Minkey Rendezvous, E-2448, 17 July 1970, MIT.

9.8.3 Rendezvous Guidance (same as 9.7.3)

9.9 STATION KEEPING MISSION PHASE

Station keeping begins with the targeting for braking as the Shuttle approaches the target vehicle sometime after TPI. This phase includes braking targeting, braking, positioning for station keeping, automatic station keeping, repositioning to station keep at a different position relative to the target vehicle and/or in preparation for docking. Automatic station keeping here means the preservation of a precise relative position with the target vehicle with no requirement for manual commands. Automatic station keeping may occur before docking, after docking, and on missions in which docking does not occur. This phase ends when the docking maneuver begins, or when the shuttle is separated from the target vehicle with no intention of preserving a precise relative position with it.

The software functions required in this mission phase are the following:

1. Estimate relative state of target vehicle based on external measurements.

2. Estimate absolute states of both shuttle and target vehicle.

3. Compute (target) the braking ΔV(s) required, their direction, and the time(s) of ignition.

4. Execute braking maneuver by commanding engine(s) on, providing attitude commands during braking, and commanding engine(s) off.

5. Powered flight navigation.

6. Automatically preserve a relative position and attitude with the target vehicle by periodic RCS engine on/off commands with a minimum-fuel technique. Spatial and angular requirements and allowable variations during automatic station keeping are TBD.

7. Provide RCS engine commands to achieve commanded attitude during ΔV maneuvers and during coast periods (digital autopilot).

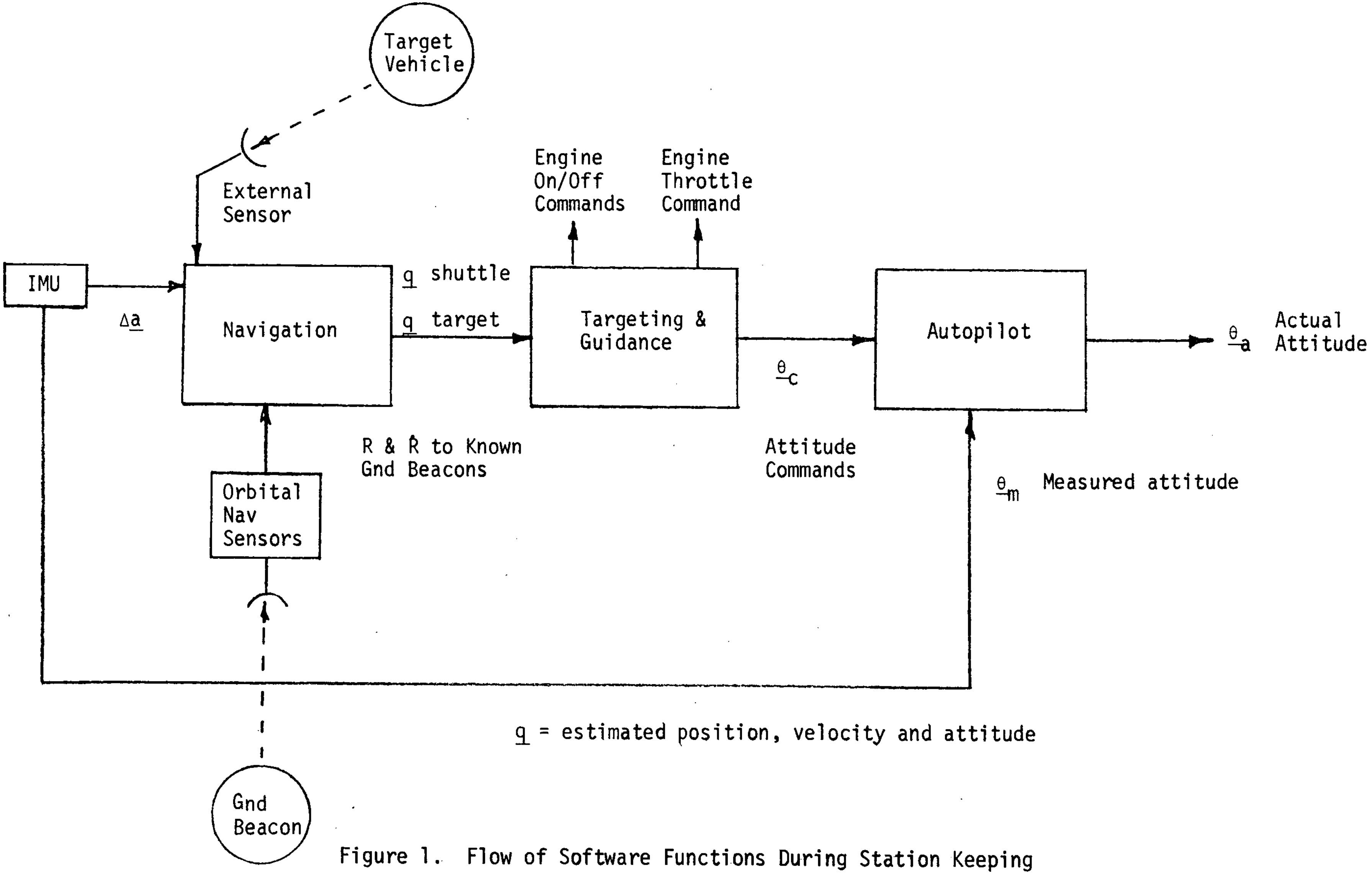
Repositioning for docking maneuver initiation, for a separation maneuver, or for station keeping at a different relative position is assumed to be a manual function and therefore no software for performing these maneuvers automatically is required.

A flow of software functions during station keeping appears in Figure 1. Some functions overlap with other mission phases and only those equations not provided in earlier sections are discussed here.

9.9.1 Relative State Estimation (TBD)

9.9.2 Station Keeping Guidance (TBD)

9.9.3 Station Keeping Attitude Control (TBD)



$\underline{q}$ = estimated position, velocity and attitude

Figure 1. Flow of Software Functions During Station Keeping

9.10 DOCKING AND UNDOCKING

The two distinct events are described as one phase since the events are essentially reversals of one another. The distinction between the docking event and terminal rendezvous is the point at which the maneuver defined by the docking constraints on such variables as range, range rate, attitude, and attitude rate is initiated.

The mode of docking is still open; that is, it has not been determined whether the docking will be performed manually or automatically, with a manual backup capability. The GN&C software functions to be performed during this phase are based on an automatic docking with manual backup. The docking SW functions are:

a) Specific force integration updates of relative states during translational burns. This function will maintain the relative state between the orbiter and its co-orbiting target during orbiter burns.

b) Maintain attitude-hold about a desired orientation.

c) Compute and command steered-attitude RCS ΔV maneuvers for docking.

d) Make high-frequency steering estimates between guidance samples for docking.

e) Provide three-axis translation control.

The SW functions for undocking are:

a) Configure all GN&C systems for the next mission phase.

b) Schedule undocking.

c) Compute and command ΔV translations.

d) Provide capability to advance inertial state vector from an initial state to a final state.

e) Provide for specific force integration updates of relative state during burns associated with undocking.

f) Compute and command attitude-hold RCS ΔV maneuvers.

Figure 1 displays a function flow diagram of the docking GN&C software functions.

Presently, no specific sensors for automatic docking have been baselined. However, control laws and a navigation routine have been approved by the GN&C Software Equation Formulation and Implementation Panel. These equation formulations are described in the following references:

a) E. T. Kubiak, "Automatic Docking Control Law," MSC EG2-3-71, date 5 January 1971.

b) E. P. Blanchard, G. M. Levine, "Docking and Undocking Navigation," MIT No. 2-71, dated January 1971.

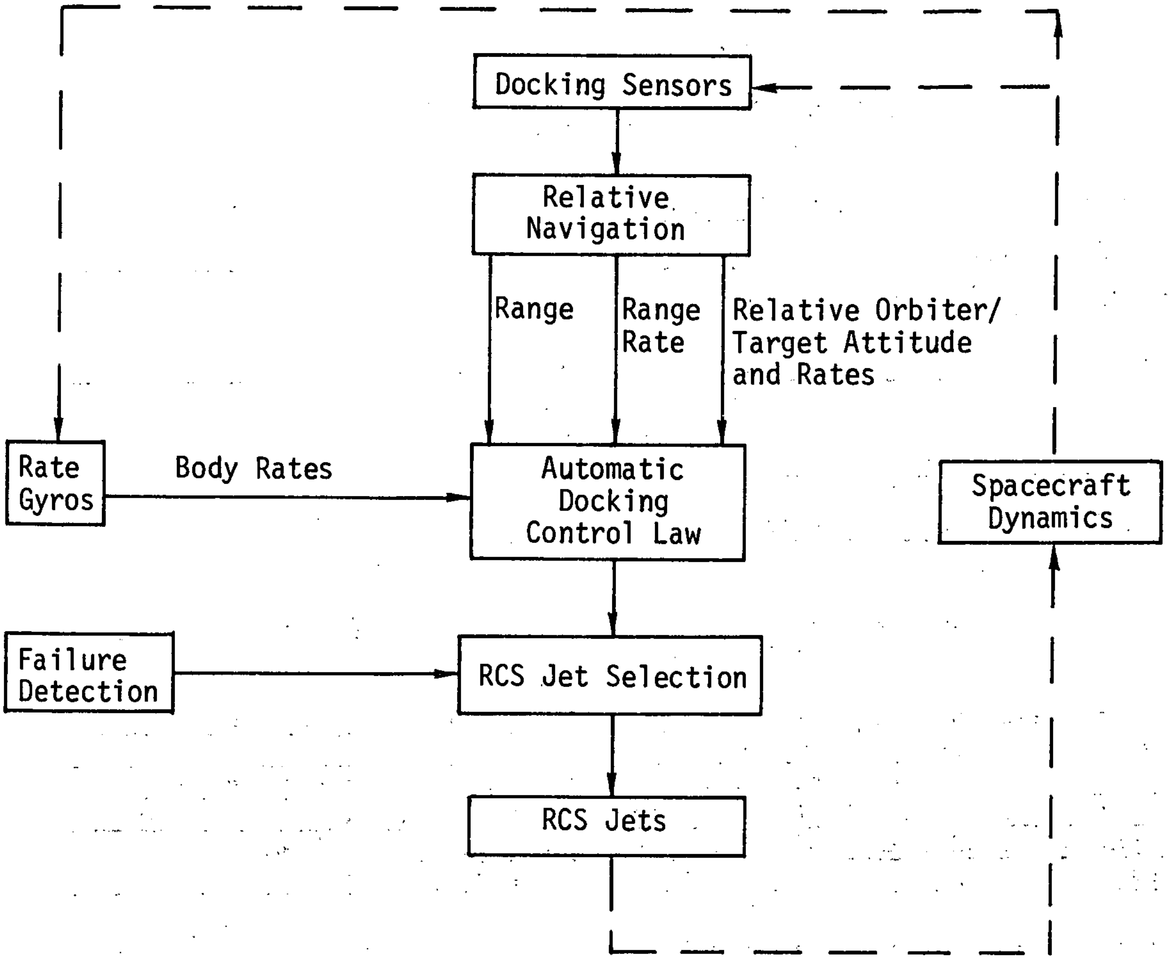


Figure 1

Overall Functional Flow Diagram for Docking and Undocking

SPACE SHUTTLE

GN&C SOFTWARE EQUATION SUBMITTAL

Software Equation Section Docking and Undocking Submittal No. 5

Function Relative Navigation

Module No. ON3 Function No. -2, -5, -8 (MSC 03690)

Submitted By: E. P. Blanchard, G. M. Levine Co. MIT
(Name)

Date: January 1971

NASA Contact: W. H. Peters Organization EG2
(Name)

Approved by Panel III K. J. Cox K.J. Cox Date 3/10/71
(Chairman)

Summary Description: The objective of the Docking and Undocking Navigation Program is to use the data from the docking sensor to determine the relative position and attitude of the target vehicle with respect to the shuttle. These quantities and their rates are computed periodically and used in the generation of guidance commands during both the docking and undocking procedure.

Shuttle Configuration: (Vehicle, Aero Data, Sensor, Et Cetera) Assumes a docking sensor which measures the azimuth and elevation angles to each of four sources located on the target vehicle.

Comments:

(Design Status) The algorithm for source identification is TBD.

(Verification Status) Open-loop testing has been performed simulating the sensor-target geometry and the sensor.

Panel Comments: The equations are baselined subject to the qualification that they are based on a sensor configuration which has not been baselined. Also, the range and range rate computations must be coordinated with those in the Automatic Docking Control Law.

9.10.1 Docking and Undocking Navigation

1. INTRODUCTION

The objective of the Docking and Undocking Navigation Program is to use the data from the docking sensor to determine the relative position and attitude of the target vehicle with respect to the shuttle. These quantities and their rates are computed periodically and used in the generation of guidance commands during both the docking and undocking procedure.

The docking sensor measures the azimuth and elevation angles to each of four sources located on the target vehicle. The configuration of these four sources is designed to permit recognition of one source by its angular position relative to the other sources under all allowable rotations of the shuttle with respect to the target vehicle within certain restricted operating limits. As long as the operating-limit restrictions are satisfied, it is not necessary for the sensor to identify individually the sources; i.e., the sensor portion of the system does not have to associate a particular source with each set of azimuth and elevation angles, that process can be accomplished computationally. Furthermore, in this case, the data from only three of the four sources are required to obtain a complete relative position and attitude solution. The velocity and attitude rates are determined by numerically differencing two position and attitude solutions.

On the other hand, if the operating-limit restrictions are violated, then the equations have multiple solutions, and all four sets of data must be used to resolve the ambiguities.

An additional reason for the presence of four sources is to provide an option for selecting the best combination of three sources; i.e., at close range to permit selection of sources which fall within the sensor field of view, and at long range at provide a combination of three sources which yield a more accurate solution.

NOMENCLATURE

A	Intermediate matrix
a_i	Azimuth angle to source i
B	Intermediate matrix
C	Cos 40^o
e_i	Elevation angle to source i
f	Rate indicator
FLAG	Flag used in iteration
$FLAG_m$	Flag used in rate calculation
I	Negative radicand indicator
$\underline{i}_{XS}$ $\underline{i}_{YS}$ $\underline{i}_{ZS}$	Unit vectors along shuttle coordinate axes
$\underline{i}_{XT}$ $\underline{i}_{YT}$ $\underline{i}_{ZT}$	Unit vectors along target vehicle coordinate axes
K	0.4 or 2.5 depending on selected source set
k	Index used in rate calculations
M	Transformation matrix
m	Index used in rate calculations
m_{ij}	Element of M
n	Index used in rate calculations
p	Source set indicator

9.10.1 Docking and Undocking Navigation (con't)

$\underline{r}$	Relative position vector between docking hatches
$\dot{\underline{r}}$	Rate of change of $\underline{r}$
$\underline{r}_i$	Vector from sensor to source i
r_i	Magnitude of $\underline{r}_i$
$\underline{r}_{ij}$	Vector from source i to source j
r_{ij}	Magnitude of $\underline{r}_{ij}$
r_M	Maximum value of s_1
r_{new}	Iteration interval end points
r_{old}	
S	Sin 40°
s_1	Trial value of r_1
$\underline{y}$	Vector from sensor to shuttle docking hatch
$\underline{z}$	Vector from source 1 to target vehicle docking hatch
$\underline{\gamma}$	$\gamma_1, \gamma_2, \gamma_3$
$\dot{\underline{\gamma}}$	Rate of change of $\underline{\gamma}$
γ_1	Rotation angles
γ_2	
γ_3	
Δr	$s_1 - r_1$
Δr_{old}	Previous value of Δr
Δt	Navigation cycle time
ϵ	Error tolerance
θ_{ij}	Angle between lines-of-sight to sources i and j
ρ	Scaling factor
Subscript S	Shuttle coordinates
Subscript T	Target vehicle coordinates

2. SOURCE CONFIGURATION AND OPERATING LIMITS

In this section, the configuration of the four sources on the target vehicle is described, and the operating limits under which a unique relative position and attitude solution can be obtained is discussed.

Refering to Fig. 1, define a coordinate system fixed in the target vehicle with origin at source 1; X axis parallel to the docking axis; and $\underline{i}_{XT}$, $\underline{i}_{YT}$, and $\underline{i}_{ZT}$ unit vectors along the three axes. Let $\underline{r}_{ij}$ be the vector from source i to source j. Then the locations of sources 2, 3, and 4 are defined by

$$\underline{r}_{12T} = 0.4\rho\begin{pmatrix} \cos 40^{\circ} \\ -\sin 40^{\circ} \\ 0 \end{pmatrix}$$

$$\underline{r}_{13T} = \rho\begin{pmatrix} \cos 40^{\circ} \\ 0 \\ \sin 40^{\circ} \end{pmatrix}$$

$$\underline{r}_{14T} = 2.5\,\rho\begin{pmatrix} \cos 40^{\circ} \\ -\sin 40^{\circ} \\ 0 \end{pmatrix}$$

where the subscript T denotes target vehicle coordinates and ρ is a scaling factor.

In order to discuss the restricted operating limits, define a coordinate system centered at the docking sensor in the shuttle with unit vectors $\underline{i}_{XS}$, $\underline{i}_{YS}$, and $\underline{i}_{ZS}$ along its axes. Again, let the X axis be parallel to the shuttle docking axis. Let γ_1, γ_2, and γ_3 be the three rotation angles which make the shuttle coordinate system parallel to the target vehicle system (the condition required for docking); i.e., a rotation of the shuttle system about the X axis through an angle γ_1, then a rotation about the resulting Y axis through an angle γ_2, and finally a rotation about the resulting Z axis through an angle γ_3 make the two systems parallel.

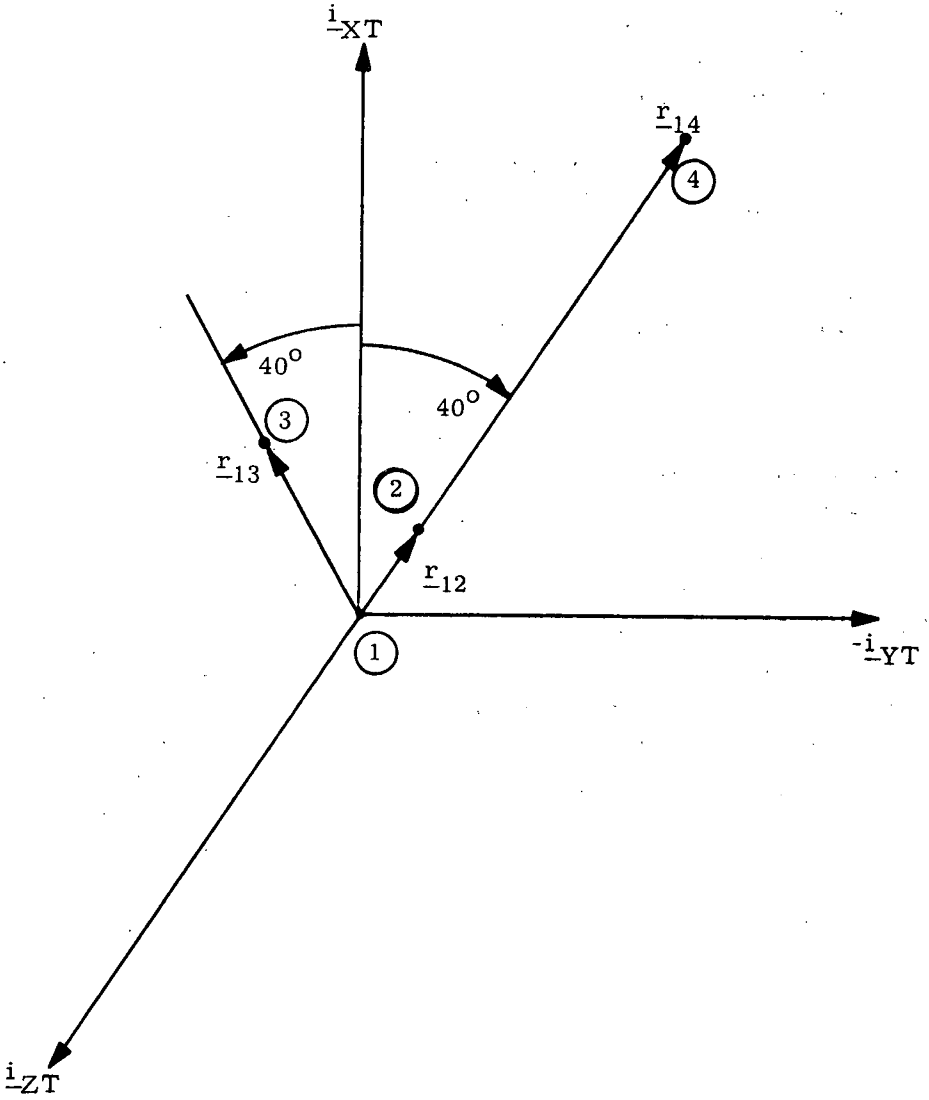


Figure 1 TARGET VEHICLE COORDINATE SYSTEM AND SOURCE CONFIGURATION

If the X (docking) axis of the shuttle is kept within 30° of the target vehicle docking axis, then an identification of the four sources can be made. Figure 2 illustrates the appearance of the sources for various relative vehicle attitudes for the case of zero γ_1. The center illustration shows the appearance of the sources when the two vehicles are properly aligned for docking; the other eight illustrations show the appearance at various points on the surface of the 30° cone defining the operation region.

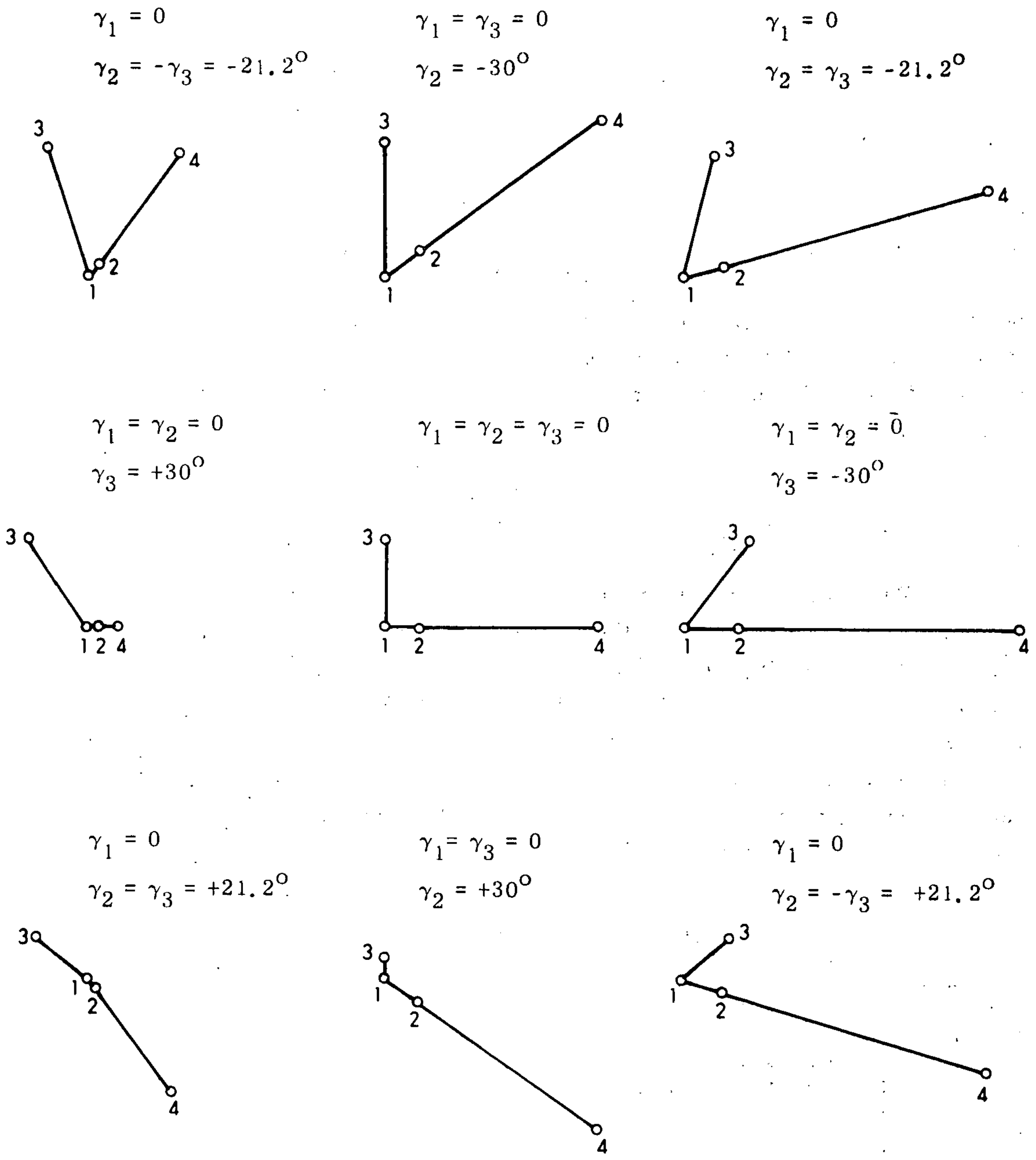
For all relative vehicle orientations within the operating region, the following two facts hold:

1) Sources 1, 2, and 4 lie on a straight line.

2) The observed distance between sources 1 and 2 always has the same ratio with respect to the observed distance between sources 1 and 4.

These two facts permit identification of the four sets of paired azimuth and elevation angles with the four sources.

The source configuration has also been selected to assure that for all relative vehicle orientations within the 30° operating region the distances from the sensor to the sources will satisfy the relationship $r_1 < r_2 < r_3 < r_4$. This relationship provides the resolution of the multiple solutions which would otherwise exist in the navigation equations.



$\gamma_1 = 0$
$\gamma_2 = -\gamma_3 = -21.2^o$

$\gamma_1 = \gamma_3 = 0$
$\gamma_2 = -30^o$

$\gamma_1 = 0$
$\gamma_2 = \gamma_3 = -21.2^o$

$\gamma_1 = \gamma_2 = 0$
$\gamma_3 = +30^o$

$\gamma_1 = \gamma_2 = \gamma_3 = 0$

$\gamma_1 = \gamma_2 = \bar{0}$
$\gamma_3 = -30^o$

$\gamma_1 = 0$
$\gamma_2 = \gamma_3 = +21.2^o$

$\gamma_1 = \gamma_3 = 0$
$\gamma_2 = +30^o$

$\gamma_1 = 0$
$\gamma_2 = -\gamma_3 = +21.2^o$

Figure 2 APPEARANCE OF SOURCES VS RELATIVE VEHICLE ATTITUDE

3. FUNCTIONAL FLOW DIAGRAM

The sequencing of functions performed by the Docking and Undocking Navigation Program is described in this section and illustrated by the functional flow diagram in Fig. 3.

The program is called periodically by the Docking and Undocking Guidance Program. The first function performed is to identify the four sources from the two facts discussed in Section 2. Next, the appropriate sources are selected and the unique relative position and attitude solution is determined. Included in this solution is the relative position of the two docking hatches. The final step is to compute velocity and angle rates by differencing two solutions for position and angle.

4. PROGRAM INPUT-OUTPUT

The required inputs to the program are the four sets of azimuth and elevation angles of the four sources relative to the docking sensor; and two indicators, the first of which indicates which of the two combinations of three sources (1, 2, and 3) or (1, 2, and 4) have been selected, and the second is used in the rate calculations. The outputs of the program are solutions for the relative position of the two docking hatches, the rotation angles between the two vehicles, and the rates of change of these quantities.

Input Parameters

$$\left.\begin{array}{l}(a_1, e_1)\\(a_2, e_2)\\(a_3, e_3)\\(a_4, e_4)\end{array}\right\}\ \text{Four sets of paired azimuth and elevation angles but not identified with any of the four sources}$$

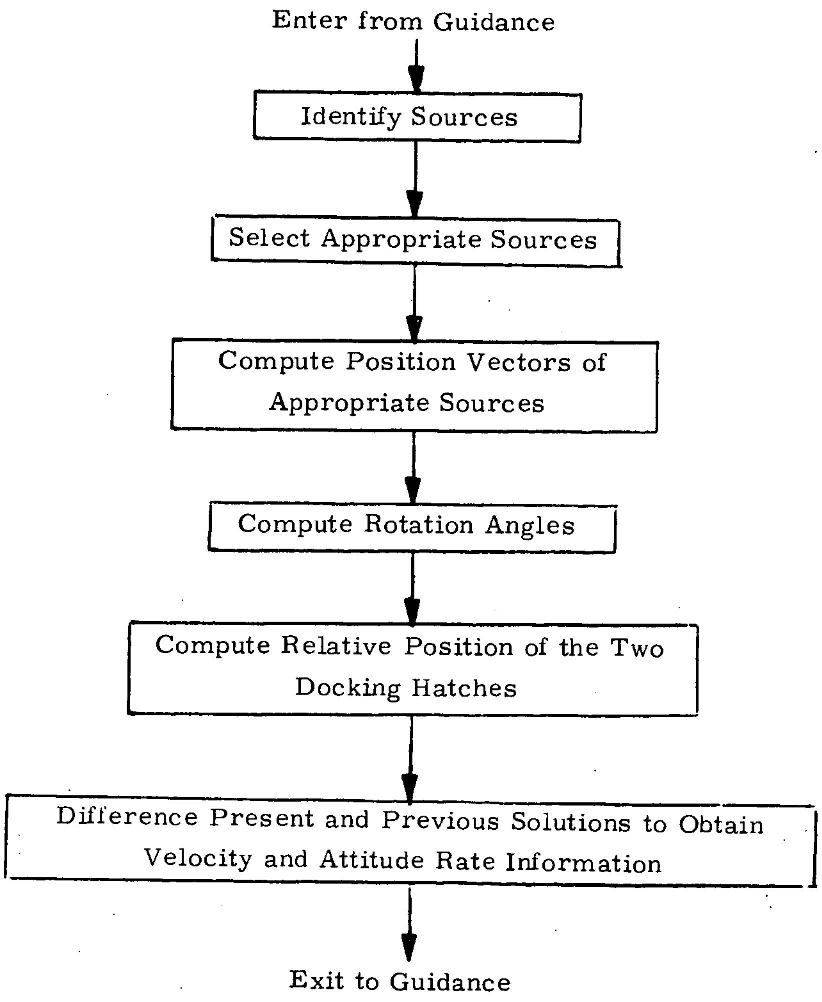

Enter from Guidance
Identify Sources
Select Appropriate Sources
Compute Position Vectors of Appropriate Sources
Compute Rotation Angles
Compute Relative Position of the Two Docking Hatches
Difference Present and Previous Solutions to Obtain Velocity and Attitude Rate Information
Exit to Guidance

Figure 3 FUNCTIONAL FLOW DIAGRAM

C.9.

The elevation angle is the angle between the line-of-sight and the XY plane of the shuttle coordinate system. The azimuth angle is the angle between the X axis of the shuttle coordinate system and the projection of the line-of-sight on the XY plane. See Figure 4.

p Source set indicator $= \begin{cases} 2 \text{ if selected source set is } (1,2,3) \\ 4 \text{ if selected source set is } (1,3,4) \end{cases}$

f Rate indicator = Number of cycles separating differenced solutions in rate calculations.

Output Parameters

$\underline{r}_S$ Position vector of target vehicle docking hatch relative to shuttle docking hatch in shuttle coordinates

$\dot{\underline{r}}_S$ Rate of change of $\underline{r}_S$

$\underline{\gamma} = (\gamma_1, \gamma_2, \gamma_3) =$ Rotation angles

$\dot{\underline{\gamma}} = (\dot{\gamma}_1, \dot{\gamma}_2, \dot{\gamma}_3) =$ Rates of change of rotation angles

5. DESCRIPTION OF EQUATIONS

The computational sequence during the Docking and Undocking Navigation Program and the related equations are described in this section. These equations are recomputed every guidance cycle.

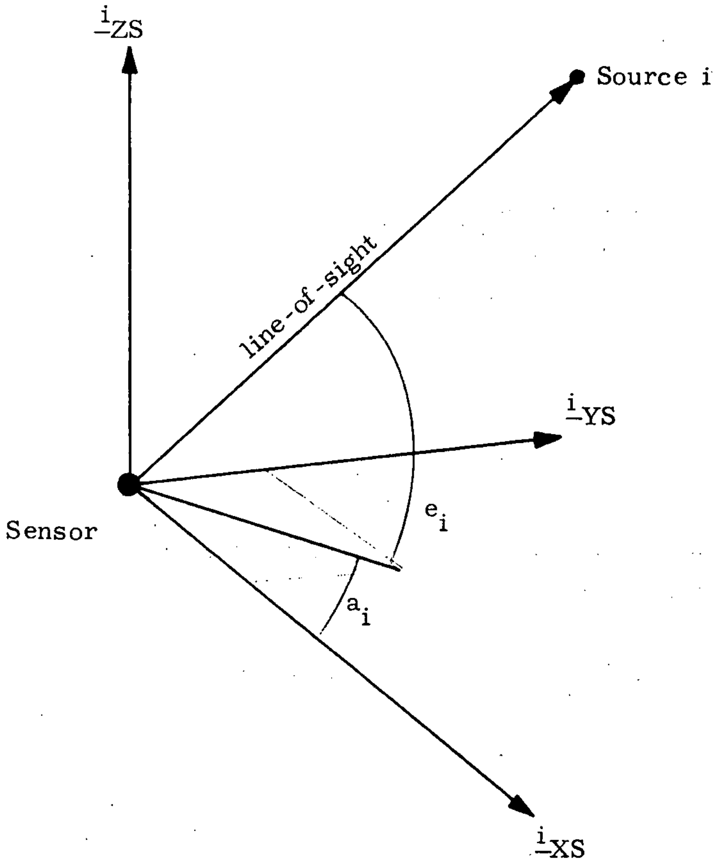


Figure 4 DEFINITION OF AZIMUTH AND ELEVATION ANGLES

5.1 Source Identification (TBD)

The first step in the program is to associate each of the four sets of azimuth and elevation angles with a particular source. The procedure for performing the association is based on the two facts discussed in Section 2; i.e.,

1) Sources 1, 2, and 4 lie on a straight line.

2) The observed distance between sources 1 and 2 has the same ratio with respect to the observed distance between sources 1 and 4.

The algorithm used is TBD.

5.2 Angles Between Lines-of-Sight

The cosines of the three angles between the lines-of-sight from the sensor to the sources in the selected set (based on indicator p) are computed from

$$\cos \theta_{ij} = \cos e_i \cos e_j \cos (a_i - a_j) + \sin e_i \sin e_j$$

for

$$ij = 13,\ 1p,\ \text{and } 3p$$

5.3 Distances to Sources

Let $\underline{r}_1$, $\underline{r}_2$, $\underline{r}_3$, and $\underline{r}_4$ be the vectors from the sensor to the four sources. The magnitudes of the three vectors associated with the selected sources satisfy

$$r_3 = r_1 \cos \theta_{13} + \sqrt{r_{13}^2 - (1 - \cos^2 \theta_{13}) r_1^2}$$

$$r_2 = r_3 \cos \theta_{32} - \sqrt{r_{32}^2 - (1 - \cos^2 \theta_{32}) r_3^2}$$

$$r_1 = r_2 \cos \theta_{12} - \sqrt{r_{12}^2 - (1 - \cos^2 \theta_{12}) r_2^2}$$

or

$$r_3 = r_1 \cos \theta_{13} + \sqrt{r_{13}^2 - (1 - \cos^2 \theta_{13}) r_1^2}$$

$$r_4 = r_3 \cos \theta_{34} + \sqrt{r_{34}^2 - (1 - \cos^2 \theta_{34}) r_3^2}$$

$$r_1 = r_4 \cos \theta_{14} - \sqrt{r_{14}^2 - (1 - \cos^2 \theta_{14}) r_4^2}$$

These equations are solved by an iterative interval-halving process in which s_1, a trial value of r_1, is used as input to compute an output value of r_1 by means of

$$r_3 = r_1 \cos \theta_{13} + \sqrt{r_{13}^2 - (1 - \cos^2 \theta_{13}) s_1^2}$$

$$r_p = r_3 \cos \theta_{3p} \pm \sqrt{r_{3p}^2 - (1 - \cos^2 \theta_{3p}) r_3^2} \qquad (1)$$

$$r_1 = r_p \cos \theta_{1p} - \sqrt{r_{1p}^2 - (1 - \cos^2 \theta_{1p}) r_p^2}$$

where the upper and lower signs correspond, respectively, with p=4 and p=2. Agreement between s_1 and r_1 indicates a correct solution.

The iteration is initiated by computing the maximum possible value for r_1 based on the sensor measurements

$$r_M = \frac{r_{13}}{\sqrt{1 - \cos^2 \theta_{13}}}$$

Then, using r_M as the first value for s_1; values for r_3, r_p, and r_1 are computed from Eq. (1). During these calculations, it is possible for one of the radicands to be negative, in which case the selected value of s_1 is too large. If this occurs, the value of s_1 is halved, and the computations are repeated. The process continues until three real numbers are obtained for r_3, r_p, and r_1 as functions of s_1. (It should be noted that once a value of s_1 which produces a real solution has been determined, then all smaller values of s_1 will also yield a real solution.)

The difference between the input and output values of r_1 is computed from

$$\Delta r = s_1 - r_1 \tag{2}$$

Assuming that a negative radicand did not occur, the value of s_1 is halved, and new values for r_3, r_j, r_1, and Δr are computed. If no sign change in Δr occurs, then s_1 is again halved and the procedure repeated until a polarity change in Δr occurs. When the sign change does occur, the last selected value of s_1 is increased by one half its value and the polarity of the new resulting Δr is tested. This interval-halving procedure, increasing or decreasing s_1 by one half of each increment taken, is repeated until the difference Δr is less than the desired error level ϵ.

This procedure is based on the fact that Eqs. (1) and (2) represent Δr as a continuous function of s_1. If there are two values of s_1, one of which yields a positive value of Δr and the other a negative value, then there is some value of s_1 between these two values for which Δr is zero - the desired condition.

During the first calculation of Δr, if a negative radicand results, then a special procedure must be followed after the value of s_1 which yields real values is found. Whereas in the first case it is known that the correct value of s_1 is not larger than r_M, in this case the solution could be larger than the value of s_1 for which real (but incorrect) values of r_3, r_p, and r_1 resulted. This ambiguity is resolved by performing one pass through Eqs. (1) and (2) with s_1 equal to zero. Comparison of the sign of the resulting Δr with the sign of the previous Δr indicates whether s_1 should be increased or decreased. This same procedure is used if, during an increase in s_1, a negative radicand occurs.

The details of the iterative procedure are shown in the flow diagrams of Section 6.

5.4 Source Position Vectors

The position vectors of the three selected sources are obtained from

$$\underline{r}_{iS} = r_i \begin{pmatrix} \cos e_i \quad \cos a_i \\ \cos e_i \quad \sin a_i \\ \sin e_i \end{pmatrix} \qquad (i = 1, 3, p)$$

where the subscript S denotes shuttle coordinates.

5.5 Transformation Matrix

The transformation matrix M from shuttle to target vehicle coordinates is computed from

$$M^T = AB$$

where

$$A = (\underline{r}_{13S} \quad \underline{r}_{1pS} \quad \underline{r}_{13S} \times \underline{r}_{1pS})$$

$$B = (\underline{r}_{13T} \quad \underline{r}_{1pT} \quad \underline{r}_{13T} \times \underline{r}_{1pT})^{-1}$$

$$= \left[\frac{1}{S^2}\begin{pmatrix} -C \\ -\frac{C^2}{S} \\ \frac{1}{S} \end{pmatrix} \quad \frac{1}{KS^2}\begin{pmatrix} C \\ \frac{C^2 - S^2}{S} \\ -\frac{C^2}{S} \end{pmatrix} \quad \frac{1}{K\rho S^2}\begin{pmatrix} 1 \\ \frac{C}{S} \\ -\frac{C}{S} \end{pmatrix} \right]$$

$$C = \cos 40^\circ$$

$$S = \sin 40^\circ$$

$$K = \begin{cases} 0.4 & \text{if } p = 2 \\ 2.5 & \text{if } p = 4 \end{cases}$$

$$\underline{r}_{13S} = \underline{r}_{3S} - \underline{r}_{1S}$$

$$\underline{r}_{1pS} = \underline{r}_{pS} - \underline{r}_{1S}$$

5.6 Rotation Angles

The rotation angles γ_1, γ_2, and γ_3 are obtained from

$$\gamma_2 = \sin^{-1}(m_{31})$$

$$\gamma_3 = -\sin^{-1}\left(\frac{m_{21}}{\cos\gamma_2}\right)$$

$$\gamma_1 = -\sin^{-1}\left(\frac{m_{32}}{\cos\gamma_2}\right)$$

where m_{31}, m_{21}, m_{32} are elements of M according to

$$M = \begin{pmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{pmatrix}$$

5.7 Relative Position Vector Between Docking Hatches

The position of the target vehicle docking hatch relative to the shuttle docking hatch is computed from

$$\underline{r}_S = - \underline{y}_S + \underline{r}_{1S} + M^T \underline{z}_T$$

where $\underline{y}$ and $\underline{z}$ are the locations of the shuttle and target vehicle docking hatches relative to their respective coordinate system origins, and the S and T subscripts indicate shuttle and target vehicle coordinates. Note that $\underline{y}_S$ and $\underline{z}_T$ are fixed constants.

5.8 Velocity and Attitude Rate

The estimated relative velocity and estimated relative attitude rate of the two vehicles are computed by differencing the current relative position and attitude solution with the solution f cycles in the past as follows:

$$\dot{\gamma}_i = \left[\gamma_i(t) - \gamma_i(t - f\Delta t) \right] / f\Delta t \qquad (i = 1, 2, 3)$$

$$\dot{\underline{r}}_S = \left[\underline{r}_S(t) - \underline{r}_S(t - f\Delta t) \right] / f\Delta t$$

9.10.1 Docking and Undocking Navigation (con't)

During the first cycle, no rate information can be computed, and during cycles 2 through f, the current and the first solutions are used in the calculations.

This procedure provides smoother estimates of the rates from cycle to cycle than if successive values of relative position and attitude were used.

6. DETAILED FLOW DIAGRAMS

This section contains detailed flow diagrams of the Docking and Undocking Navigation Program.

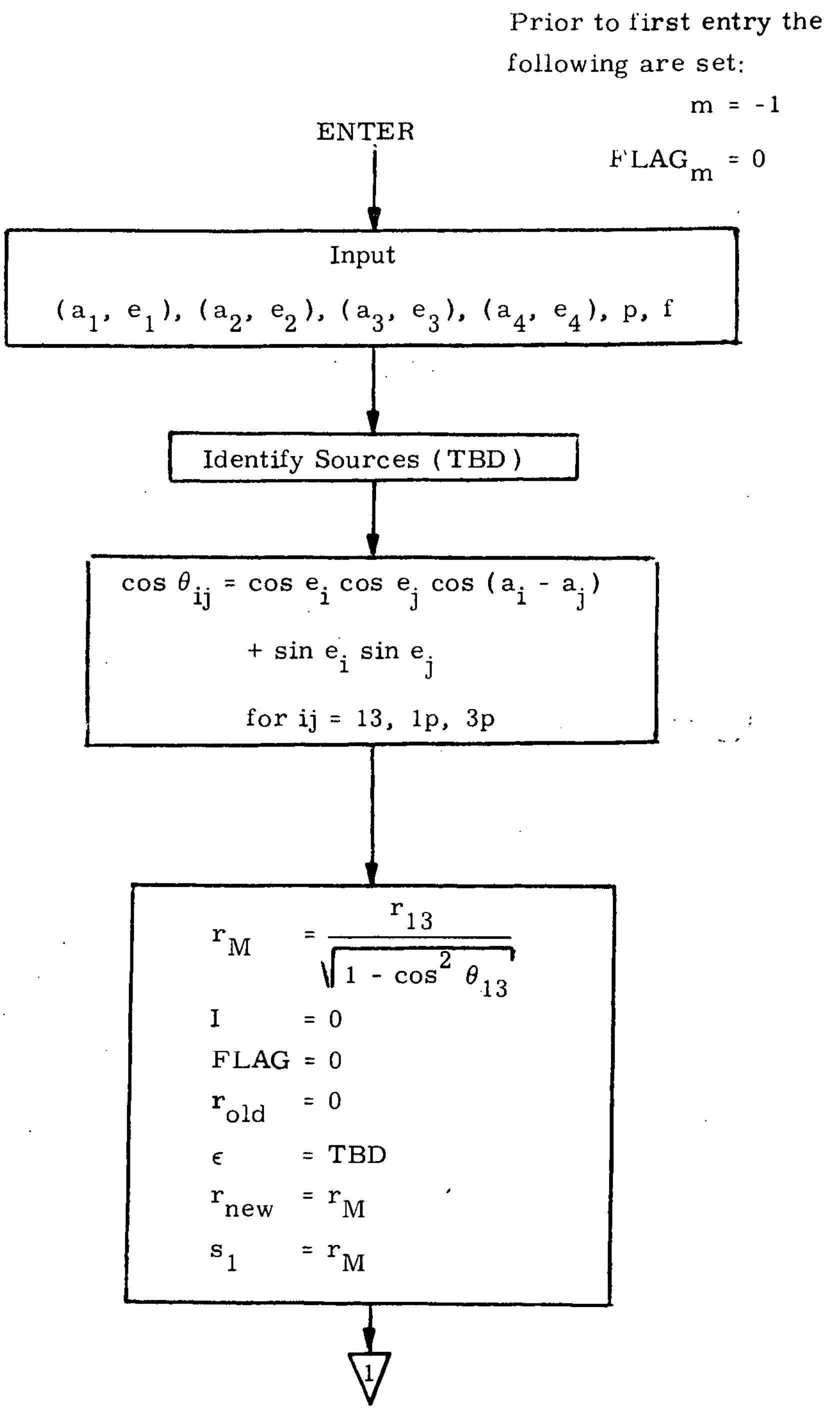

Prior to first entry the following are set:
$m = -1$
$FLAG_m = 0$
ENTER
Input
$(a_1, e_1), (a_2, e_2), (a_3, e_3), (a_4, e_4), p, f$
Identify Sources (TBD)
$\cos \theta_{ij} = \cos e_i \cos e_j \cos (a_i - a_j) + \sin e_i \sin e_j$
for ij = 13, 1p, 3p
$r_M = \frac{r_{13}}{\sqrt{1 - \cos^2 \theta_{13}}}$
$I = 0$
$FLAG = 0$
$r_{old} = 0$
$\epsilon = TBD$
$r_{new} = r_M$
$s_1 = r_M$
1

Figure 5a DETAILED FLOW DIAGRAM

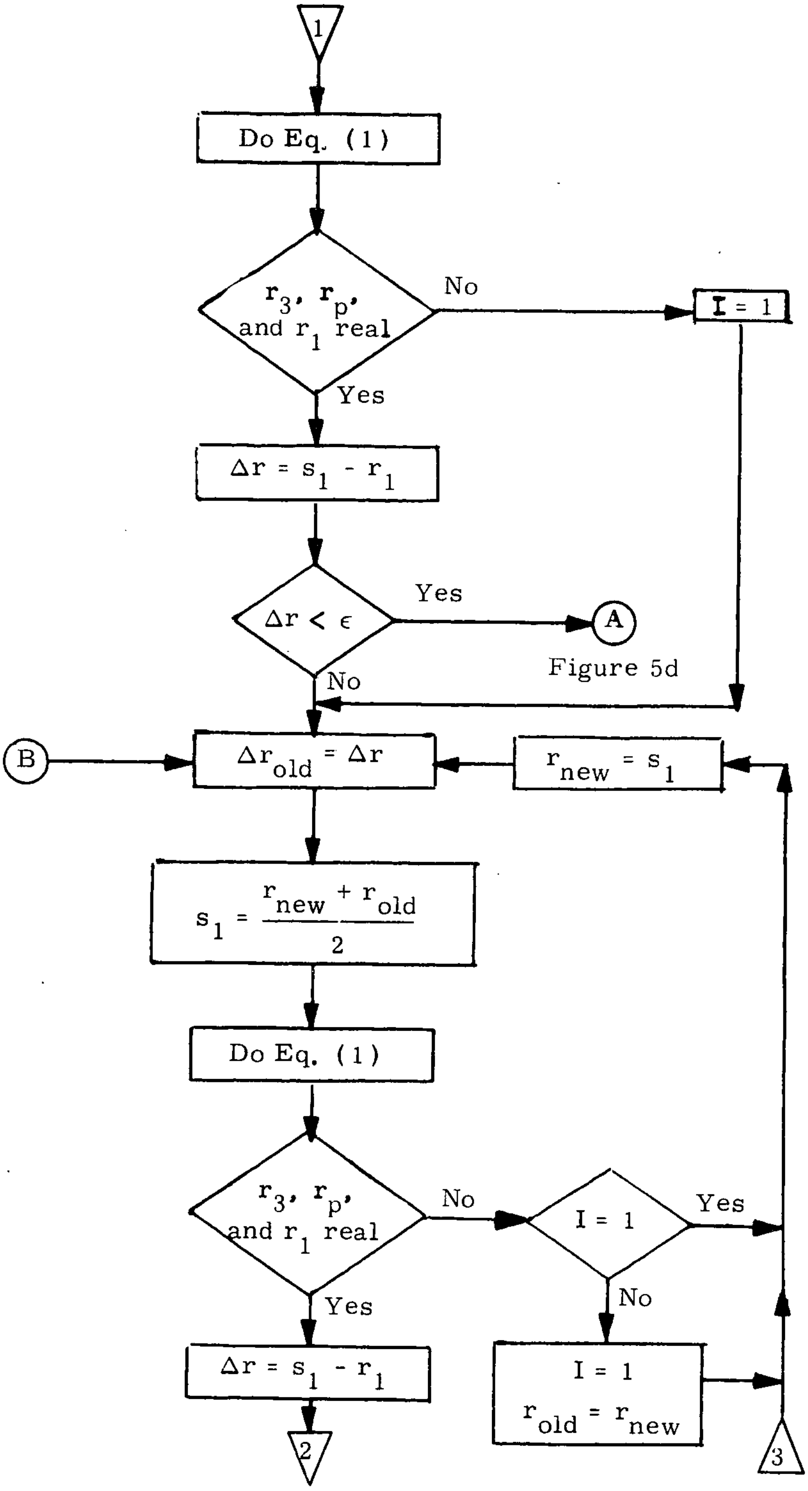


Figure 5b DETAILED FLOW DIAGRAM

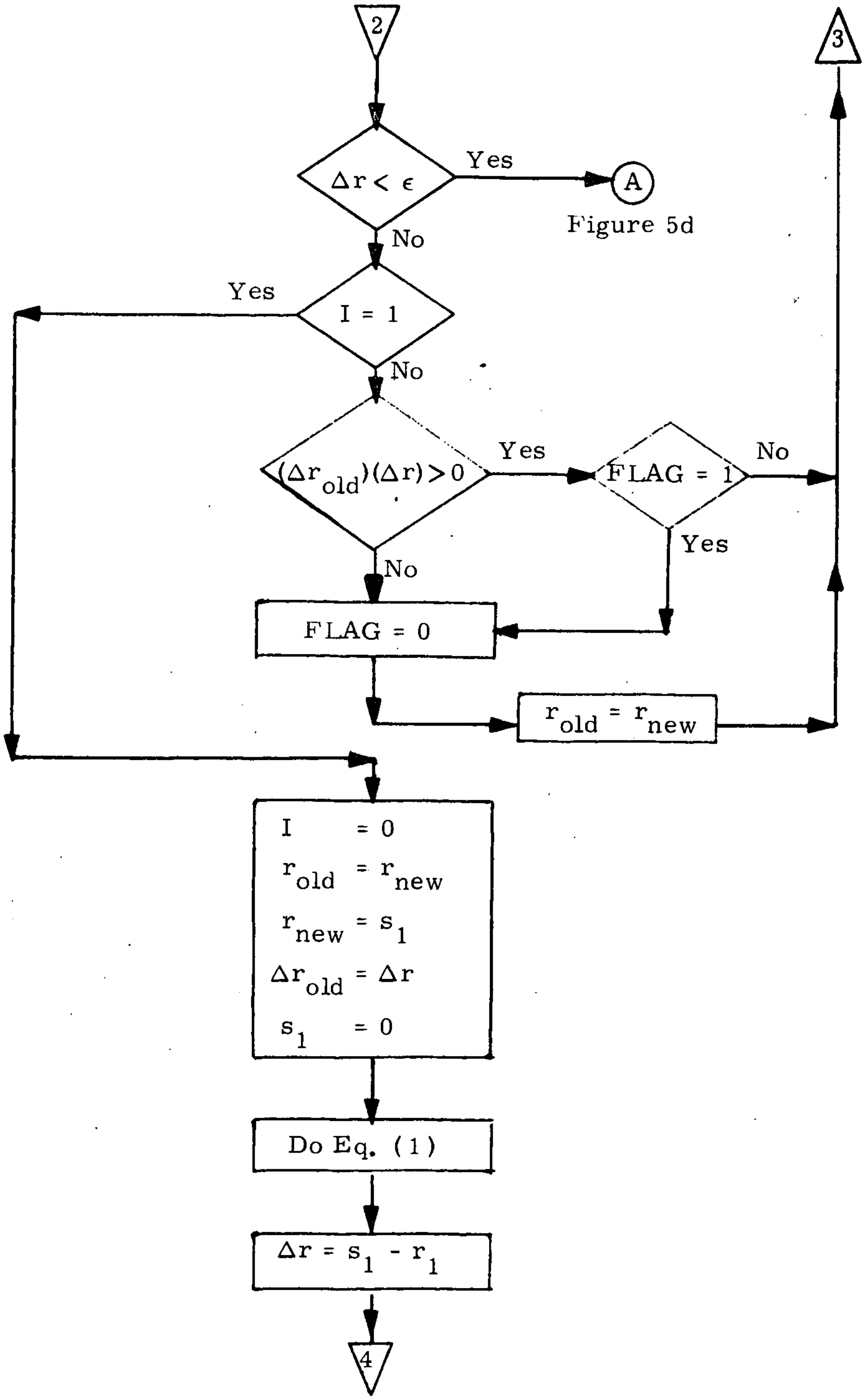


Figure 5c DETAILED FLOW DIAGRAM

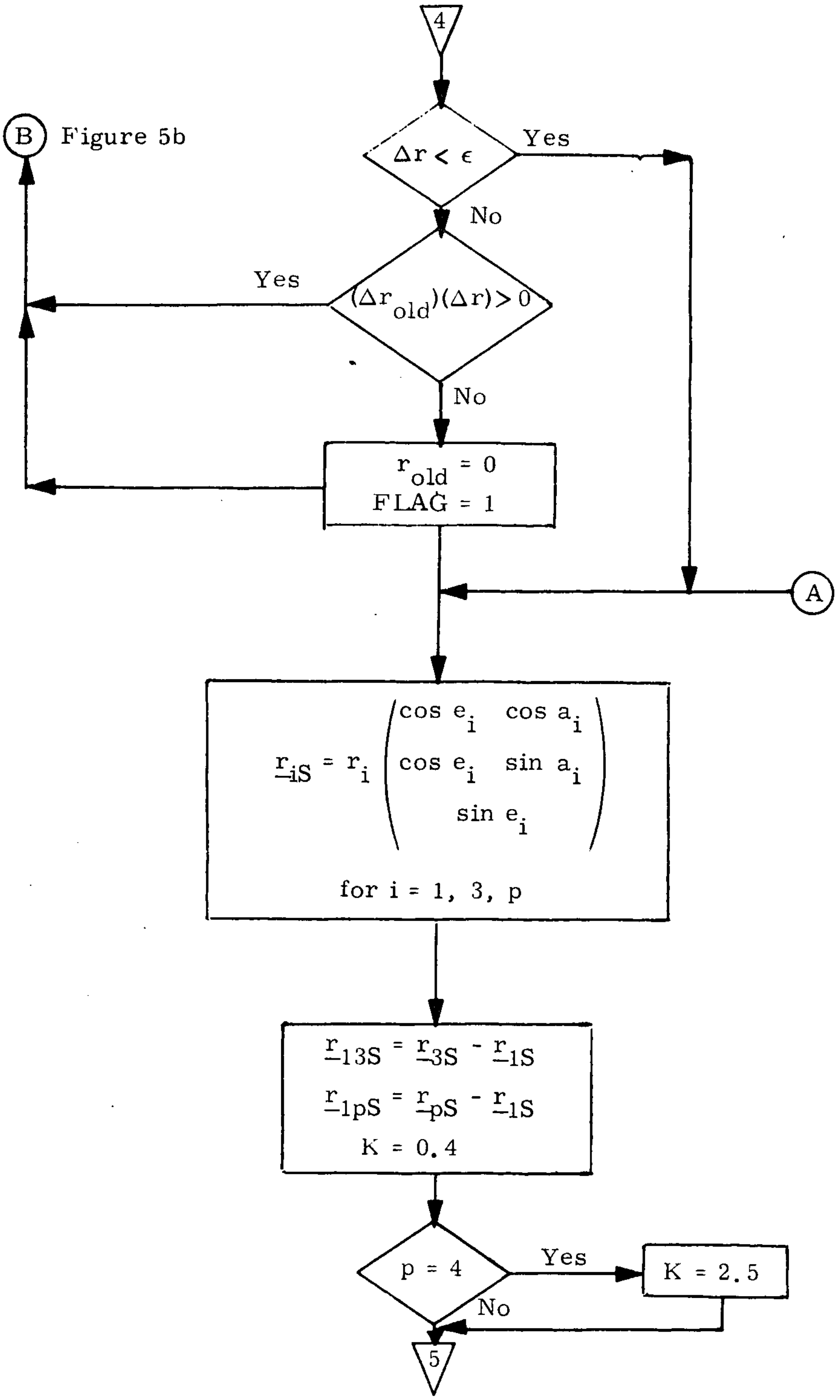
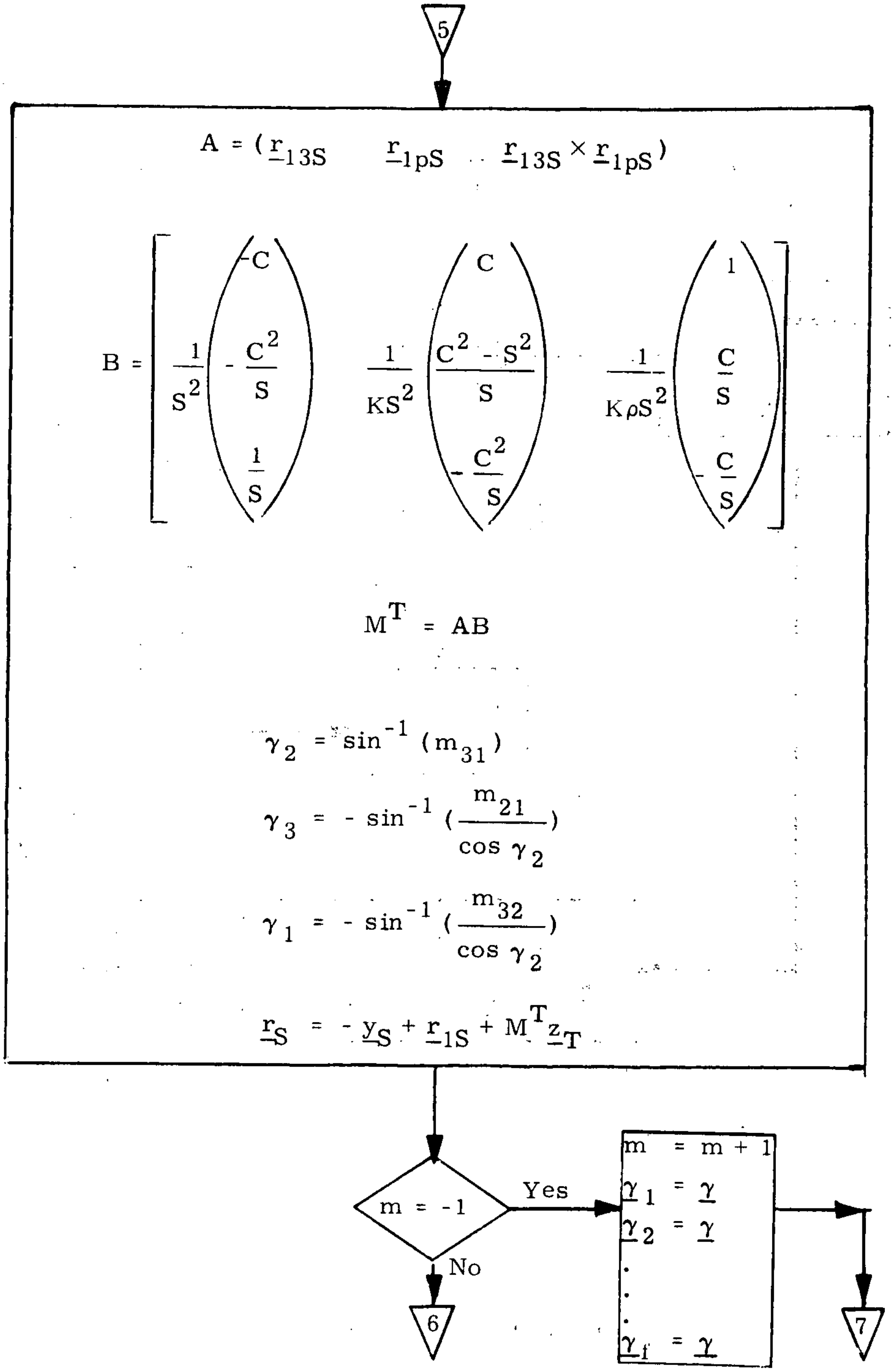

4
B
Figure 5b
$\Delta r < \epsilon$
Yes
No
Yes
$(\Delta r_{old})(\Delta r) > 0$
No
$r_{old} = 0$
FLAG = 1
A
$\underline{r}_{iS} = r_i \begin{pmatrix} \cos e_i & \cos a_i \\ \cos e_i & \sin a_i \\ \multicolumn{2}{c}{\sin e_i} \end{pmatrix}$
for i = 1, 3, p
$\underline{r}_{13S} = \underline{r}_{3S} - \underline{r}_{1S}$
$\underline{r}_{1pS} = \underline{r}_{pS} - \underline{r}_{1S}$
K = 0.4
p = 4
Yes
K = 2.5
No
5

Figure 5d DETAILED FLOW DIAGRAM



$$A = (\underline{r}_{13S} \quad \underline{r}_{1pS} \quad \underline{r}_{13S} \times \underline{r}_{1pS})$$

$$B = \left[\frac{1}{S^2}\begin{pmatrix} -C \\ -\frac{C^2}{S} \\ \frac{1}{S} \end{pmatrix} \quad \frac{1}{KS^2}\begin{pmatrix} C \\ \frac{C^2 - S^2}{S} \\ -\frac{C^2}{S} \end{pmatrix} \quad \frac{1}{K\rho S^2}\begin{pmatrix} 1 \\ \frac{C}{S} \\ -\frac{C}{S} \end{pmatrix} \right]$$

$$M^T = AB$$

$$\gamma_2 = \sin^{-1}(m_{31})$$

$$\gamma_3 = -\sin^{-1}\left(\frac{m_{21}}{\cos\gamma_2}\right)$$

$$\gamma_1 = -\sin^{-1}\left(\frac{m_{32}}{\cos\gamma_2}\right)$$

$$\underline{r}_S = -\underline{y}_S + \underline{r}_{1S} + M^T\underline{z}_T$$

$m = -1$

$m = m + 1$

$\underline{\gamma}_1 = \underline{\gamma}$

$\underline{\gamma}_2 = \underline{\gamma}$

$\vdots$

$\underline{\gamma}_f = \underline{\gamma}$

Figure 5e DETAILED FLOW DIAGRAM

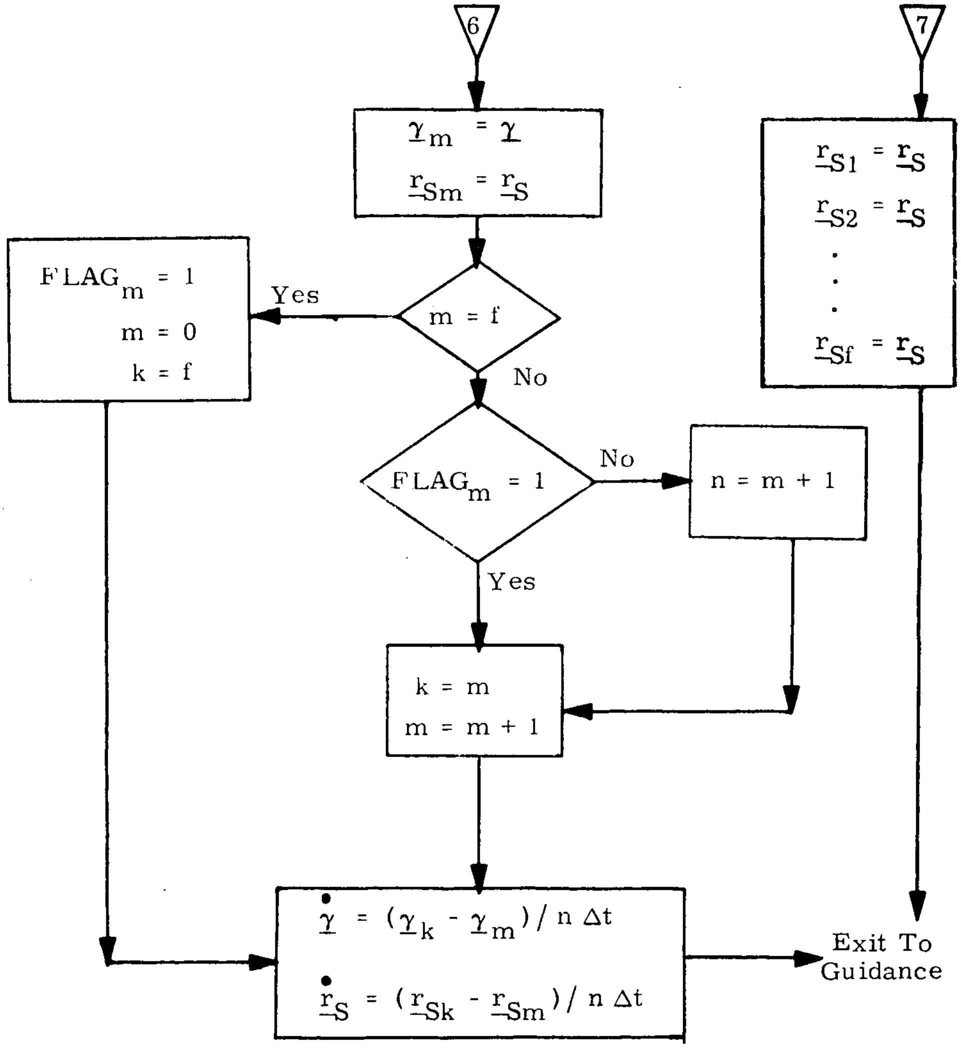

6
7
$\underline{\gamma}_m = \underline{\gamma}$
$\underline{r}_{Sm} = \underline{r}_S$
$\underline{r}_{S1} = \underline{r}_S$
$\underline{r}_{S2} = \underline{r}_S$
$\underline{r}_{Sf} = \underline{r}_S$
$m = f$
Yes
No
FLAG$_m$ = 1
m = 0
k = f
FLAG$_m$ = 1
No
Yes
n = m + 1
k = m
m = m + 1
$\dot{\underline{\gamma}} = (\underline{\gamma}_k - \underline{\gamma}_m)/n\,\Delta t$
$\dot{\underline{r}}_S = (\underline{r}_{Sk} - \underline{r}_{Sm})/n\,\Delta t$
Exit To Guidance

Figure 5f DETAILED FLOW DIAGRAM

7. SUPPLEMENTARY INFORMATION

The Docking and Undocking Navigation Program described in this report has been operated as an open loop, simulating the sensor-target geometry, the sensor, and the computations yielding as outputs the relative state vector and attitude between vehicles. The program is valid and the configuration chosen performs as expected. A chart and tabulated results appear in Ref. 1.

It is planned to continue the present program effort to provide a closed loop capability which will include a guidance law* for Docking and Undocking, and an autopilot with capability to operate with the guidance law and the vehicle and engine characteristics. The navigation program will be modified to incorporate Kalman Filtering which should enhance the navigation and provide better assessment of the relative state vector. It is also planned to add a scale change or zoom capability to the sensor model used such that improvement in the accuracy of the state vector can be achieved at long ranges.

*A simplified guidance law will be implemented initially with growth to more sophisticated guidance laws as deemed necessary.

9.10.1 Docking and Undocking Navigation

Reference

1. Blanchard, Earle P., NAS 9-10268 Automatic Docking GN& C Equation Development, 21 December 1970, 70-408L-7.

SPACE SHUTTLE

GN&C SOFTWARE EQUATION SUBMITTAL

Software Equation Section Docking and Undocking Submittal No. 3

Function Automatic Docking Control Law

Module No. OC4 Function No. -4, -6 (MSC 03690).

Submitted By: E. T. Kubiak Co. MSC/GCD
(Name)

Date: January 26, 1971

NASA Contact: W. H. Peters Organization EG2
(Name)

Approved by Panel III K. J. Cox K.J.Cox Date January 26, 1971

Summary Description: The automatic docking control laws provide the attitude and translational commands for the docking procedure which is defined to begin at a range of 1000 ft. The procedure involves two sequential control tasks. The first brings the orbiter within stationkeeping range (≈150 ft.) and the second accomplishes docking with minimum docking hardware contact position dispersions.

Shuttle Configuration: (Vehicle, Aero Data, Sensor, Et Cetera) No docking sensor configuration is defined but jet accelerations are assumed.

Comments:

(Design Status) The design is in the conceptual stage with required filters still to be designed.

(Verification Status) Will be simulated on an orbiter docking engineering simulator.

Panel Comments: The range, range rate, and relative attitude computations in these equations must be coordinated with similar computations in the Docking and Undocking Navigation equations.

9.10.2 Automatic Docking Control Law

1. Introduction

The docking procedure is defined to begin at a range of approximately 1000 feet. From this point, there are two sequential control tasks. The first task is to bring the orbiter within stationkeeping range, say 150 feet, with a lateral displacement of 10 feet or less from the desired approach path and relative rates of one half ft/sec/axis or less. The second control task is a successful docking with minimum docking hardware contact position dispersion and transmitted impulses.

Significant improvements over the original control law (Reference 1) are (1) minimum use of relative angle measurements which have large errors, (2) direct control of the probe tip which provides tighter control, and (3) reduced time for the docking procedure due to improved logic. The first two points are also discussed in the reference.

In generating this control law, the following assumptions have been used as ground rules:

a) Measured quantities available from the sensors are range, R; LOS (line-of-sight) angles for pitch, α, and yaw, β; and relative orbiter/target attitude (ϕ_R, θ_R, Ψ_R). As the orbiter is to be autonomous, no other information (e.g., target position or attitude) is available from ground tracking or computer initialization.

b) Range and LOS angle measurements will have greater accuracy than relative attitude angle measurements (particularly at longer ranges).

c) It is desirable to have at least a brief station keeping period prior to the final phase (assumed to begin at 100 ft range) of docking, providing the opportunity for a final check of thrusters, docking mechanisms, GN&C systems, and sensor systems.

d) The docking procedure begins at approximately 1000 ft range and should conclude in 5 to 10 minutes (plus any time spent in the station keeping mode).

e) Sensor measurements provide the only available information with regards to the passive vehicle's relative state (no data link).

f) It is assumed that the target vehicle is under attitude control and that any target vehicle motion due to attitude control limit cycling is negligible (a good assumption for CMG control).

Nomenclature

a	Translational acceleration
α	LOS pitch angle
β	LOS yaw angle
K	Factor in phase-plane switching lines representing the relative importance of time vs. fuel minimization
ℓ	Distance along +X body axis from orbiter c.g. to sensor location
LOS	Line-of-sight
ϕ_R	Relative orbiter/target roll attitude
Ψ_R	Relative orbiter/target yaw attitude
R	Range
T	Total closure time
θ_R	Relative orbiter/target pitch attitude
u_Ψ	$\pm\Psi$ yaw torques
u_θ	$\pm\theta$ pitch torques
u_y	$\pm y$ thruster forces
u_z	$\pm Z$ thruster forces
$\underline{\omega}_{BODY}$	Orbiter body rate

9.10.2 Automatic Docking Control Law (con't)

$\underline{\omega}_{LOS}$	LOS angular rate
X_0	Initial separation distance
$\dot{X}_0$	Initial closing rate
X_{cg}, Y_{cg}, Z_{cg}	C.G. position errors
X_{LCS}, Y_{LCS}, Z_{LCS}	Position errors in LOS coordinate system
X_p, Y_p, Z_p	Probe position errors

2. Coordinate System Definition

Before proceeding to equation formulation, the following coordinate systems must be defined (see also Figure 1).

a) Body coordinate system (BCS) - origin at c.g. of +X axis towards nose along centerline, +Y towards right wing, +Z down.

b) Sensor coordinate system (SCS) - sensor and docking mechanism location assumed coincident along +X body axis at distance ℓ from orbiter c.g., which also defines the origin location. Direction of axes, same as body axes.

c) LOS Coordinate System (LCS) - origin same as the SCS. Direction of axes defined by LOS pitch and yaw rotations from SCS +X axis.

d) Target coordinate system (TCS) - origin located at passive vehicle docking mechanism assumed coincident with reflectors. -X axes defines the desired final approach path. Y and Z complete the right hand system.

3. Functional Flow Diagram

The sequencing of functions performed by the Automatic Docking Control Law is described in this section and illustrated in the functional flow diagram in Figures 2a and 2b.

The program calculates the probe to target vector and determines whether Phase 1 or Phase 2 control is desired. If Phase 1 control is required, calculate the position and velocity errors for phase-plane control using the sensor measured pitch and yaw LOS angles, c.g. to target range, and the estimated vehicle to target attitude. Based on these values for position and velocity errors, enter the X, Y, and Z-axis phase-plane control logic and compute translational commands.

For Phase 2 control, compute the range position error using the sensor measured pitch and yaw LOS angles, c.g. to target range, and the estimated vehicle to target attitude. Passing this signal through a

9.10.2 Automatic Docking Control Law (cont'd)

(Y-Axes out of page)

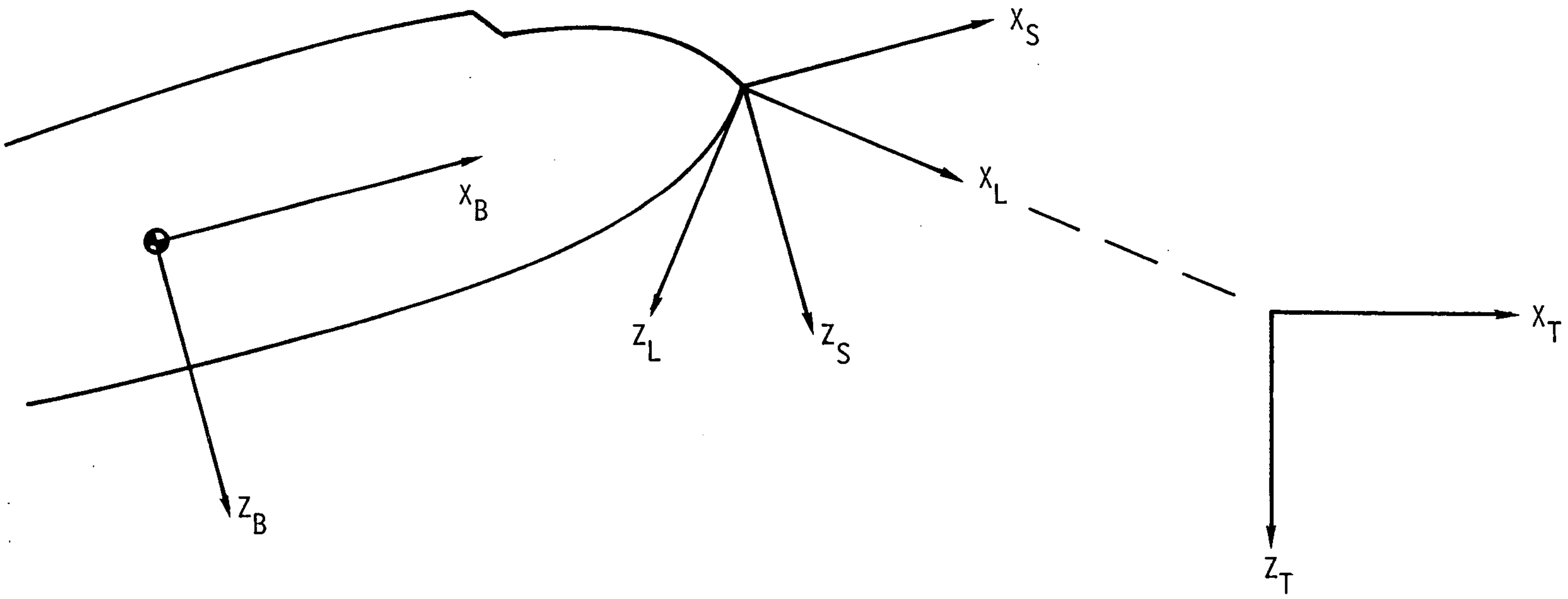


Figure 1

Coordinate System Definition

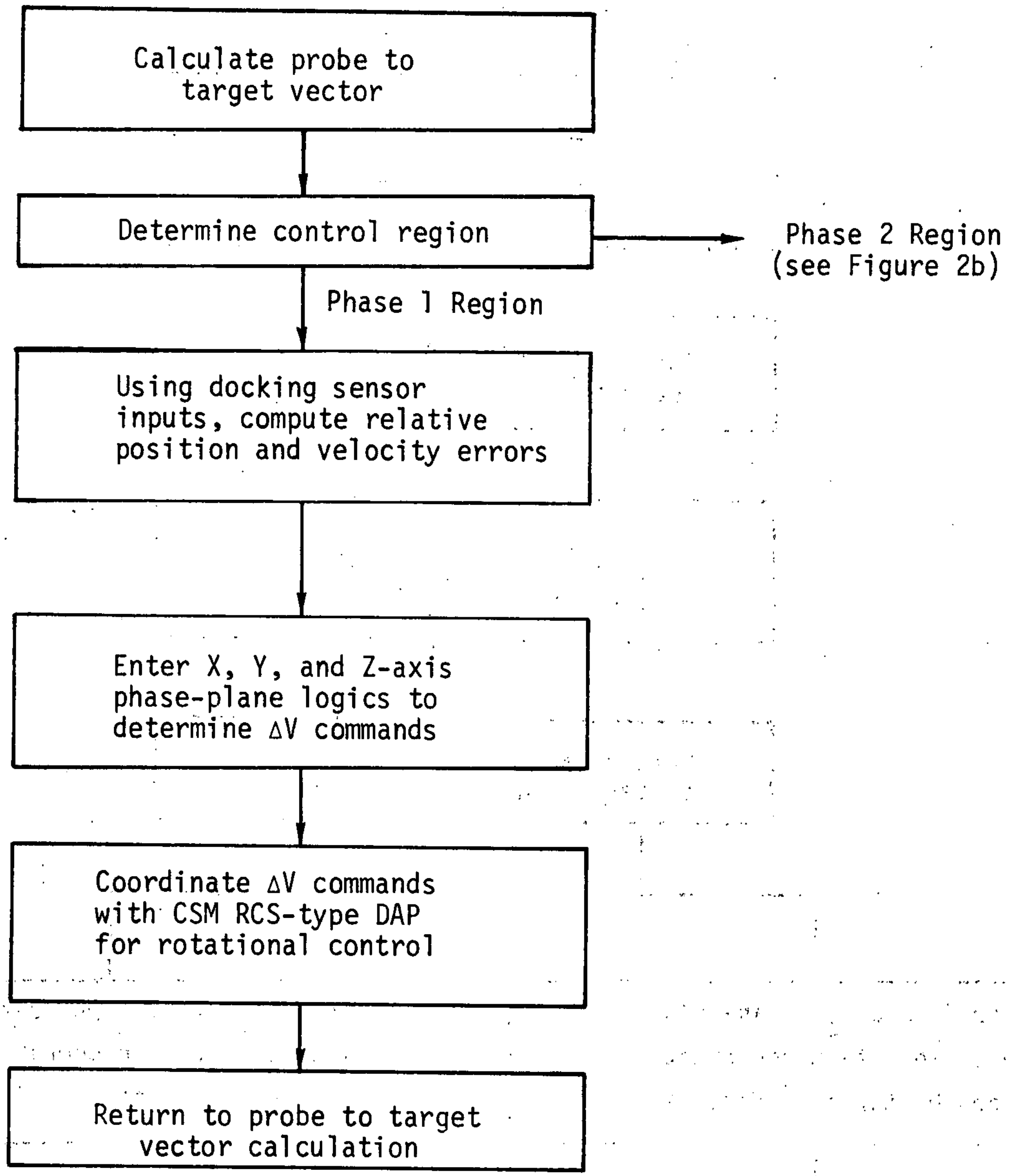

Calculate probe to target vector
Determine control region
Phase 2 Region (see Figure 2b)
Phase 1 Region
Using docking sensor inputs, compute relative position and velocity errors
Enter X, Y, and Z-axis phase-plane logics to determine ΔV commands
Coordinate ΔV commands with CSM RCS-type DAP for rotational control
Return to probe to target vector calculation

Figure 2a. Phase I Control Functional Flow Diagram

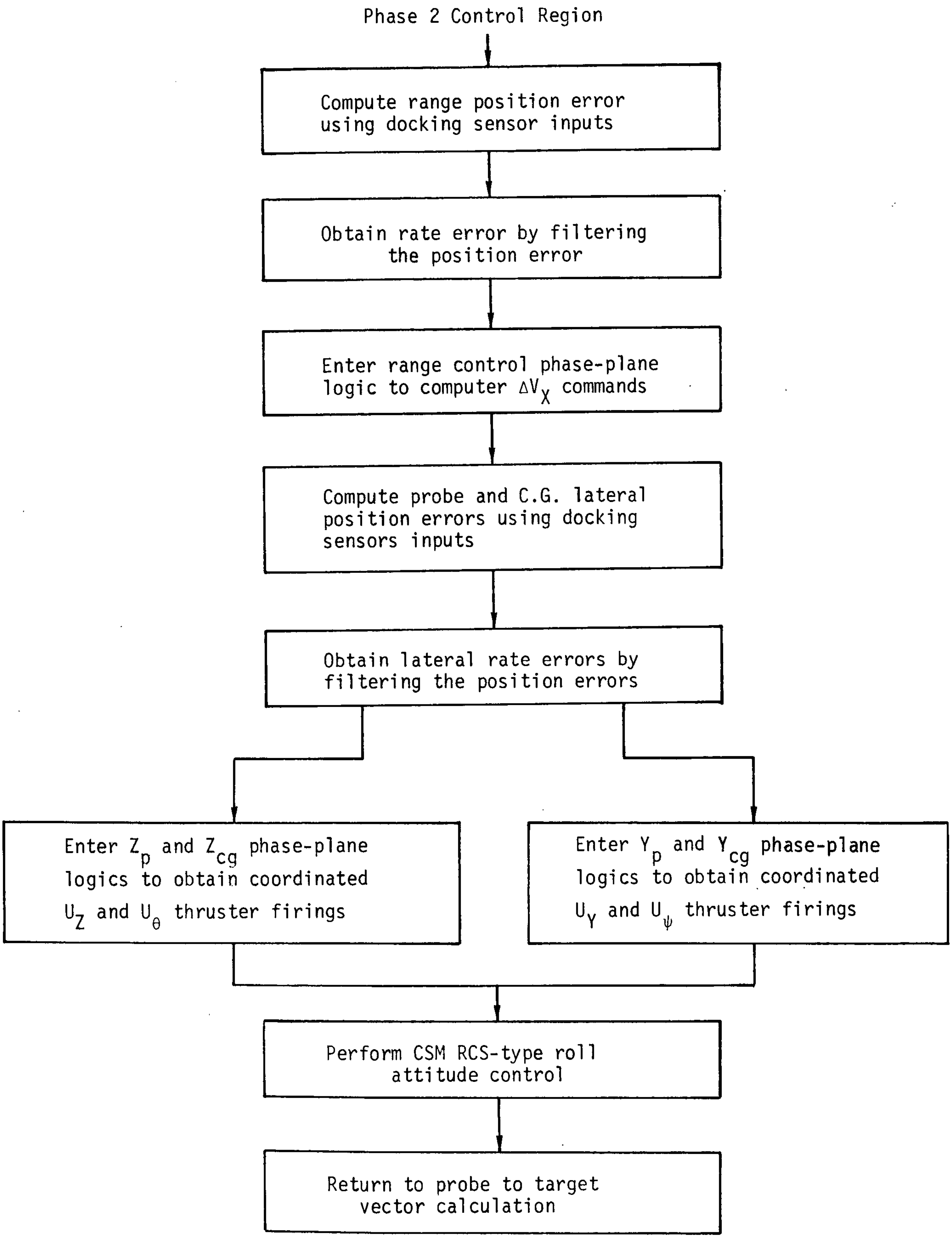

Phase 2 Control Region
Compute range position error using docking sensor inputs
Obtain rate error by filtering the position error
Enter range control phase-plane logic to computer ΔV_X commands
Compute probe and C.G. lateral position errors using docking sensors inputs
Obtain lateral rate errors by filtering the position errors
Enter Z_p and Z_{cg} phase-plane logics to obtain coordinated U_Z and U_θ thruster firings
Enter Y_p and Y_{cg} phase-plane logics to obtain coordinated U_Y and U_ψ thruster firings
Perform CSM RCS-type roll attitude control
Return to probe to target vector calculation

Figure 2b. Phase II Control Functional Flow Diagram

filter, obtain the rate error. Enter the range control phase-plane logic to compute jet firing times for X-axis translation control. Compute the lateral position errors of the probe and c.g. and determine the respective rates by a filtering routine. Enter the c.g. and probe phase-plane logics to obtain coordinated ±Z thruster firings and $\pm\theta$ pitch torques for Z-axis and pitch control, and coordinated ±Y thruster firings and $\pm\Psi$ yaw torques for Y-axis and yaw control.

4. Program Input-Output

The docking sensors have not been baselined, but in this development, basic inputs have been identified. These inputs include range, LOS pitch and yaw angles, relative orbiter/target attitude, body rates, the distance between the orbiter c.g. and probe as measured along the +X body axis, and estimates of the RCS jet control authorities. The outputs of the program are RCS jet firing times.

Input Parameters

R	Range between orbiter and target vehicle
α	LOS pitch angle
β	LOS yaw angle
ϕ_R	Relative orbiter/target roll angle
θ_R	Relative orbiter/target pitch angle
Ψ_R	Relative orbiter/target yaw angle
$\underline{\omega}_{BODY}$	Orbiter angular rates
ℓ	Distance between orbiter c.g. and probe as measured along +X body axis
a	Translational acceleration capability of the orbiter (lateral and ±X body)
U_Y, U_Z	RCS translational acceleration along Y and Z axes

u_θ, u_ψ	RCS angular acceleration about pitch and yaw axes

Output Parameters

$t_{Region\ X}$	RCS jet firing time (and sign) for various regions of the phase-plane logics

5. Description of Equation

5.1 Phase I Control

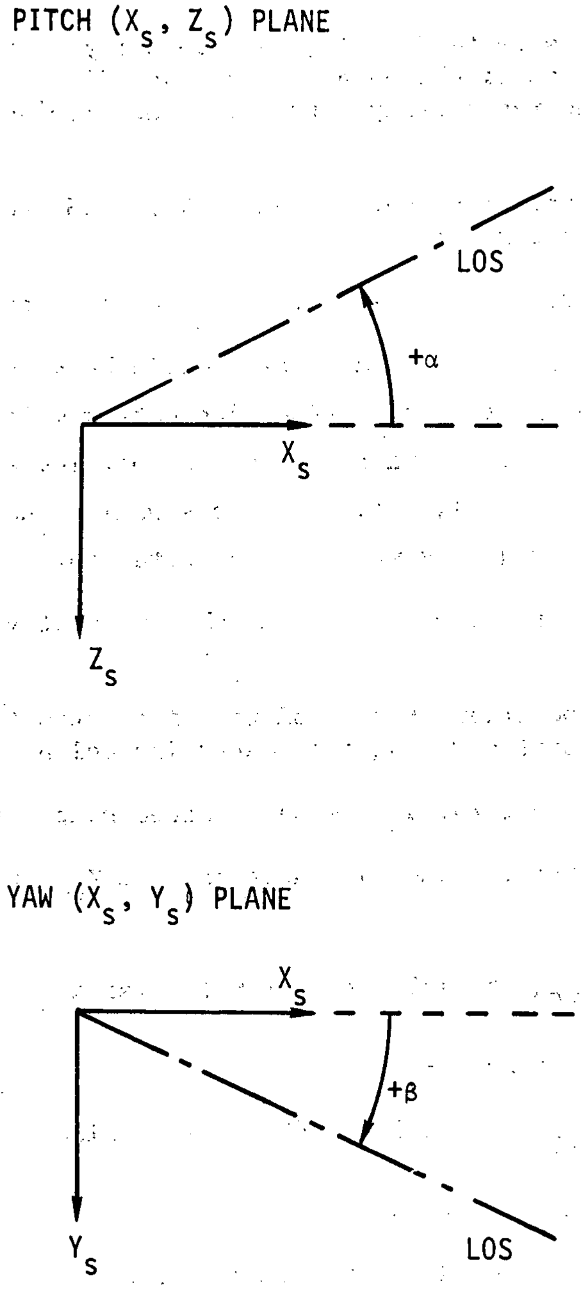
Phase 1 is defined as the control period during which the orbiter is brought from some post rendezvous state (range about 1000 feet) into the stationkeeping state. In the sequence of control actions, the first step is to define as a pitch/yaw reference, the LOS vector from the sensor to the target (i.e., $\alpha = \beta = 0$, see Figure 3).

The roll reference is defined such that Z is parallel to Z_T (i.e., the relative roll angle is zero). The attitude error, (ϕ_R, α, β) will change slowly due to relative motion and vehicle body rotation. This error will be measured and filtered once per second. Control logic will be basically the same as the CSM RCS DAP with a deadband of 5°. When the vehicle's attitude is within the deadband for all three axis translational control is begun.

In the translational control formulation the TCS is considered to be inertial (orbiter mechanics neglected). The control problem is to translate the orbiter from its initial state to a limit cycle region which has as its position reference (-150, 0, 0) in the TCS. The ideal trajectory, time and fuel-wise, is the straight line between the initial condition and (-150, 0, 0) in the TCS. One of the more precise control processes which could be used to follow this trajectory is:

a) Generate displacement and rate vector in the TCS from measurements and matrix computations.

9.10.2 Automatic Docking Control Law (cont'd)



PITCH (X_s, Z_s) PLANE

YAW (X_s, Y_s) PLANE

Figure 3

LOS Angle Definition

b) Select a delta V to (1) null velocity component normal to displacement vector, and (2) provide the desired closing rate along the displacement vector.

c) Determine components of delta V in BCS and implement commands.

d) Reiterate computations to null residual errors.

The performance of such a process would be very dependent on the relative angle measurements used in numerous matrix multiplications. As these measurements are not highly accurate, particularly at initialization range (1000 ft or more), another process will be used which performs the same function and requires much less computation.

a) Compute position error and vehicle relative rates in LCS.

b) Input position errors and relative rates to phase plane switching logic to determine delta V commands

c) Recycle according to some selected sample frequency.

For this scheme, Figure 4a shows how the Z_{LCS} position error is determined to be

$$Z_{LCS} = 150 \sin (\alpha + \theta_R) - \ell \sin\alpha$$

Similarly, Figure 4b indicates

$$Y_{LCS} = -150 \sin (\beta + \Psi_R) + \ell \sin\beta$$

Finally, the X position error is

$$X_{LCS} = R + \ell \cos\theta_R \cos\Psi_R - 150$$

The relative velocity of the orbiter with respect to the target vehicle in the LCS is equal to the negative of the derivative of $(\underline{R} + \underline{\ell})$. As $\underline{R}$ is rotating in inertial space with an angular velocity of $(\omega_{LOS} + \omega_{BODY})$, the expression for the derivative is

$$\frac{d}{dt} (\underline{R} + \underline{\ell})_{LOS} = \dot{\underline{R}} + (\underline{\omega}_{LOS} + \underline{\omega}_{BODY}) \times \underline{R} + \underline{\omega}_{BODY} \times \underline{\ell}$$

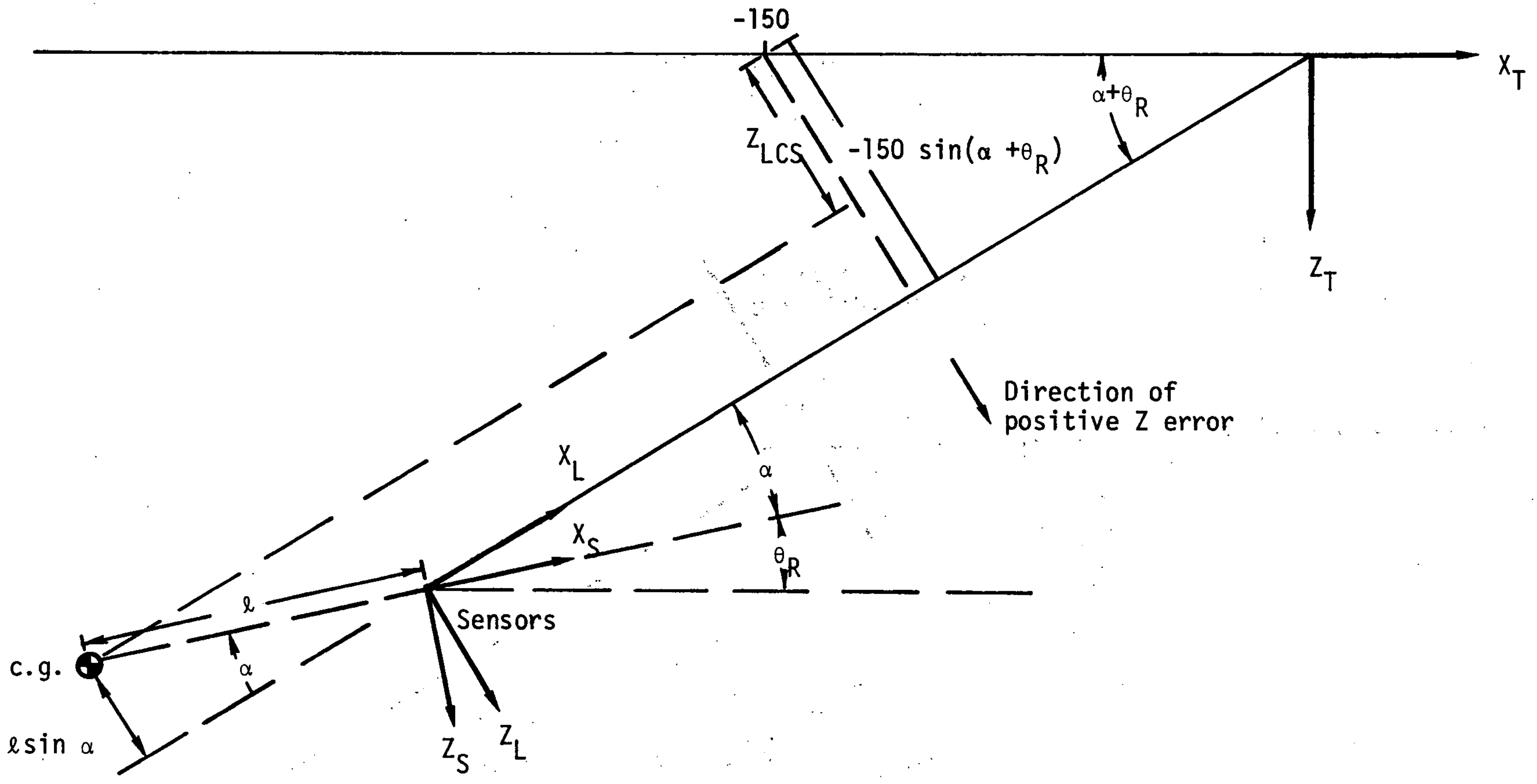


Figure 4a

Calculation of Z_{LCS}

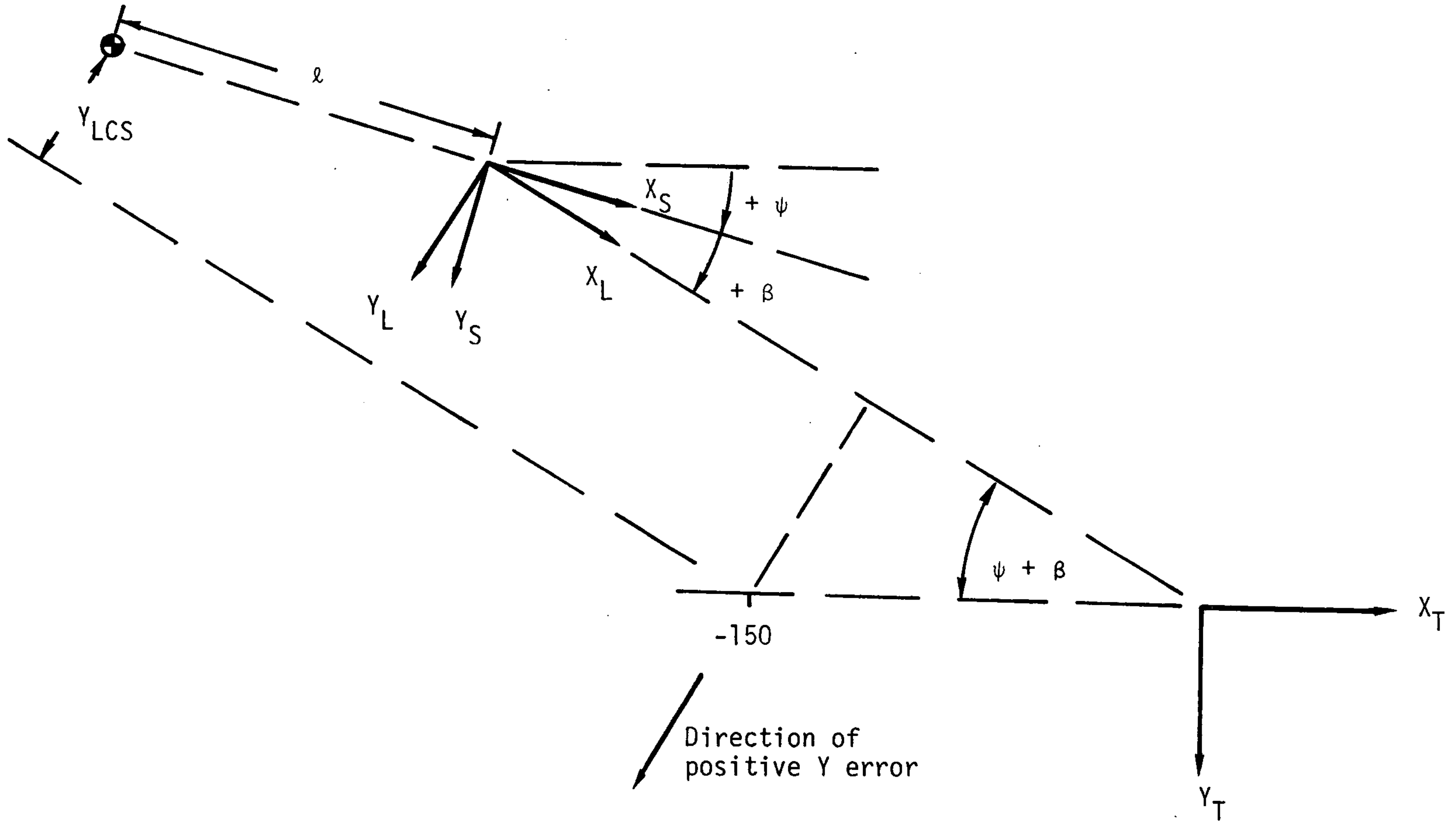


Figure 4b

Calculation of Y_{LCS}

Assuming $\underline{\ell}$ is colinear with the X_{LCS} axis, then $\underline{\ell} = [\ell \ 0 \ 0]^T$

Also,

$$\underline{R} = \begin{bmatrix} R \\ 0 \\ 0 \end{bmatrix} \qquad \underline{\omega}_{LOS} = \begin{bmatrix} 0 \\ \alpha \\ \beta \end{bmatrix} \qquad \underline{\omega}_{BODY} = \begin{bmatrix} \phi \\ \theta \\ \Psi \end{bmatrix}$$

which yields the following derivatives

$$\dot{X}_{LCS} = -\dot{R}$$

$$\dot{Y}_{LCS} = -R(\dot{\Psi}_R + \dot{\beta}) - \ell\dot{\Psi}_R$$

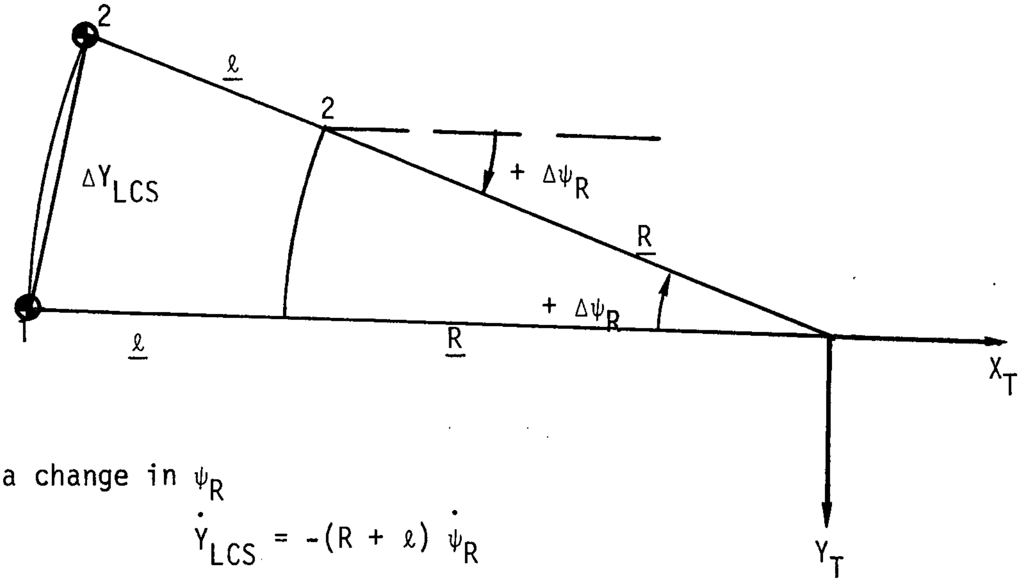
$$\dot{Z}_{LCS} = R(\dot{\theta}_R + \dot{\alpha}) + \ell\dot{\theta}_R$$

Figures 5a and 5b show the geometry relating to the $\dot{Y}_{LCS}$ and $\dot{Z}_{LCS}$ equations. It should be noted in the position and rate equations that R and ℓ are always positive quantities.

Also, as $\dot{\theta}_{BODY}$ and $\dot{\Psi}_{BODY}$ very nearly equal $\dot{\theta}_R$ and $\dot{\Psi}_R$ and further as the body rates may be known much more accurately than the relative angle rates, θ_R and Ψ_R may be replaced in the Y_{LCS} and Z_{LCS} computations by $\dot{\theta}_{BODY}$ and $\dot{\Psi}_{BODY}$.

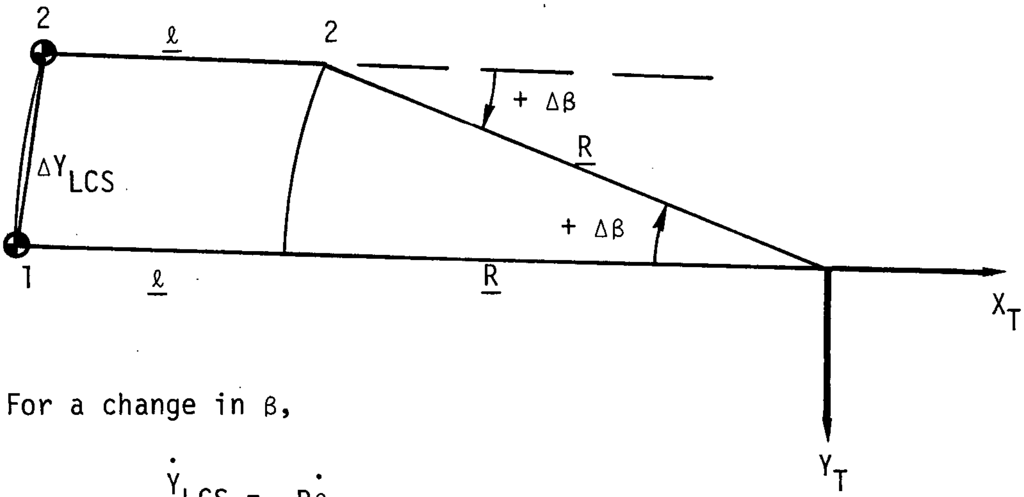
The translational control law is based upon the parabolic switching logic which is the optimal control for minimizing time and fuel for a $1/s^2$ or double integrator plant. Figure 6 illustrates this optimal logic where the available control acceleration is $u = \pm a$.

The factor K in the $f_2(\dot{X})$ and $f_4(\dot{X})$ switching curves is the relative importance of time vs. fuel minimization (i.e., increasing K decreases time and increases fuel and vice versa). As $K \rightarrow \infty$, $f_2(\dot{X}) \rightarrow f_1(\dot{X})$ and $f_4(\dot{X}) \rightarrow f_3(\dot{X})$, which is the time optimal solution (no coast zones). The docking logic will have separate values of K for range control (X) and lateral control (Y, Z) and those will be selected from the allowable docking time constraints.



For a change in ψ_R

$$\dot{Y}_{LCS} = -(R + \ell)\ \dot{\psi}_R$$



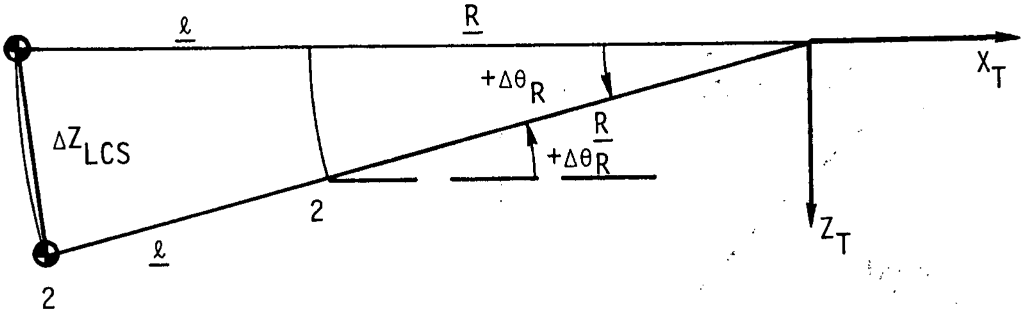
For a change in β,

$$\dot{Y}_{LCS} = -R\dot{\beta}$$

$$\text{Total } \dot{Y}_{LCS} = -R(\dot{\psi} + \dot{\beta}) - \ell\dot{\psi}$$

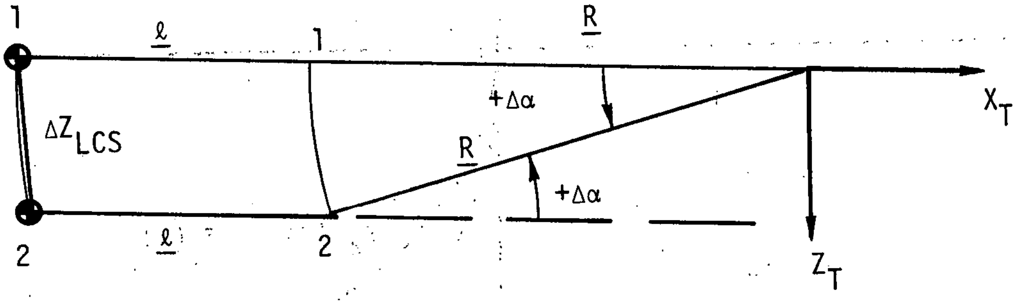
Figure 5a

$\dot{Y}_{LCS}$ Calculation



For change in θ_R,

$$\dot{Z}_{LCS} = (R + \ell)\ \dot{\theta}_R$$



For change in α,

$$\dot{Z}_{LCS} = R\dot{\alpha}$$

$$\text{Total } \dot{Z}_{LCS} = R(\dot{\theta}_R + \dot{\alpha}) + \ell\dot{\theta}_R$$

Figure 5b

$\dot{Z}_{LCS}$ Calculation

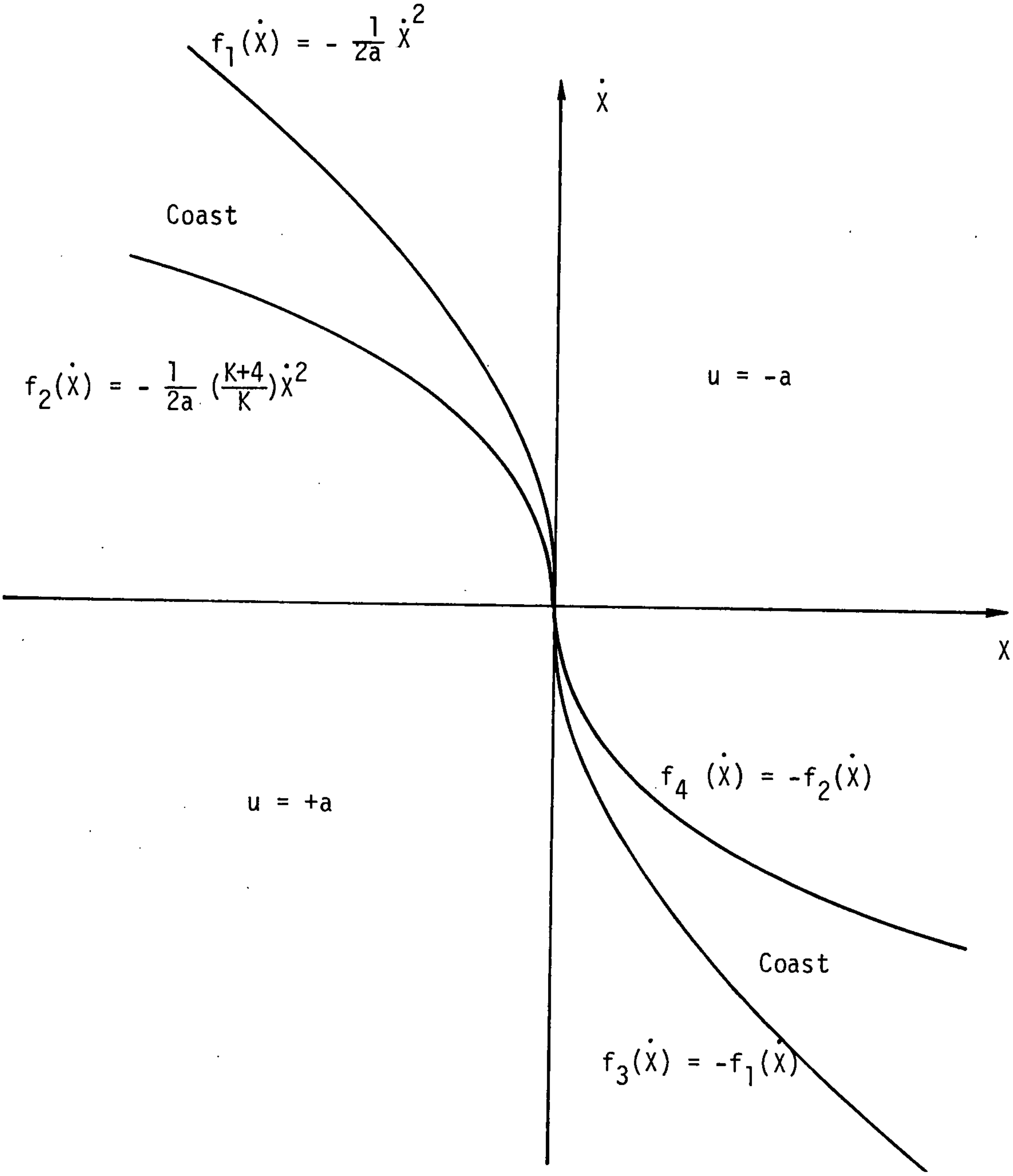

$f_1(\dot{X}) = -\frac{1}{2a}\dot{X}^2$
$\dot{X}$
Coast
$f_2(\dot{X}) = -\frac{1}{2a}\left(\frac{K+4}{K}\right)\dot{X}^2$
$u = -a$
X
$f_4(\dot{X}) = -f_2(\dot{X})$
$u = +a$
Coast
$f_3(\dot{X}) = -f_1(\dot{X})$

Figure 6

Time-Fuel Optimal Control Logic

5.2 Phase I Range Control

The maximum desired docking time is 5 minutes or 300 seconds and the control acceleration is .2 ft/sec^2 (using two of the available four thrusters for finer control). Assuming a worst case initial separation and closing rate of 1500 feet and zero, respectively, the slowest possible path is shown in Figure 7 (A to B to C).

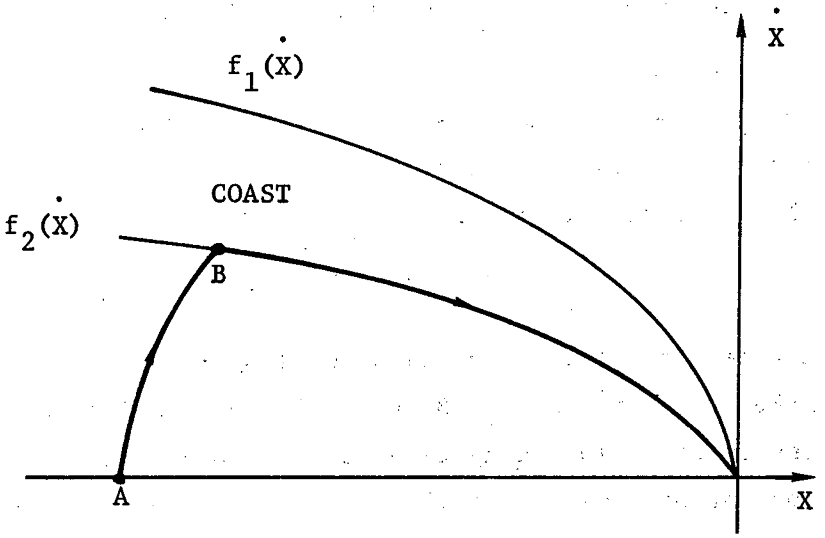


Figure 7 - Maximum Closure Time Trajectory

A to B is the control trajectory and B to C is the slowest trajectory in the coast zone. Hence, the total closure time is

$$T = t_{AB} + t_{BC}$$

The equation for total position change is,

$$X_0 = \frac{1}{2}a\ t^2_{AB} + \frac{1}{2}a\ (\frac{K}{K+4})t^2_{BC}$$

Also, as the rate changes from A to B and B to C must be equal,

$$a\ t_{AB} = a\ (\frac{K}{K+4})\ t_{BC}$$

From these three equations, one can solve for K and $K/(K+4)$

$$K = \frac{8X_0}{aT^2 - 4X_0} \qquad \frac{K}{K+4} = \frac{2X_0}{aT^2 - 2X_0}$$

For the given values of a, T and X_0, $K = 1$ and $K/(K+4) = 0.02$. (1)

Finally, to permit coasting between some position deadband, modifications must be made to the optimal logic shown in Figure 6. Assuming a ±5 foot deadband the complete modified logic is shown in Figure 8.

Figure 9 defines the phase plane switching regions. In Region I, the desired control action is to drive the rate to 0.25 ft/sec (line segment AB). The thruster firing time is determined from $\Delta\dot{X} = at$ or

$$t_{\text{Region I}} = \frac{\dot{X} - .25}{.2} = 5\dot{X} - 1.25 \qquad (2)$$

This firing time, of course, should be no longer than the control sample period to make use of feedback. Because of inaccuracies in modeling it may be necessary to include a hysteresis line bordering Region I to eliminate chattering (see Figure 9). This will be determined at a future date.

In Region II, the desired control action brings the state into the coast zone with an opposite rate sign (example trajectory CD shown in Figure 9). The desired rate change can be found by first writing the equation for the trajectory CD and then simultaneously solving this equation with $f_4(\dot{X})$. The former equation is

$$X - X' = -\frac{1}{2a}\dot{X}^2$$

where $$X' = X_0 + \frac{1}{2}\frac{\dot{X}_0^2}{a}$$

and the latter $$X = \frac{1}{2a}\left(\frac{K+4}{K}\right)\dot{X}^2$$

9.10.2 **Automatic Docking Control Law (cont'd)**

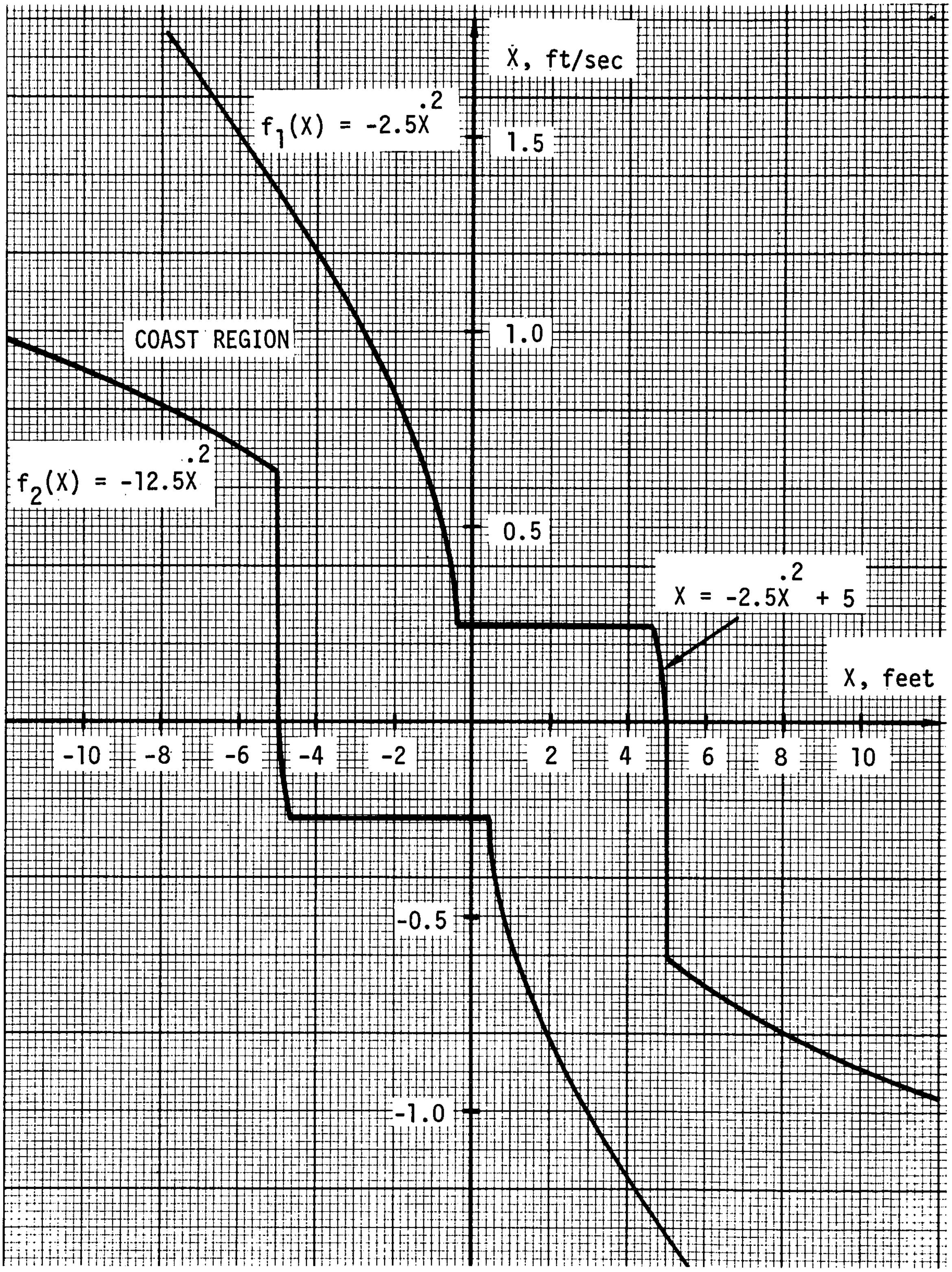


Figure 8

Phase 1 Range Control

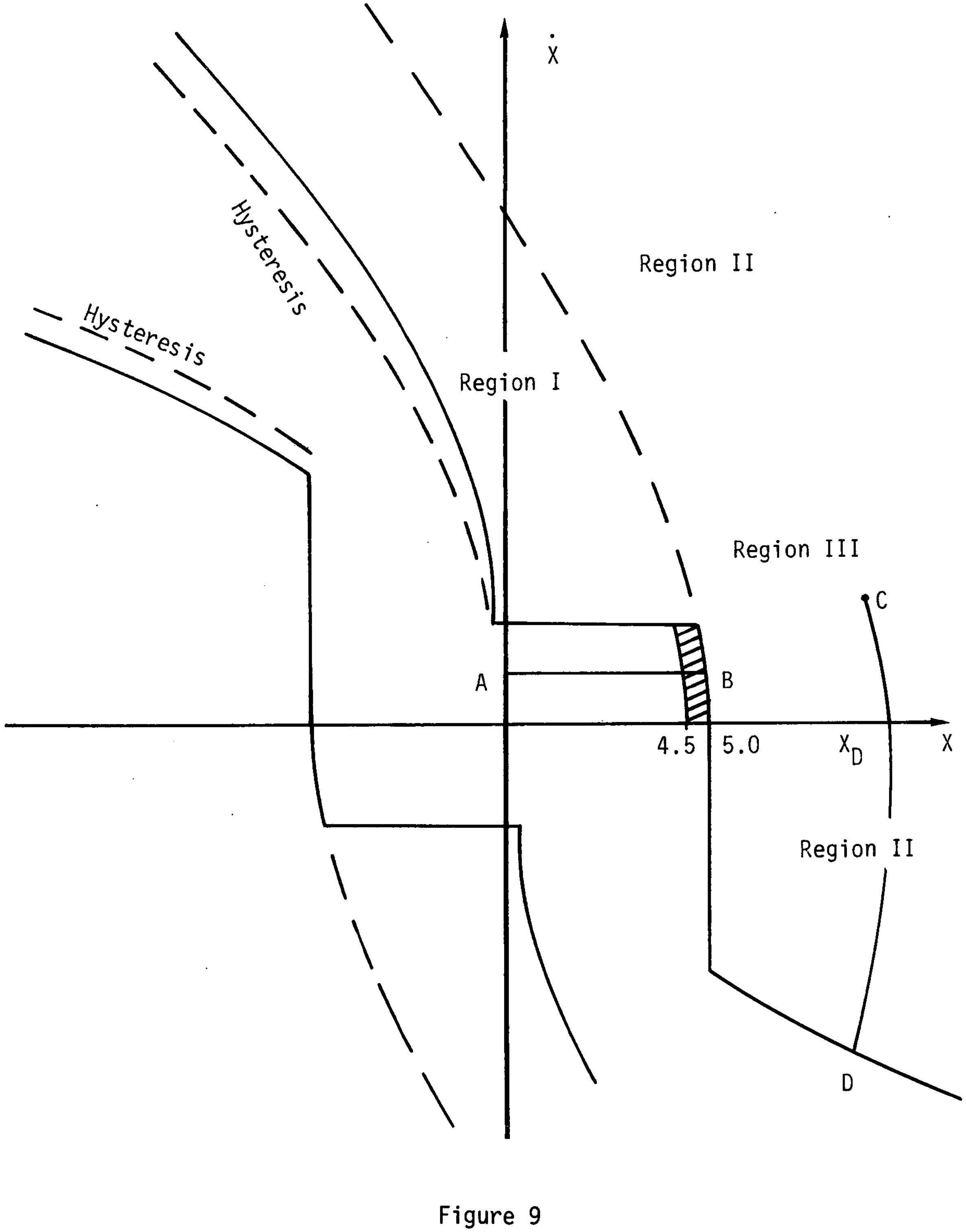


Figure 9

Phase 1 Range Control Regions

The intersection is given by

$$X_{INT} = \left(\frac{K + 4}{K + 2}\right) \frac{X'}{2}$$

$$\dot{X}_{INT} = -\sqrt{\frac{aKX'}{K + 2}}$$

Hence, the firing time is given by

$$t_{Region\ II} = \frac{\dot{X}_0 + \sqrt{\frac{aKX'}{K + 2}}}{a} = \frac{\dot{X}_0}{a} + \sqrt{\frac{K}{K + 2} \frac{X'}{a}} \qquad (3)$$

Finally, Region III is designed to provide smooth limit cycle operation, the control action is to drive the rate to zero. Hence,

$$t_{Region\ III} = \frac{\dot{X}_0}{a} = 5\dot{X}_0 \qquad (4)$$

5.3 Phase 1 Lateral Control

Similarly, K for lateral control, can also be found from the constraints, the acceleration is .2 ft/sec^2, maximum docking time equals 300 seconds and maximum initial position and velocity errors of 150 feet and 3 ft/sec, respectively. Figure 10 shows the slowest trajectory.

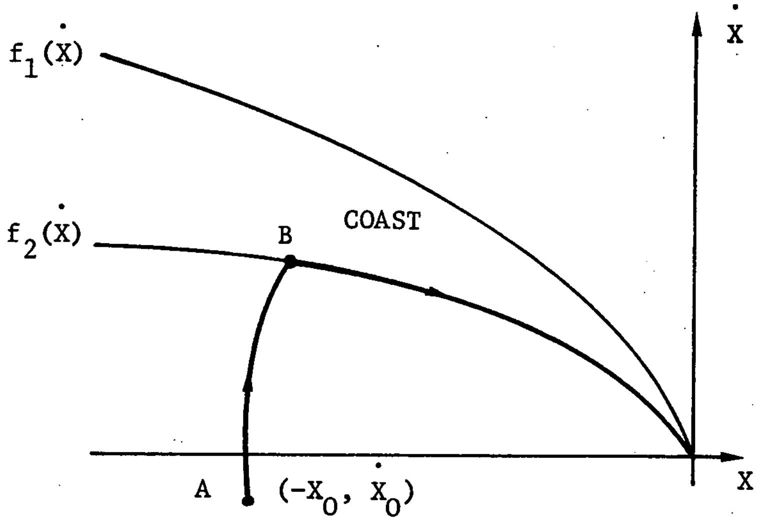


Figure 10 - Maximum Lateral Closure Time

Again, three equations can be found to solve for K,

time, $T = t_{AB} + t_{BC}$

position, $X_0 = -|\dot{X}_0|T + \frac{1}{2}at^2_{AB} + \frac{1}{2}a\left(\frac{K}{K+4}\right)t^2_{BC}$

rate, $a\,t_{AB} = a\left(\frac{K}{K+4}\right)t_{BC} + |\dot{X}_0|$

The equations for K and $\left(\frac{K}{K+4}\right)$ are found to be

$$K = \frac{8a\,(X_0 + |\dot{X}_0|T) - 4|\dot{X}_0|^2}{|\dot{X}_0|^2 + (aT)^2 - 4a\,(X_0 + |\dot{X}_0|T)}$$

$$\frac{K}{K+4} = \frac{2a\,(X_0 + |\dot{X}_0|T) - |\dot{X}_0|^2}{(aT)^2 - 2a\,(X_0 + |\dot{X}_0|T)}$$

K and $\left(\frac{K}{K+4}\right)$ are calculated to be 0.594 and 0.129, respectively. Finally, Figure 11 depicts the lateral Y, Z control logic which also has a ±5 foot deadband modification. The control regions and thruster firing times are of the same format as that shown for range control.

5.4 Phase 2 Control

Phase 2 control begins at the stationkeeping state and ends at contact. For a minimum dispersion docking the following parameters and their derivatives need to be controlled:

a) Range

b) Lateral probe position errors

c) Lateral c.g. position errors

d) Relative roll

e) Relative pitch and yaw

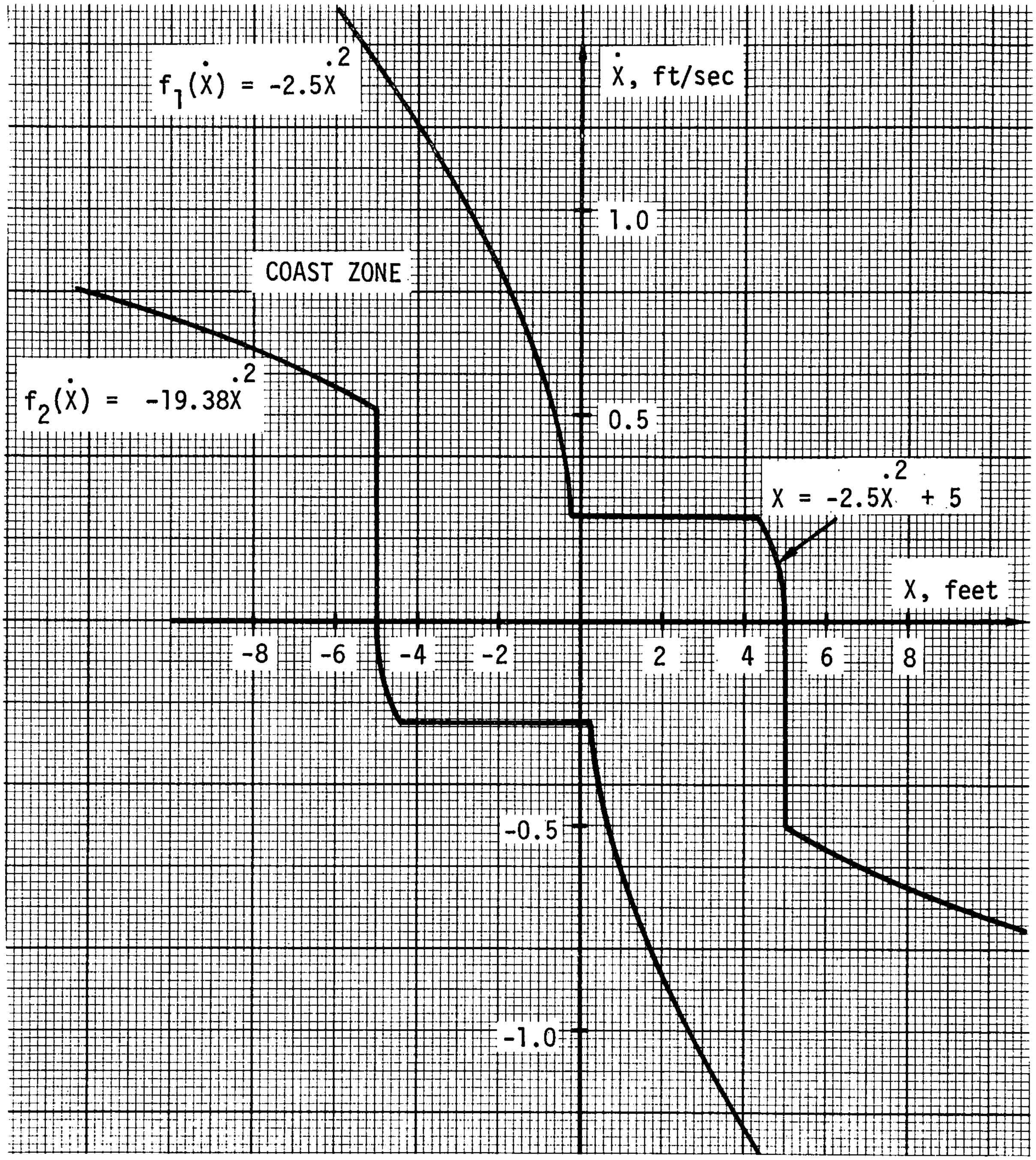


Figure 11

Phase 1 Lateral (Y,Z) Control

5.5 Phase 2 Range Control

In addition to the obvious constraints of maximum time to docking and impact velocity, there may be other constraints; for example, jet plume impingement restrictions. However, until these later constraints are defined, they will be neglected.

The coordinate system used is the TCS. The position error is given by

$$X = R\cos(\theta_R + \alpha)\cos(\Psi_R + \beta) + \ell\cos\theta_R\cos\Psi_R - \ell$$

This quantity will be filtered to provide $\dot{X}$.

The control law will basically be the same as the previously discussed time-fuel optimal logic with the addition of a rate limiting zone for coasting during the final "d" feet of the docking maneuver (see Figure 12). $f_1(X)$ is the curve dictated by two jet braking. K for $f_2(X)$ can be determined by choosing a maximum time for reaching the rate limiting logic for a worst case set of initial conditions. Selecting a maximum time of 3 minutes and a worst case I.C. of $\dot{X}_0 = .25$ ft/sec and $X_0 = 150$ feet, than $K = 0.2734$ and $(\frac{K}{K+4}) = 0.06404$. For this value of K, it can be shown that the maximum closing rate is less than 2 ft/sec.

The upper boundary in the rate limiting zone is set by the maximum impact velocity constraint which is assumed to be 0.1 ft/sec. The lower boundary is a function of maximum allowable time for coast and the distance for coast, d. Assuming a 100 seconds and 5 feet, respectively, the lower limit is 0.05 ft/sec.

There are three control regions. The first is the one lying to the right of the parabolic coast zone and above the rate limiting coast zone. Here the control should aim for a rate of 0.075 ft/sec (mid-way in rate limiting zone).

$$t_{\text{Region I}} = \frac{\dot{X}_0 - .075}{a} \qquad (5)$$

9.10.2 Automatic Docking Control Law (cont'd)

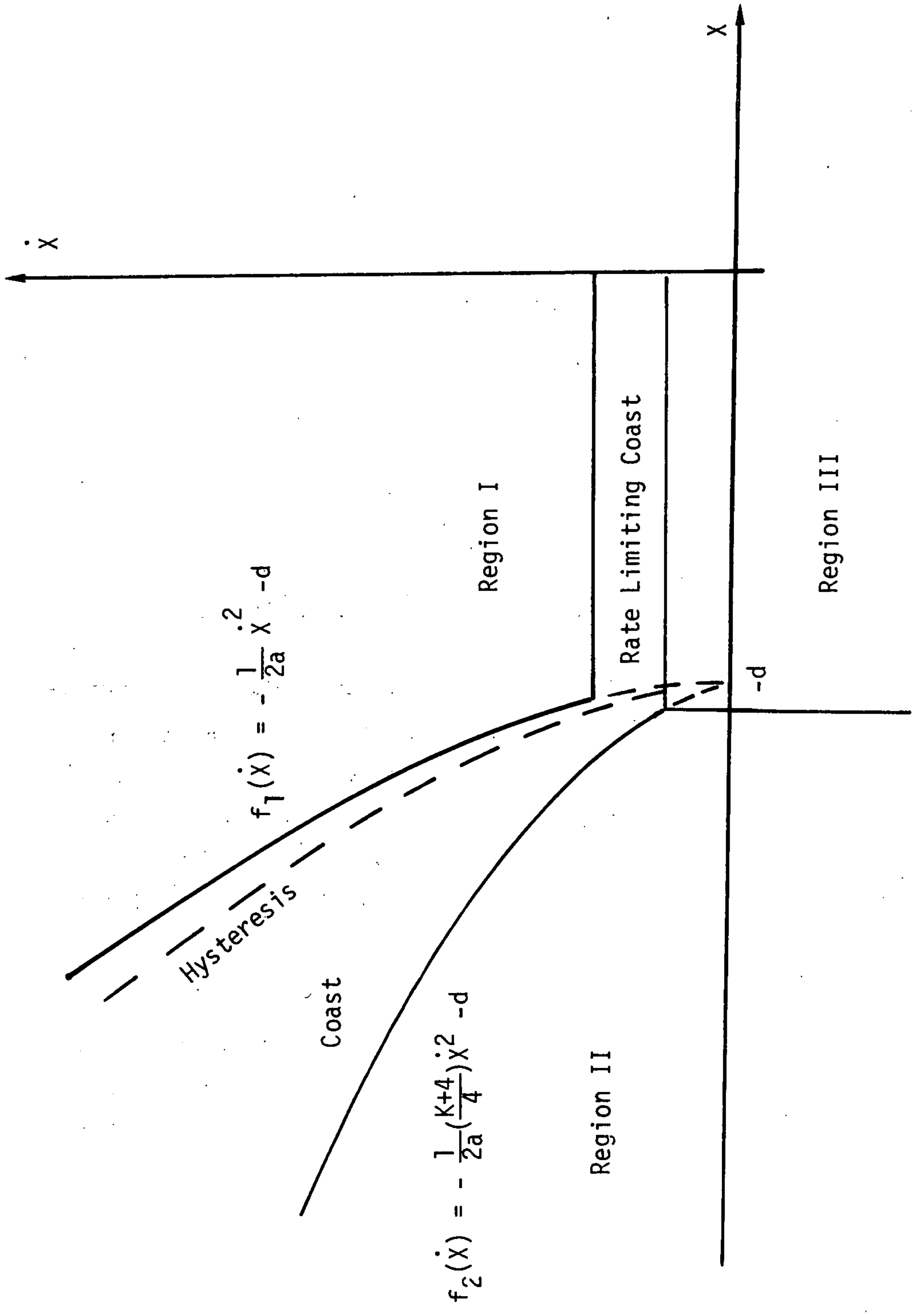

X
$\dot{X}$
Region I
Rate Limiting Coast
Region III
-d
$f_1(\dot{X}) = -\frac{1}{2a}\dot{X}^2 - d$
Hysteresis
Coast
$f_2(\dot{X}) = -\frac{1}{2a}(\frac{K+4}{4})\dot{X}^2 - d$
Region II

Figure 12
Phase 2 Range Control Phase Plane

Region II lies to the left of the parabolic coast zone and the line X = -(d + .1). Control requirement for this region is to bring the state up to the lower parabolic switch line. Firing has been derived previously in a similar calculation.

$$t_{\text{Region II}} = \left\| \frac{X_0}{a} + \sqrt{\left(\frac{K}{K+2}\right) \left| \left| \frac{X'}{a} \right| \right|} \right\| \qquad (6)$$

where $X' = X_0 + \frac{1}{2} \frac{X_0^{\;2}}{a}$

Finally, Region III lies to the right of the line X = -(d + .1) and below the rate limiting coast zone. The firing time is

$$t_{\text{Region III}} = \frac{0.075 - \dot{X}}{a} \qquad (7)$$

5.6 Control of Lateral Probe and C. G. Position Errors and Relative Pitch and Yaw Angles

Lateral probe position error should be controlled directly because the allowable lateral probe displacement at impact is likely to be quite small (one foot or less). Indirect control, by simply nulling c.g. position and relative pitch and yaw attitudes, can cause significant lateral dispersions (see reference). However, as lateral probe position error is a function of lateral c.g. position errors and the relative pitch and yaw angles, the controls for all three must be coordinated.

Considering the X-Z plane-first there are three pairs of variables to be controlled (Z_{cg}, $\dot{Z}_{cg}$) (Z_p, $\dot{Z}_p$) and (θ_R, $\dot{\theta}_R$). During this phase of control the two translational parameters will be calculated as follows in the TCS (see Figure 13).

$$Z_{cg} = R \sin(\theta_R + \alpha) + \ell \sin \theta_R$$

$$Z_p = R \sin(\theta_R + \alpha)$$

$$Z_{cg} = R \sin (\theta_R + \alpha) + \ell \sin \theta_R$$

$$Z_p = R \sin (\alpha + \theta_R)$$

Note: Negative θ_R shown (wrt TCS)

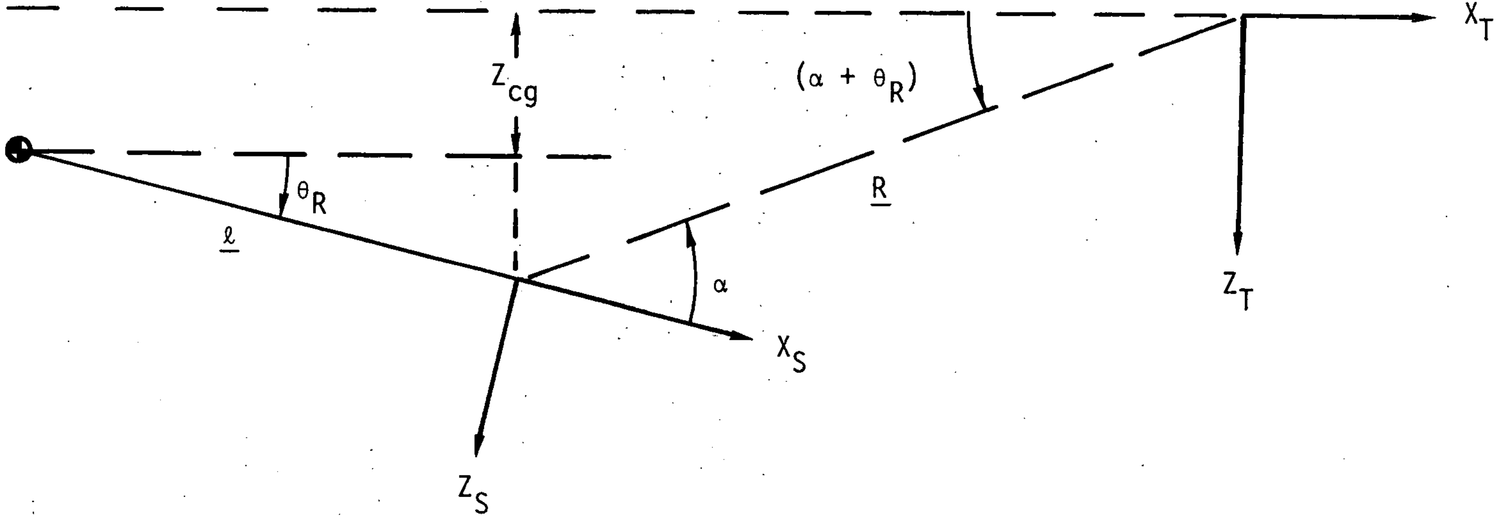


Figure 13

Illustration of Z_{cg} and Z_p

The respective rates will be obtained by a filtering routine to be determined.

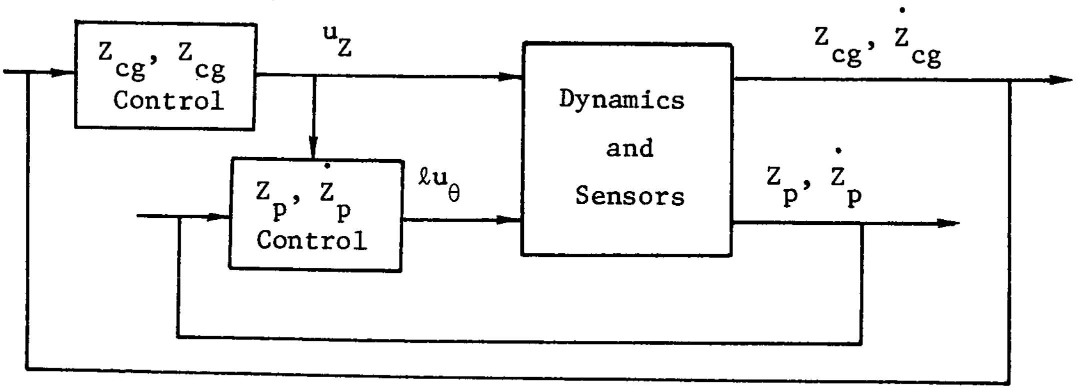
The control inputs are, of course, the ±Z thruster forces, u_Z, and the ±θ pitch torques, u_θ, which are applied in the following manner:

$$\ddot{Z}_p = u_Z - \ell u_\theta$$

$$\ddot{Z}_{cg} = u_Z$$

$$\ddot{\theta}_R = u_\theta$$

Intuitively, it can be seen that explicit control of any two of the variables (Z_p, Z_{cg}, θ_R), implicitly controls the third. For example, a control which forces two of the variables into prescribed limit cycles, indirectly bounds the remaining variable into some limit cycle. As Z_p has already been chosen as one of the variables to be controlled directly, it only remains to select either Z_{cg} or θ_R for the other directly controlled variable. Either is acceptable; however, Z_{cg} is chosen because u_θ has five times more control authority than u_Z and by this choice ℓu_θ can be used exclusively for Z_p control. Summarizing, at this point we have u_Z for exclusive control of Z_{cg} and ℓu_θ for exclusive control of Z_p where u_Z is a known disturbance of Z_p. In block diagram form, this is represented as:



As an aid in defining the control, it is helpful to visualize the desired control state. Realizing that there is a minimum control impulse on Z_{cg} and θ_R which necessitates deadbands, Figure 14 indicates the ideal control state. Effectively, we have Z_p located on the approach path with Z_{cg} moving up and down in a deadband. Requirements for this condition are

$$(1) \quad \dot{Z}_{cg} - \ell\dot{\theta}_R = 0$$

$$(2) \quad \dot{Z}_p = 0$$

A method of approximating this control state is to (1) drive $(Z_{cg}, \dot{Z}_{cg})$ into a deadbanded limit cycle (consistent with allowable θ_R range), (2) drive (Z_p, Z_p) into a very small deadband limit cycle, and (3) use differential jet firings to approach $Z_p = 0$ (i.e., take advantage of the small control impulse available from $u_Z - \ell u_\theta$).

A final consideration is that Z_{cg} and Z_p should be within their deadbands before the range control has reached the rate limiting zone. As the maximum closing rate is 2 ft/sec, the minimum time for this is 150/2 or 75 seconds.

Switching logic for $(Z_{cg}, \dot{Z}_{cg})$

The switching logic will have the same form as that used for Phase 1 except there will be a different deadband and value of K for the $f_2(X)$ function. The deadband is dependent on the per axis allowable misalignment angle, γ.

Hence, we have

$$Z_{cg} \text{ DEADBAND} = \ell \sin\gamma$$

Assuming a γ of 2° the Z_{cg} deadband is 2.5 feet. To determine K, we insert T = 75 seconds and worst case initial conditions into the previously derived formula

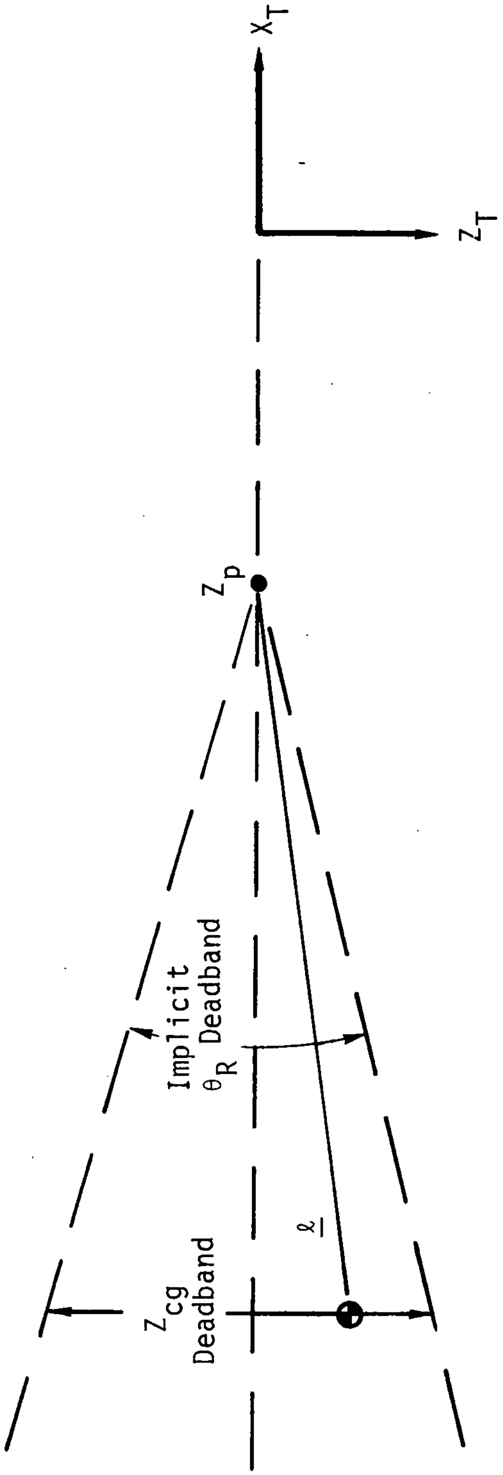


Figure 14
Desired Control State

$$K = \frac{8a\ (X_0 + |\dot{X}_0|T) - 4\ |\dot{X}_0|^2}{|\dot{X}_0|^2 + a^2T^2 - 4a(X_0 + |\dot{X}_0|T)}$$

Assuming X_0 = 10 ft and X_0 = -.25 ft/sec, K = .2265. However, as the value of K used for Phase 1 control, 0.594, is more conservative (i.e., longer hence quicker) it will be used for simplicity of logic coding. A final modification from the Phase 1 logic is necessitated by the differential jet firing technique which will be used to null Z_p. It requires the minimum delta V impulses available from the Z-translation and pitch jets to be the same. As u_Z is approximately five times smaller than u_θ, the minimum impulse from u_Z must be increased proportionately. The control regions are the same as that previously used for lateral control.

5.7 Switching logic for $(Z_p, \dot{Z}_p)$

The control for (Z_p, Z_p) will also be a modified form of the time - fuel optimal switching logic. Figure 15 illustrates this logic. The linear acceleration from the pitch thrusters is five times greater than that from the translational thrusters, hence, for two thruster acceleration

$$f_1(\dot{Z}_p) = \frac{1}{2a}\ (\dot{Z}_p)^2 = \frac{1}{2}\dot{Z}_p^{\ 2}$$

To determine K for the $f_2(Z_p)$ function we must again revert back to the worst case initial conditions and maximum allowable time (this was chosen in the range control law to be 3 minutes). Worst case initial conditions from the stationkeeping phase are shown in Figure 16. The position error is seen to be about -11 feet whereas velocity error is about -1.2 ft/sec (due to minimum impulse rates from translational and rotational control). Using this I.C., K is found to be 0.0575 and K/(K + 4) equals 0.0142. These small values may be increased slightly because of the high sensitivity of the $f_2(Z_p)$ function to a rate error in Z_p. The desired deadband as shown

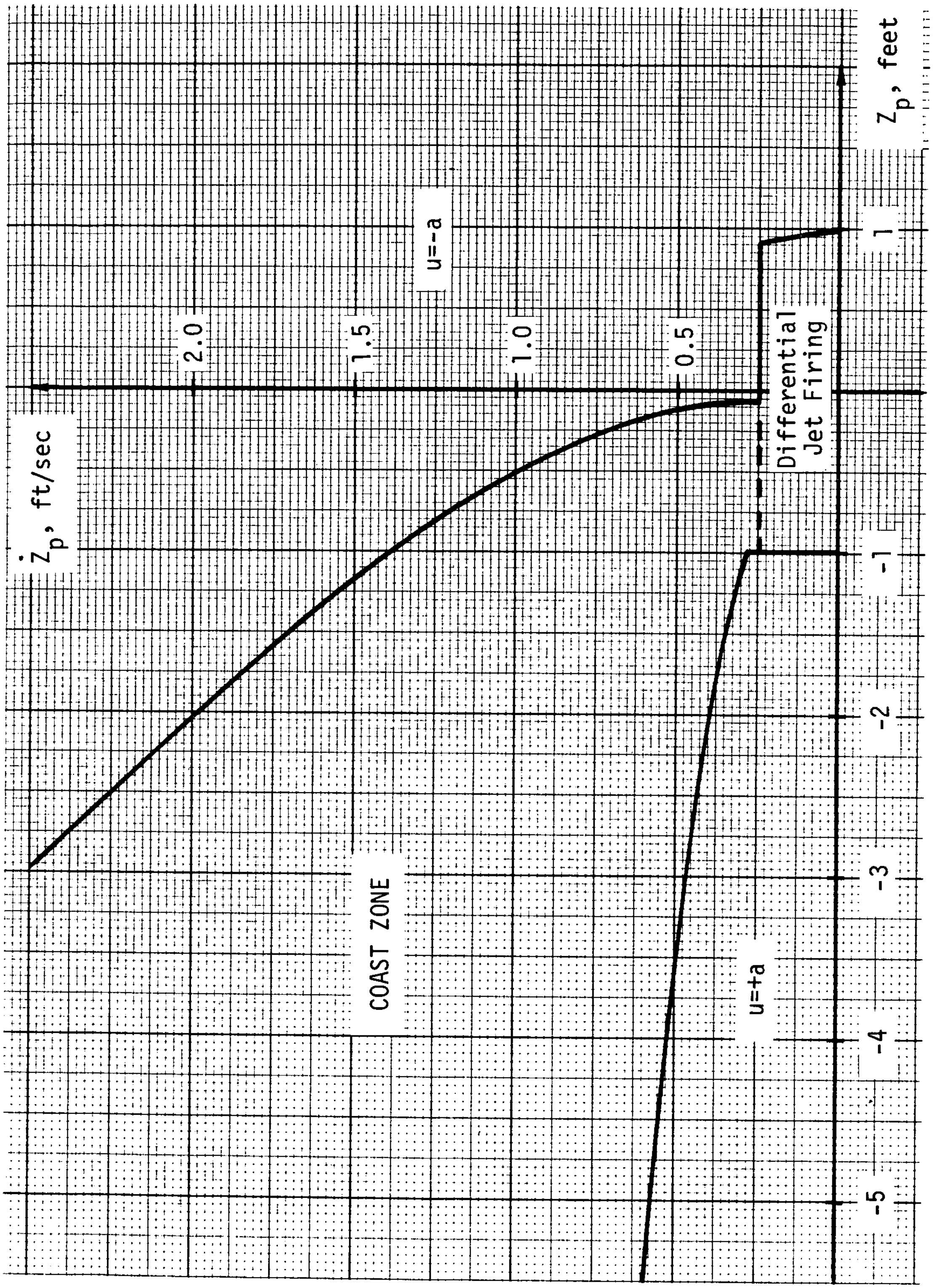


Figure 15

Z_p, $\dot{Z}_p$ Phase Plane

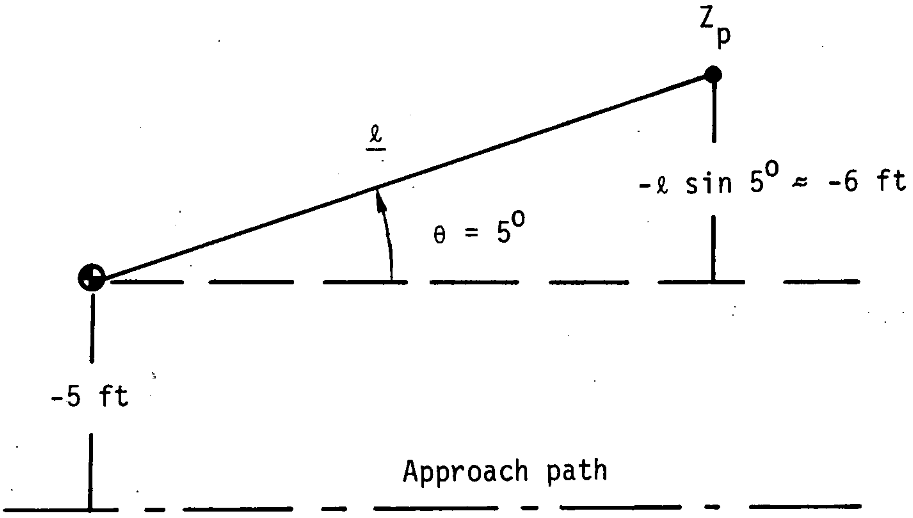


Figure 16

Worst Case Position I.C.

in Figure 15 is one foot. To account for u_Z thruster firings the phase plane is broken into several control regions, as illustrated in Figure 17. With no Z-thruster control disturbances, u_Z, the firing times for Regions I, II, and III are calculated in the same manner as the other phase planes. Namely,

$$t_{\text{Region I}} = \frac{\dot{X} - .125}{a} = \dot{X} - .125 \qquad (8)$$

$$t_{\text{Region II}} = \frac{X_0}{a} + \sqrt{\left(\frac{K}{K+2}\right) \frac{X'}{a}} \qquad (9)$$

$$t_{\text{Region III}} \equiv \frac{\dot{X}_0}{a} = \dot{X}_0 \qquad (10)$$

However, in Regions I, II, and III, if a non-zero command is scheduled from the $(Z_{cg}, \dot{Z}_{cg})$ phase plane, and this would cause the $(Z_p, \dot{Z}_p)$ state to diverge (because of disagreement in sign), then this command is treated as a disturbance and the $(Z_p, \dot{Z}_p)$ firing time is increased proportionately for opposing commands

$$t_{\text{Region I, II}} = t_{\text{Region I, II}} + \frac{1}{5} t_{(Z, \dot{Z})} \qquad (11)$$

However, as the net acceleration during this disturbance period is reduced by 20 percent,convergence time in the phase plane may be increased. If this proves a significant factor,K will be increased. Commands in the proper direction are not compensated for as this would cause chattering during long Z firing times. In Region III a u_Z command in either direction should be compensated for

$$t_{\text{Region III}} = t_{\text{Region III}} \pm \frac{1}{5} t_{(Z,\dot{Z})} \qquad (12)$$

as the desired control action is to drive the rate to zero.

Finally, in Region IV, no control action is taken unless there is a u_Z command; then the desired control action is to drive the rate to

9.10.2 <u>Automatic Docking Control Law (cont'd)</u>

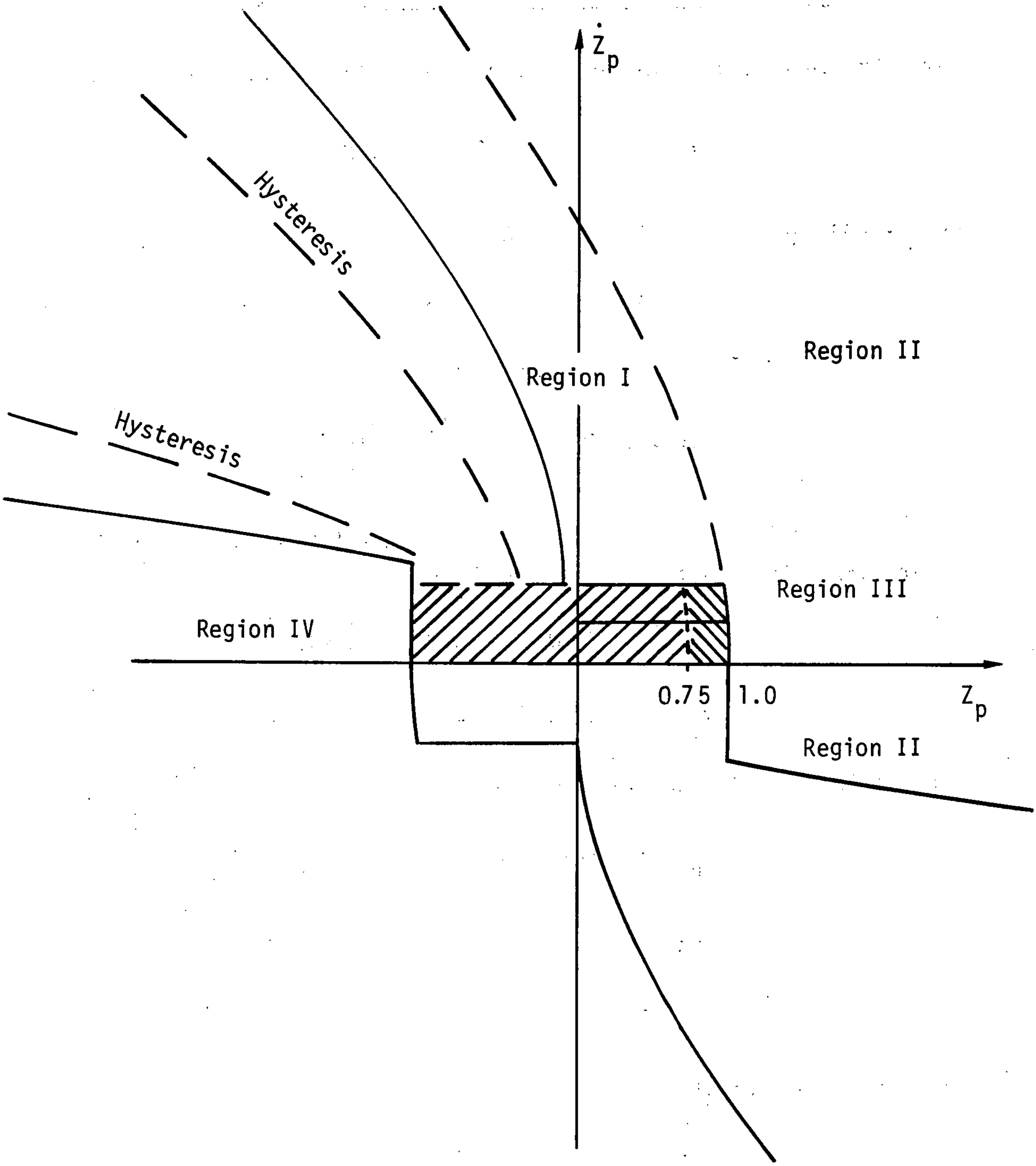


Figure 17

Z_p, $\dot{Z}_p$ Control Regions

zero through differential jet firings. The firing time equation is

$$\dot{Z}_p + u_Z \, t_{(Z,\dot{Z})} - \ell u_\theta \, t_{(Z_p,\, \dot{Z}_p)} = 0$$

Hence, the sign of the control is given by

$$\text{sign}\ (\ell u_\theta) = \text{sign}\ (\dot{Z}_p + u_Z T_{(Z,\, \dot{Z})}) \qquad (13)$$

The firing time by

$$t_{\text{Region IV}} = \frac{\dot{Z}_p + u_Z \, t_{(Z,\, Z)}}{\ell u_\theta} \qquad (14)$$

A dual relation exists for control in the X-Y plane (see Figure 18). The position errors are

$$Y_{cg} = -R \sin\ (\Psi_R + \beta) - \ell \sin \Psi_R$$

$$Y_p = -R \sin\ (\Psi_R + \beta)$$

Applicable control accelerations are

$$\ddot{Y}_p = u_Y + \ell u_\Psi$$

$$\ddot{Y}_{cg} = u_Y$$

$$\ddot{\Psi}_R = u_\Psi$$

The desired end condition requires

$$(1) \quad Y_{cg} + \ell \dot{\Psi}_R = 0$$

$$(2) \quad Y_p, \dot{Y}_p = 0$$

$$Y_{cg} = -R \sin (\Psi_R + \beta) - \ell \sin \Psi_R$$

$$Y_p = -R \sin (\Psi_R + \beta)$$

Note: Negative β shown

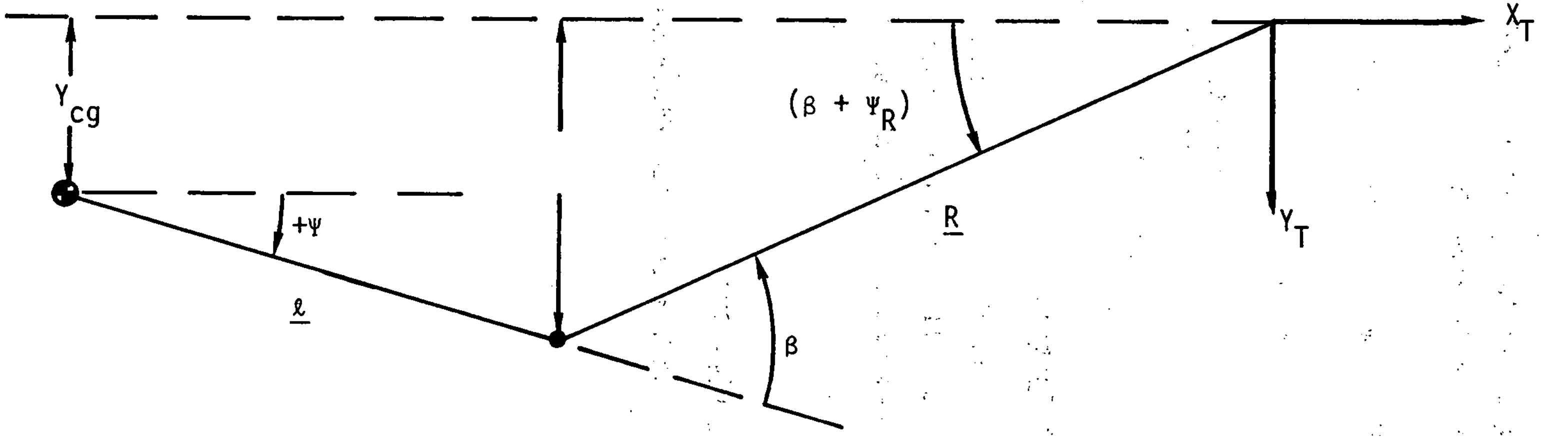


Figure 18

Illustration of Y_{cg} and Y_p

The switching logic is identical except for Region IV where the sign of the control effort is given by

$$\text{Sign } (\ell u_{\Psi}) = -\text{sign } (\dot{Y}_p + u_Y \, t_{(Y,\ \dot{Y})}) \qquad (15)$$

and the firing time is given by

$$t_{\text{Region IV}} = \frac{\dot{Y}_p + u_Y \, t_{(Y,\ \dot{Y})}}{-\ell u_{\Psi}} \qquad (16)$$

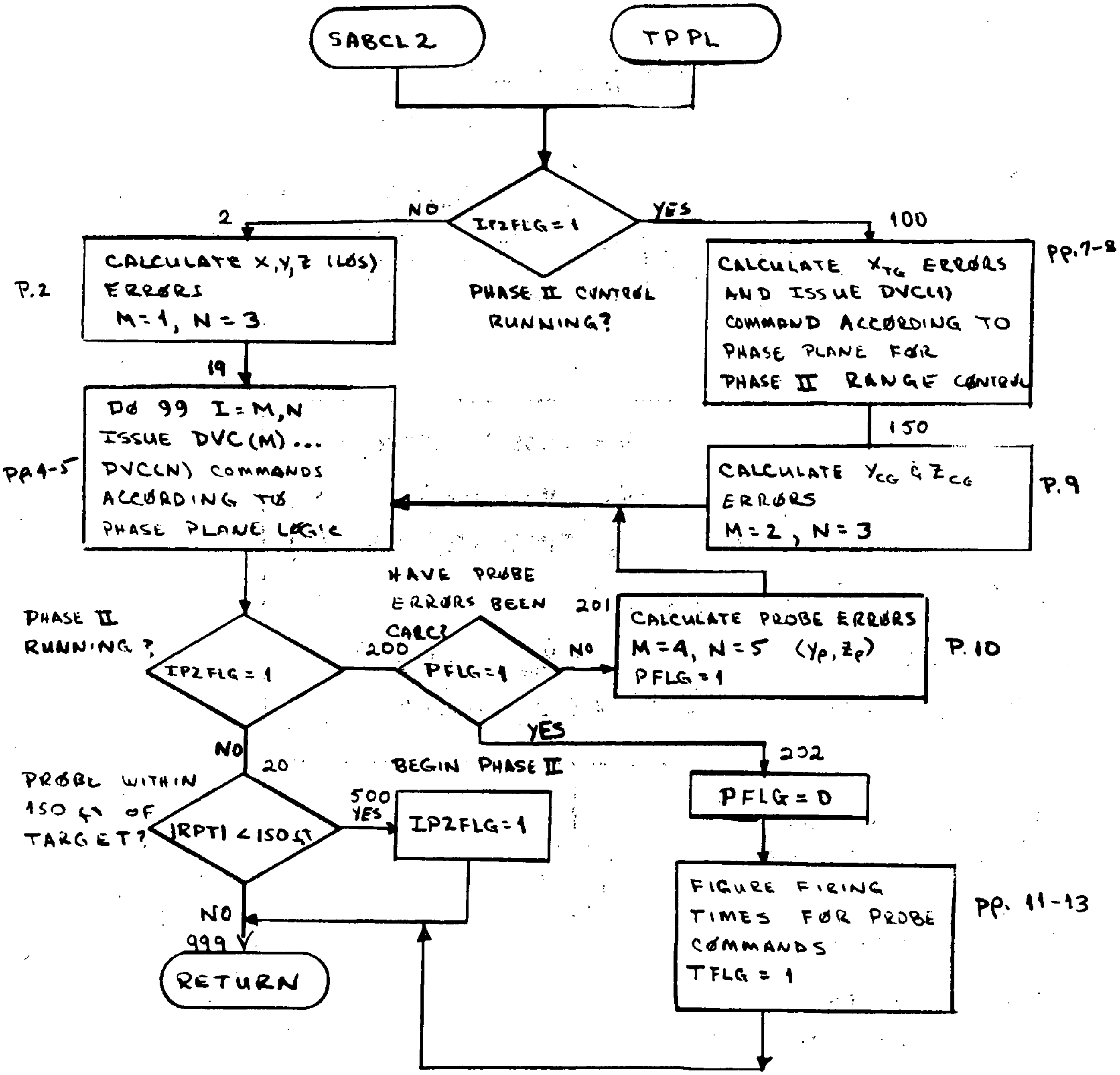
5.8 Relative Roll Control

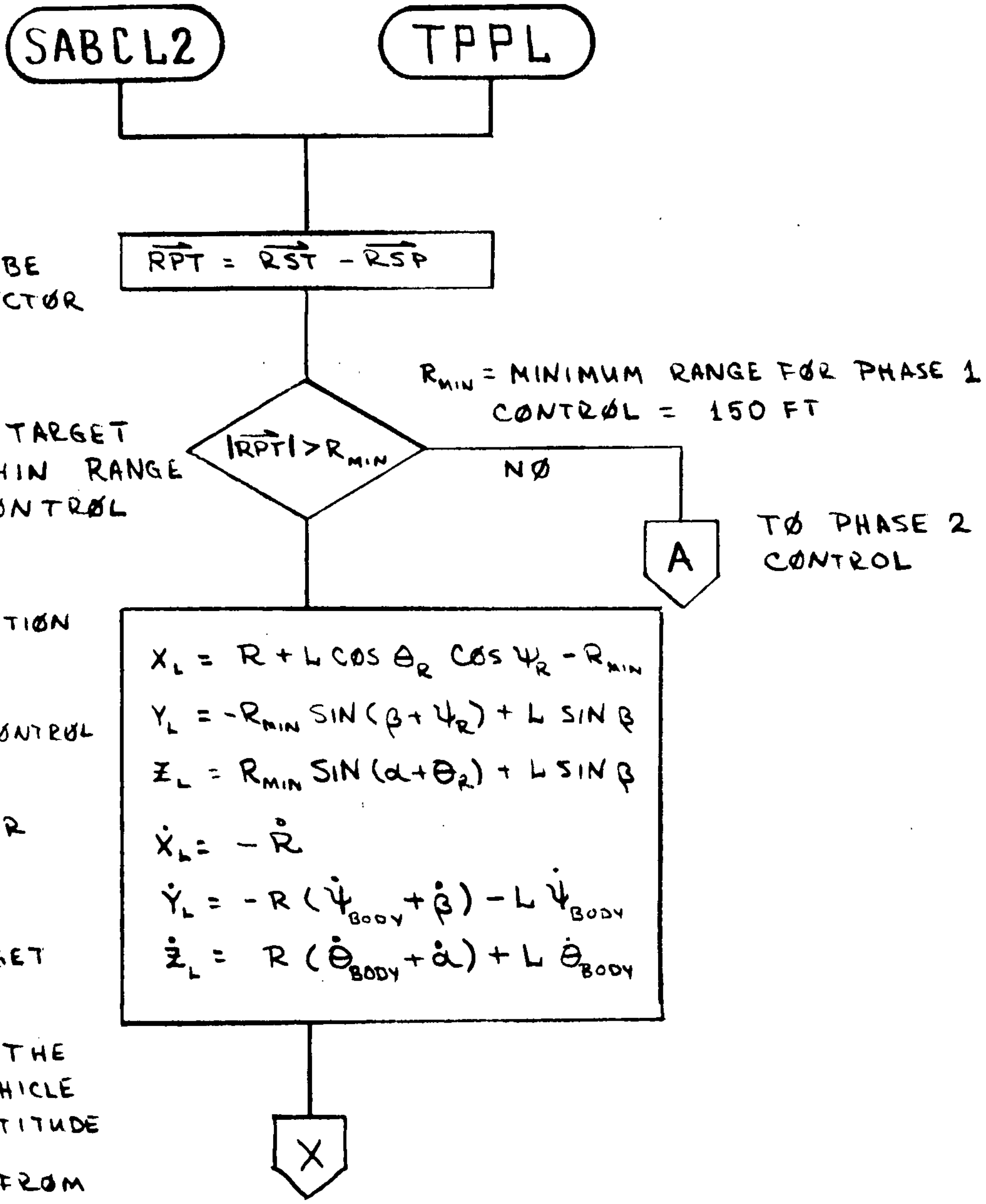
Relative roll control will be the same as that used in Phase 1 except that the deadband will be reduced to comply with docking constraints and close-in measurement accuracies. Two degrees will be assumed initially.

6. Detailed Flow Diagrams

This section contains the flow diagrams for the Automatic Docking Control Law.

THE FØLLØWING FLØW CHART SHØWS THE FUNCTIØNAL FLØW OF PHASE I AND PHASE II CØNTRØL.


SABCL2
TPPL
IP2FLG = 1
NO
YES
PHASE II CØNTRØL RUNNING?
2
CALCULATE X,Y,Z (LØS) ERRØRS M=1, N=3
P.2
100
CALCULATE X_{TG} ERRØRS AND ISSUE DVC(1) COMMAND ACCØRDING TO PHASE PLANE FØR PHASE II RANGE CØNTRØL
PP. 7-8
19
DØ 99 I=M,N ISSUE DVC(M)... DVC(N) COMMANDS ACCØRDING TØ PHASE PLANE LØGIC
PP 4-5
150
CALCULATE Y_{CG} & Z_{CG} ERRØRS M=2, N=3
P.9
PHASE II RUNNING?
IPZFLG = 1
HAVE PRØBE ERRØRS BEEN CALC?
200
PFLG=1
NO
201
CALCULATE PRØBE ERRØRS M=4, N=5 (Y_p, Z_p) PFLG=1
P.10
YES
202
PFLG = 0
NO
20
PRØBE WITHIN 150 ft OF TARGET?
|RPT| < 150 ft
500 YES
BEGIN PHASE II
IP2FLG=1
FIGURE FIRING TIMES FØR PRØBE CØMMANDS TFLG = 1
PP. 11-13
NO
999
RETURN


SABCL2
TPPL
$\overrightarrow{RPT} = \overrightarrow{RST} - \overrightarrow{RSP}$
R_{MIN} = MINIMUM RANGE FØR PHASE 1 CØNTRØL = 150 FT
$|\overrightarrow{RPT}| > R_{MIN}$
NØ
A
TØ PHASE 2 CØNTROL
$X_L = R + L \cos \theta_R \cos \psi_R - R_{MIN}$
$Y_L = -R_{MIN} \sin(\beta + \psi_R) + L \sin \beta$
$Z_L = R_{MIN} \sin(\alpha + \theta_R) + L \sin \beta$
$\dot{X}_L = -\dot{R}$
$\dot{Y}_L = -R(\dot{\psi}_{BODY} + \dot{\beta}) - L \dot{\psi}_{BODY}$
$\dot{Z}_L = R(\dot{\theta}_{BODY} + \dot{\alpha}) + L \dot{\theta}_{BODY}$
X

CALCULATE PRØBE TØ TARGET VECTØR

IS PRØBE TØ TARGET DISTANCE WITHIN RANGE ØF PHASE 1 CØNTRØL

CALCULATE PØSITIØN AND VELØCITY ERRØRS FØR PHASE PLANE CØNTRØL

α, β ARE SENSØR PITCH AND YAW LØS ANGLES

R = CG TØ TARGET RANGE

$(\phi_R, \theta_R, \psi_R)$ IS THE ESTIMATED VEHICLE TØ TARGET ATTITUDE

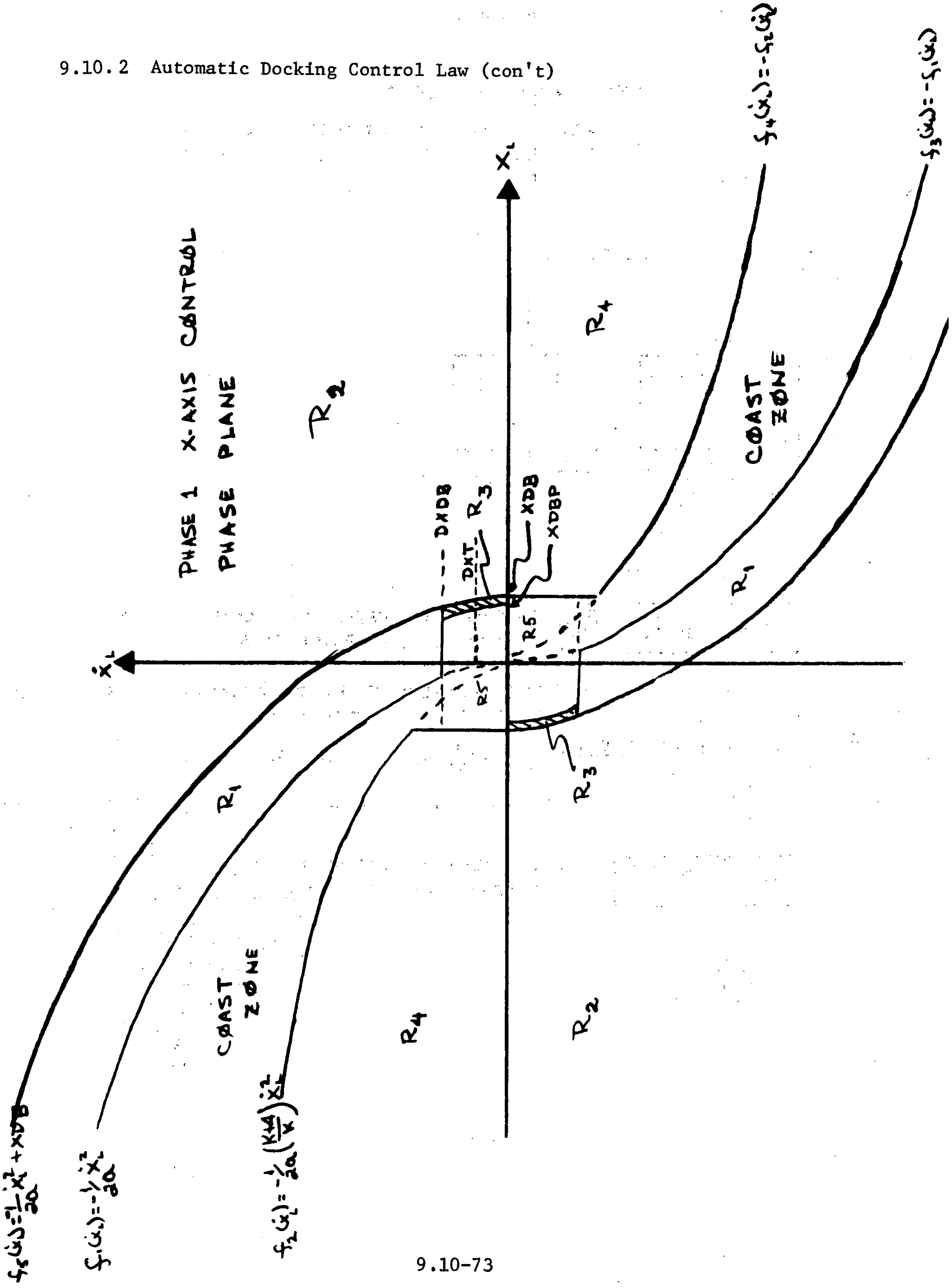
L = DISTANCE FRØM CG TØ SENSØR

$(\dot{\phi}_{BODY}, \dot{\theta}_{BODY}, \dot{\psi}_{BODY})$ IS THE VEHICLE BØDY RATE

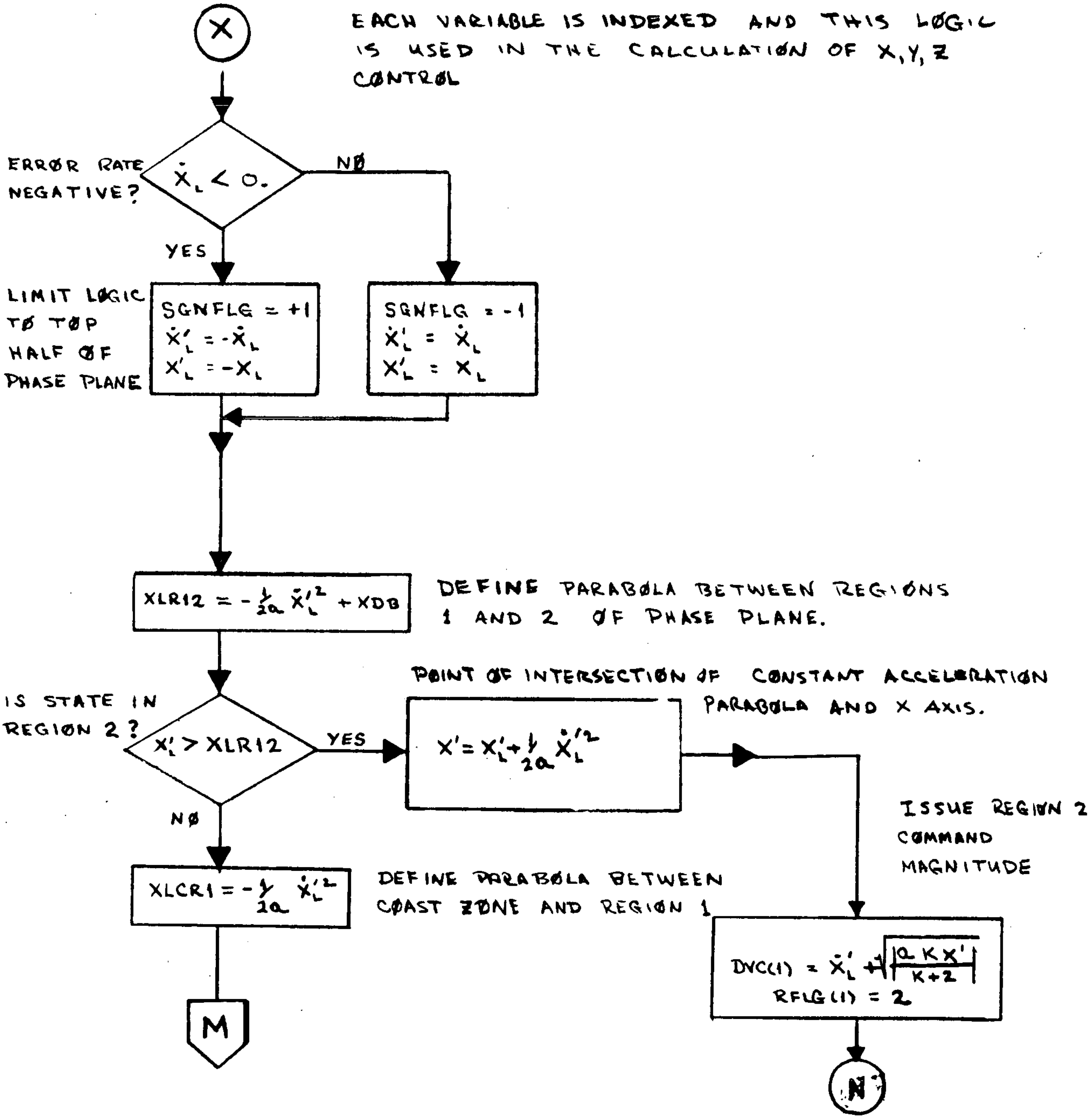
$\dot{\alpha}, \dot{\beta}$ = LØS RATES

$\dot{R}$ = CG TØ TARGET CLØSING RATE

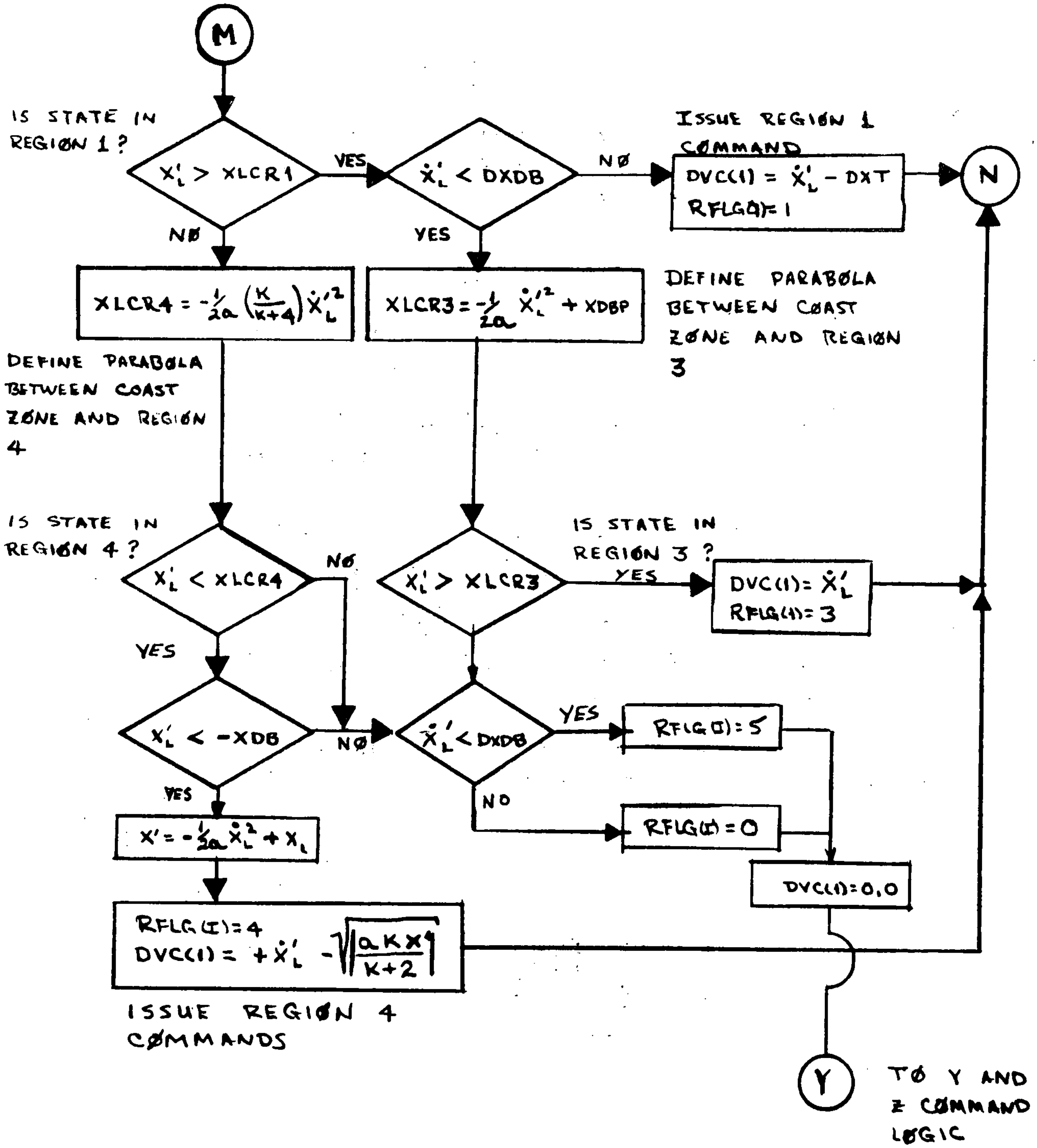
C.10

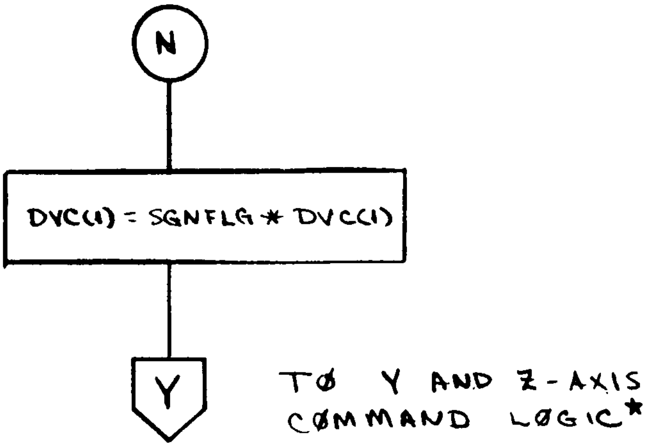

PHASE 1 X-AXIS CONTROL
PHASE PLANE
x_L
$\dot{x}_L$
R_2
R_4
R_1
R_3
R5
COAST ZONE
DXDB
DXT
XDB
XDBP
$f_4(\dot{x}_L) = -f_2(\dot{x}_L)$
$f_3(\dot{x}_L) = -f_1(\dot{x}_L)$
$f_5(\dot{x}_L) = \frac{-1}{2a}\dot{x}_L^2 + XDB$
$f_1(\dot{x}_L) = -\frac{1}{2a}\dot{x}_L^2$
$f_2(\dot{x}_L) = -\frac{1}{2a}\left(\frac{K+1}{K}\right)\dot{x}_L^2$

X-AXIS COMMAND LOGIC


EACH VARIABLE IS INDEXED AND THIS LOGIC IS USED IN THE CALCULATION OF X, Y, Z CONTROL
X
ERROR RATE NEGATIVE?
$\dot{X}_L < 0.$
NO
YES
LIMIT LOGIC TO TOP HALF OF PHASE PLANE
SGNFLG = +1
$\dot{X}'_L = -\dot{X}_L$
$X'_L = -X_L$
SGNFLG = -1
$\dot{X}'_L = \dot{X}_L$
$X'_L = X_L$
$XLR12 = -\frac{1}{2a}\dot{X}_L'^2 + XDB$
DEFINE PARABOLA BETWEEN REGIONS 1 AND 2 OF PHASE PLANE.
IS STATE IN REGION 2?
$X'_L > XLR12$
YES
POINT OF INTERSECTION OF CONSTANT ACCELERATION PARABOLA AND X AXIS.
$X' = X'_L + \frac{1}{2a}\dot{X}_L'^2$
ISSUE REGION 2 COMMAND MAGNITUDE
NO
$XLCR1 = -\frac{1}{2a}\dot{X}_L'^2$
DEFINE PARABOLA BETWEEN COAST ZONE AND REGION 1
$DVC(I) = \dot{X}'_L + \sqrt{\frac{a K X'}{K+2}}$
RFLG(I) = 2
M
N

9.10.2 Automatic Docking Control Law (con't)





<u>CØNSTANTS FØR X-AXIS CØMMAND LØGIC</u>

$a = 0.2$ FT/SEC^2 = CØNSTANT ACCELERATIØN AVAILABLE ALØNG X-AXIS

K = 1.0 = ØPTIMIZATIØN FACTØR FØR TIME VS. FUEL

XDB = 5.0 FT. (SEE PHASE PLANE IN THE FIGURE)

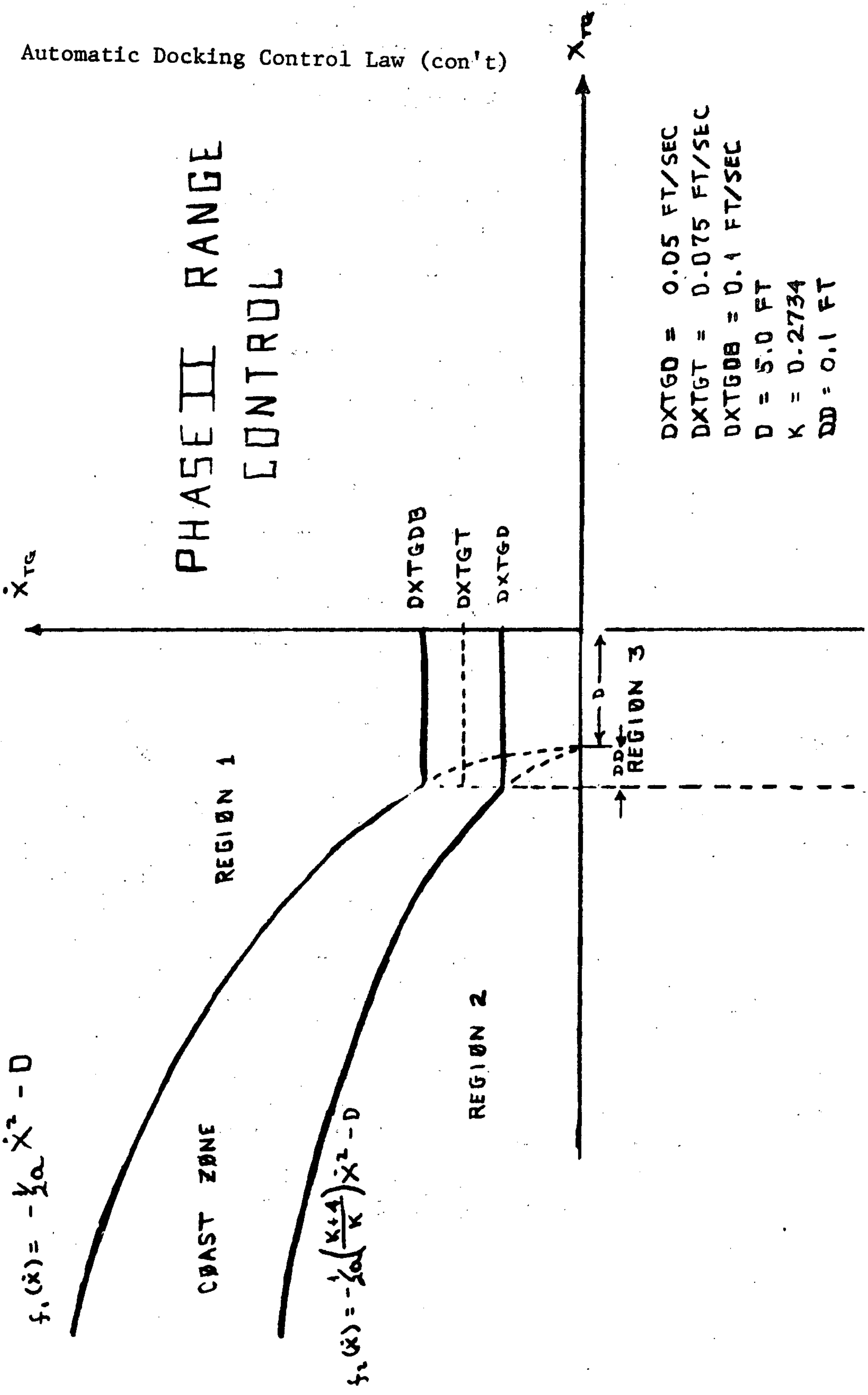
DXDB = 0.25 FT/SEC (SEE PHASE PLANE)

XDBP = 4.5 FT (SEE PHASE PLANE)

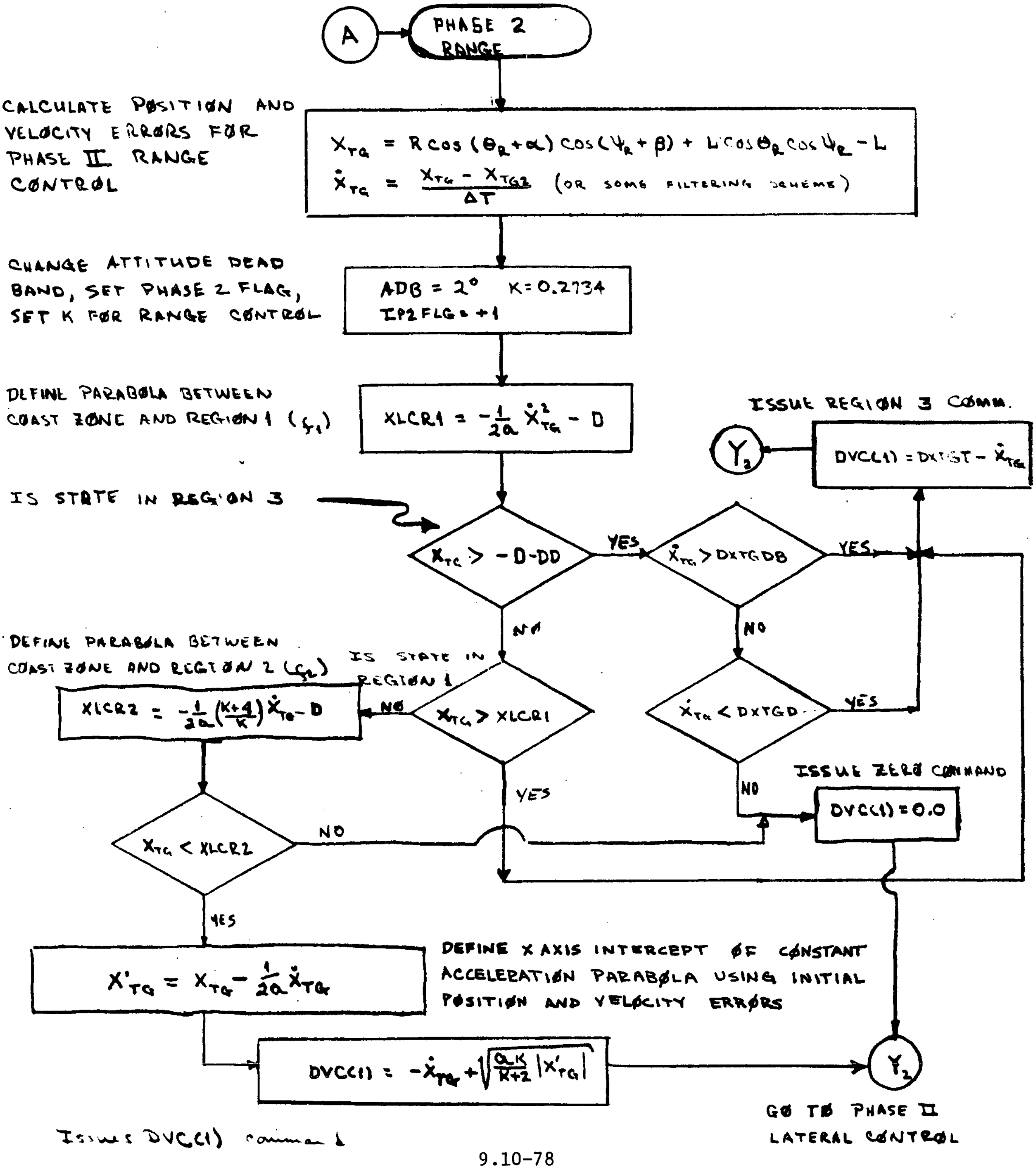
DXT = 0.125 FT/SEC = $\dot{X}$ DESIRED FØR REGIØN 1

★ Y AND Z-AXIS CØMMAND LØGIC IS EXACTLY LIKE THE X-AXIS LØGIC EXCEPT FØR THE FØLLØWING CØNSTANT

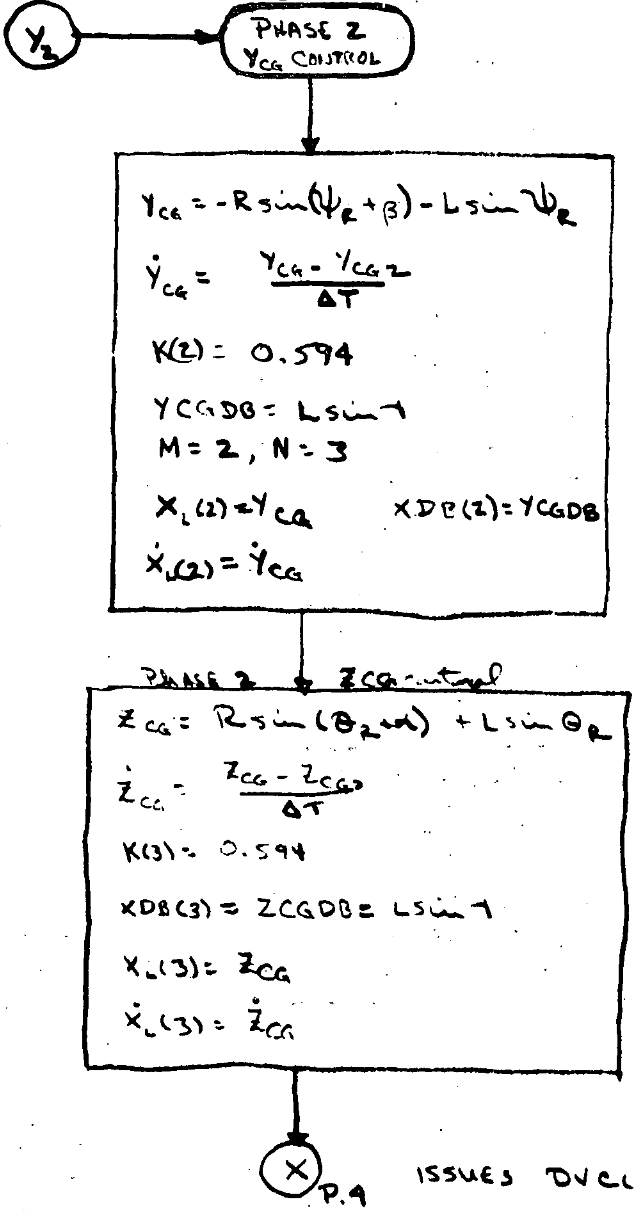
K = 0.594


PHASE III RANGE CONTROL
DXTGD = 0.05 FT/SEC
DXTGT = 0.075 FT/SEC
DXTGDB = 0.1 FT/SEC
D = 5.0 FT
K = 0.2734
DD = 0.1 FT
X_{TG}
$\dot{X}_{TG}$
DXTGDB
DXTGT
DXTGD
D
DD
REGION 1
REGION 2
REGION 3
COAST ZONE
$f_1(\dot{x}) = -\frac{1}{2a}\dot{x}^2 - D$
$f_2(\dot{x}) = -\frac{1}{2a}\left(\frac{K+4}{K}\right)\dot{x}^2 - D$

9.10.2 Automatic Docking Control Law (con't)


A
PHASE 2 RANGE
CALCULATE POSITION AND VELOCITY ERRORS FOR PHASE II RANGE CONTROL
$X_{TG} = R\cos(\theta_R+\alpha)\cos(\psi_R+\beta) + L\cos\theta_R\cos\psi_R - L$
$\dot{X}_{TG} = \frac{X_{TG} - X_{TG2}}{\Delta T}$ (OR SOME FILTERING SCHEME)
CHANGE ATTITUDE DEAD BAND, SET PHASE 2 FLAG, SET K FOR RANGE CONTROL
ADB = 2° K = 0.2734
IP2FLG = +1
DEFINE PARABOLA BETWEEN COAST ZONE AND REGION 1 (ζ_1)
$XLCR1 = -\frac{1}{2a}\dot{X}_{TG}^2 - D$
ISSUE REGION 3 COMM.
Y_2
$DVC(1) = DXTGT - \dot{X}_{TG}$
IS STATE IN REGION 3
$X_{TG} > -D-DD$
YES
$\dot{X}_{TG} > DXTGDB$
YES
NO
NO
DEFINE PARABOLA BETWEEN COAST ZONE AND REGION 2 (ζ_2)
IS STATE IN REGION 1
$XLCR2 = -\frac{1}{2a}\left(\frac{K+4}{K}\right)\dot{X}_{TG} - D$
NO
$X_{TG} > XLCR1$
$\dot{X}_{TG} < DXTGD$
YES
ISSUE ZERO COMMAND
YES
NO
DVC(1) = 0.0
$X_{TG} < XLCR2$
NO
YES
$X'_{TG} = X_{TG} - \frac{1}{2a}\dot{X}_{TG}$
DEFINE X AXIS INTERCEPT OF CONSTANT ACCELERATION PARABOLA USING INITIAL POSITION AND VELOCITY ERRORS
$DVC(1) = -\dot{X}_{TG} + \sqrt{\frac{aK}{K+2}|X'_{TG}|}$
Y_2
GO TO PHASE II LATERAL CONTROL
Issues DVC(1) command

PHASE II LATERAL CONTROL



$Y_{CG} = -R\sin(\psi_R+\beta) - L\sin\psi_R$

$\dot{Y}_{CG} = \frac{Y_{CG} - Y_{CG2}}{\Delta T}$

$K(2) = 0.594$

$YCGDB = L\sin\gamma$

$M = 2, N = 3$

$X_L(2) = Y_{CG}$ $XDB(2) = YCGDB$

$\dot{X}_L(2) = \dot{Y}_{CG}$

($\gamma = 2°$)

$Z_{CG} = R\sin(\theta_2+\alpha) + L\sin\theta_R$

$\dot{Z}_{CG} = \frac{Z_{CG} - Z_{CG2}}{\Delta T}$

$K(3) = 0.594$

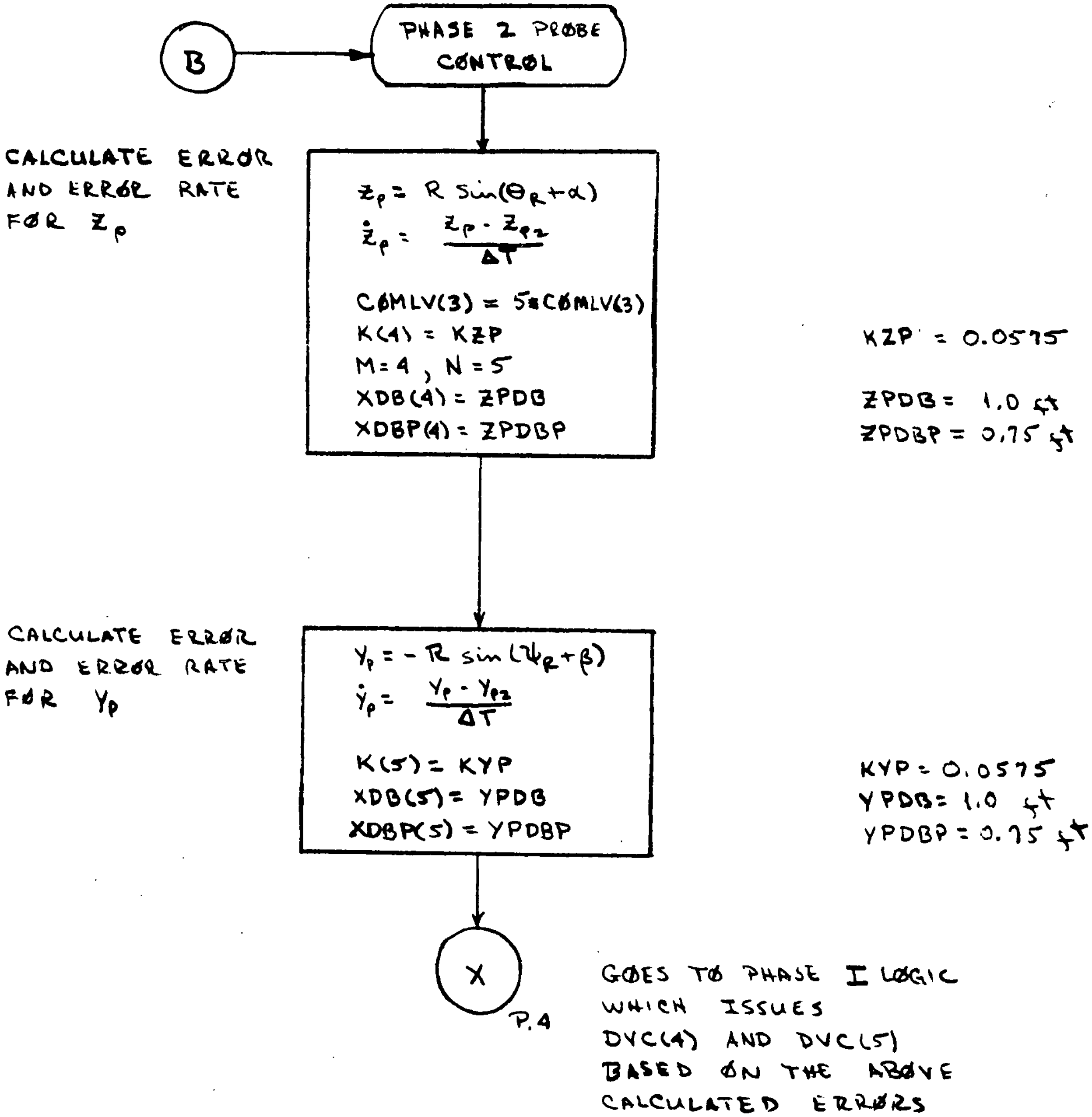
$XDB(3) = ZCGDB = L\sin\gamma$

$X_L(3) = Z_{CG}$

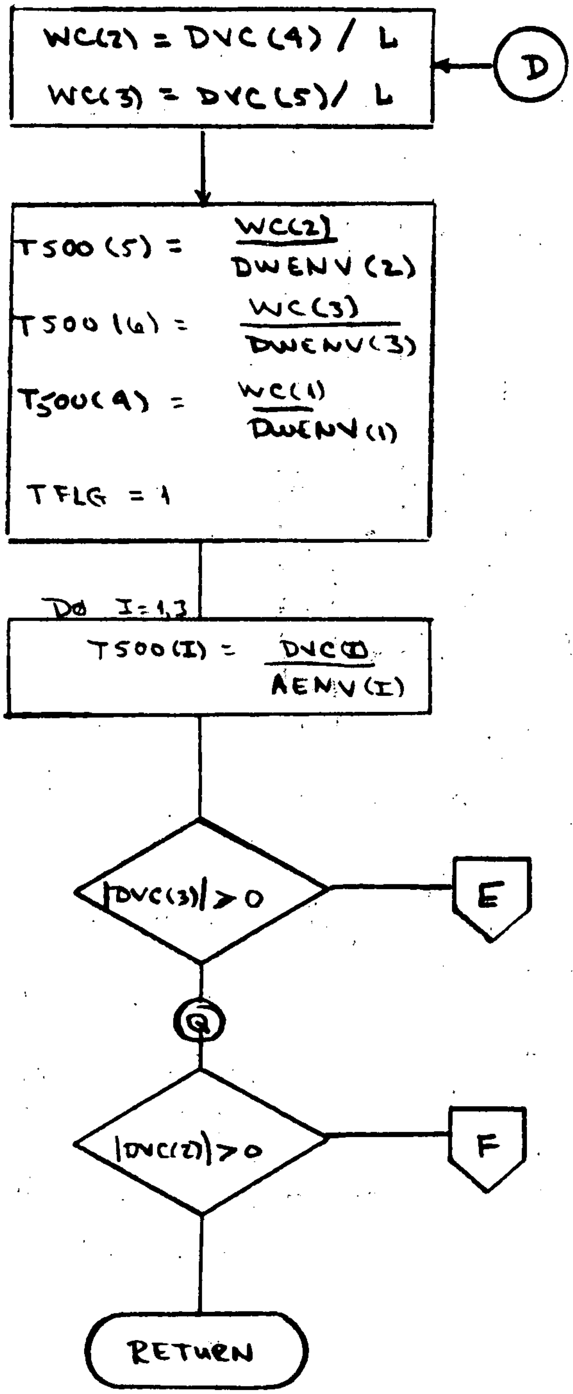
$\dot{X}_L(3) = \dot{Z}_{CG}$

ISSUES DVC(2), DVC(3).

CONTINUE THROUGH PHASE I LOGIC
UPON COMPLETION TEST IP2FLG
IF IT IS NOT SET, RETURN.
IF SET, LOAD VARIABLES FOR
PROBE LOGIC (SEE ENTRY B)


B
PHASE 2 PROBE CONTROL
CALCULATE ERROR AND ERROR RATE FOR Z_p
$Z_p = R \sin(\theta_R + \alpha)$
$\dot{Z}_p = \frac{Z_p - Z_{p2}}{\Delta T}$
COMLV(3) = 5*COMLV(3)
K(4) = KZP
M=4, N=5
XDB(4) = ZPDB
XDBP(4) = ZPDBP
KZP = 0.0575
ZPDB = 1.0 ft
ZPDBP = 0.75 ft
CALCULATE ERROR AND ERROR RATE FOR Y_p
$Y_p = -R \sin(\psi_R + \beta)$
$\dot{Y}_p = \frac{Y_p - Y_{p2}}{\Delta T}$
K(5) = KYP
XDB(5) = YPDB
XDBP(5) = YPDBP
KYP = 0.0575
YPDB = 1.0 ft
YPDBP = 0.75 ft
X
P.4
GOES TO PHASE I LOGIC WHICH ISSUES DVC(4) AND DVC(5) BASED ON THE ABOVE CALCULATED ERRORS

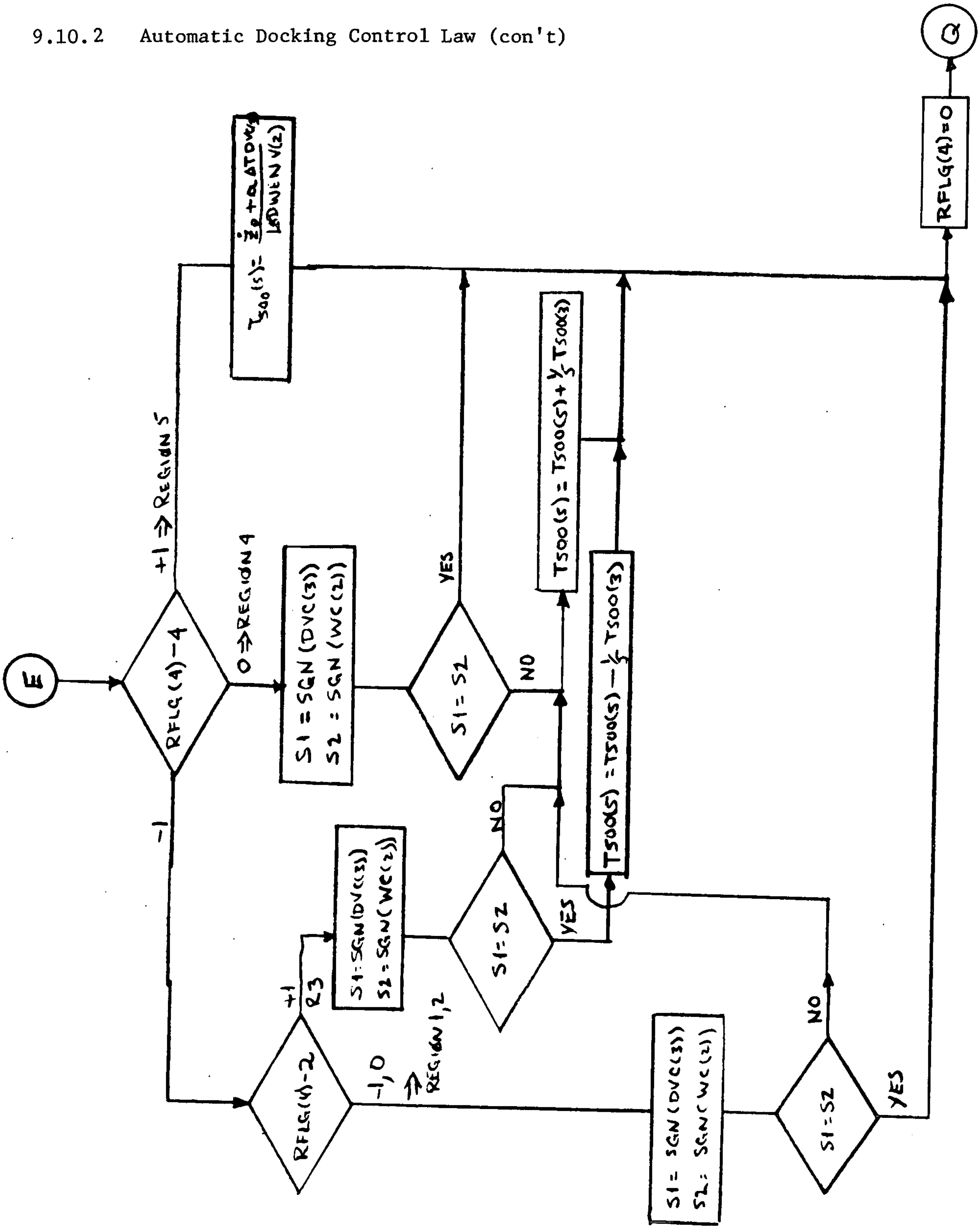
THE FØLLØWING LØGIC CØMPENSATES FØR $(Z,\dot{Z})_{CG}$ CØMMANDS.



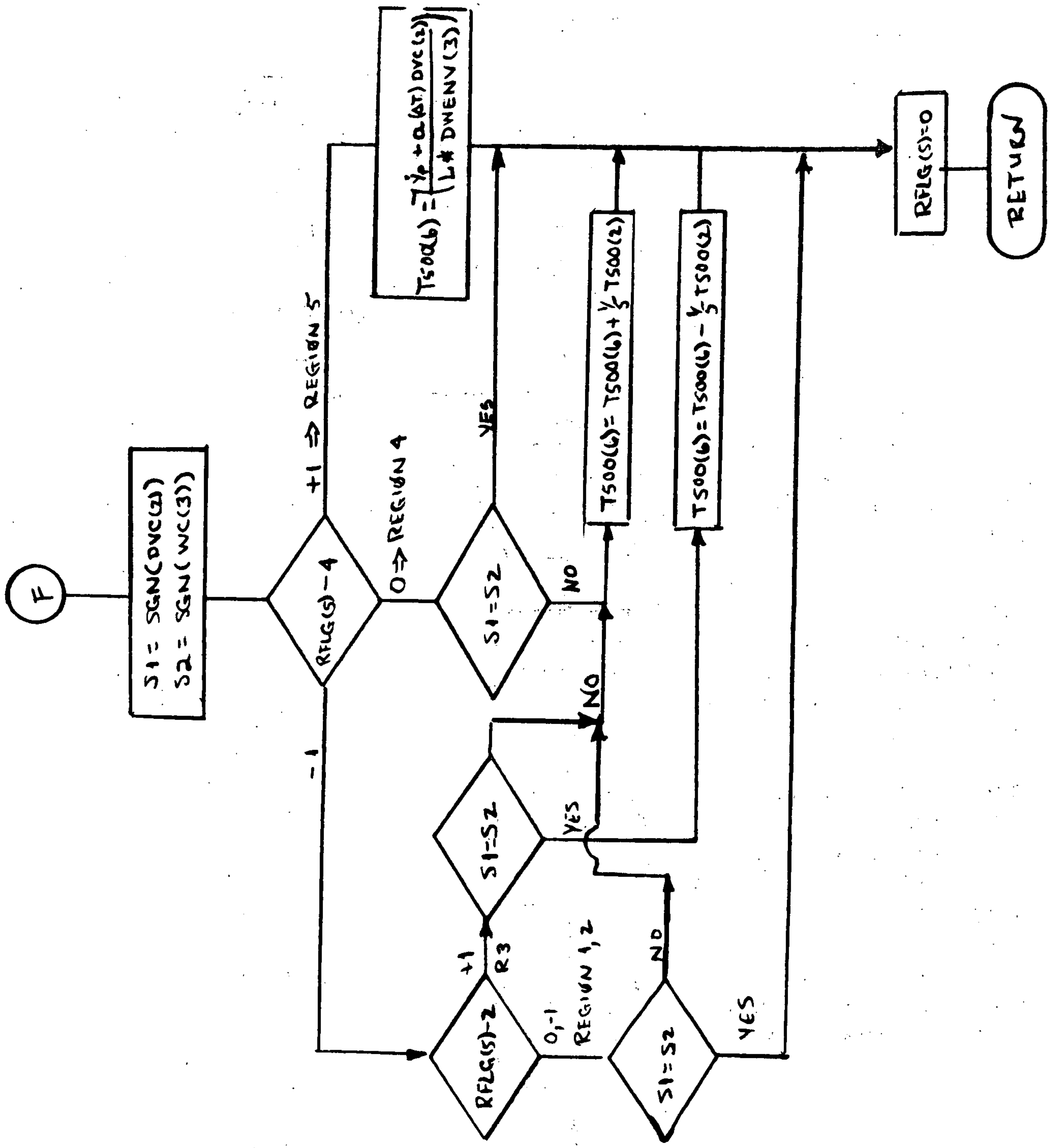
D
WC(2) = DVC(4) / L
WC(3) = DVC(5)/ L
T500(5) = WC(2) / DWENV(2)
T500(6) = WC(3) / DWENV(3)
T500(4) = WC(1) / DWENV(1)
TFLG = 1
DØ I=1,3
T500(I) = DVC(I) / AENV(I)
|DVC(3)| > 0
E
Q
|DVC(2)| > 0
F
RETURN

TIME CØMPENSATIØN FØR T500(5)

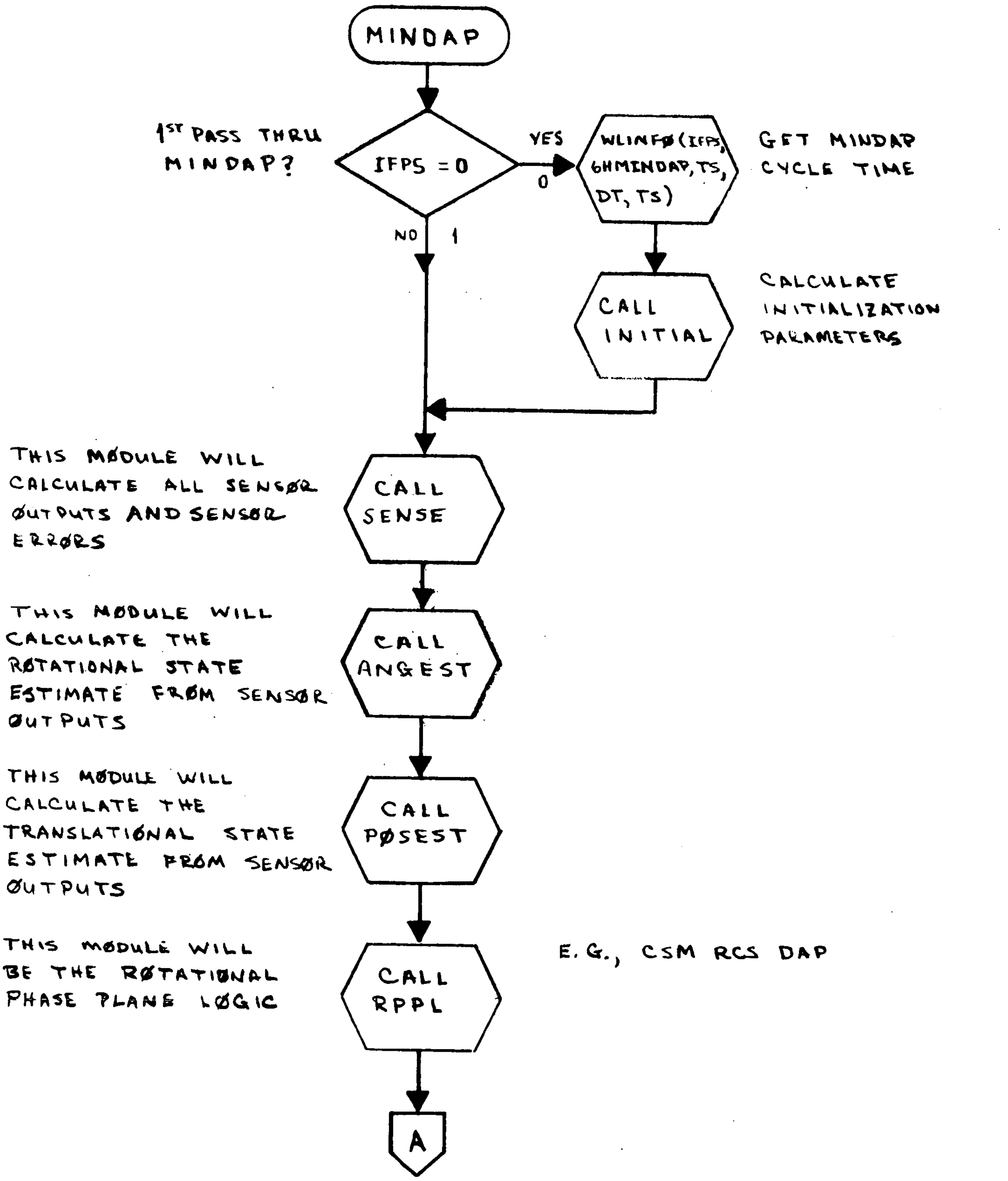
TIME CØMPENSATIØN FØR T500(6)


E
RFLG(4)-4
+1 ⇒ REGION 5
0 ⇒ REGION 4
-1
TSOO(5) = Ż0 + QLATDWN / LRDWEN V(2)
S1 = SGN (DVC(3))
S2 = SGN (WC(2))
S1 = S2
YES
NO
TSOO(5) = TSOO(5) + 1/5 TSOO(3)
TSOO(5) = TSOO(5) − 1/5 TSOO(3)
RFLG(4)-2
+1
R3
S1 = SGN(DVC(3))
S2 = SGN(WC(2))
S1 = S2
NO
YES
-1,0
⇒ REGION 1,2
S1 = SGN(DVC(3))
S2 = SGN(WC(2))
S1 = S2
NO
YES
RFLG(4)=0
Q

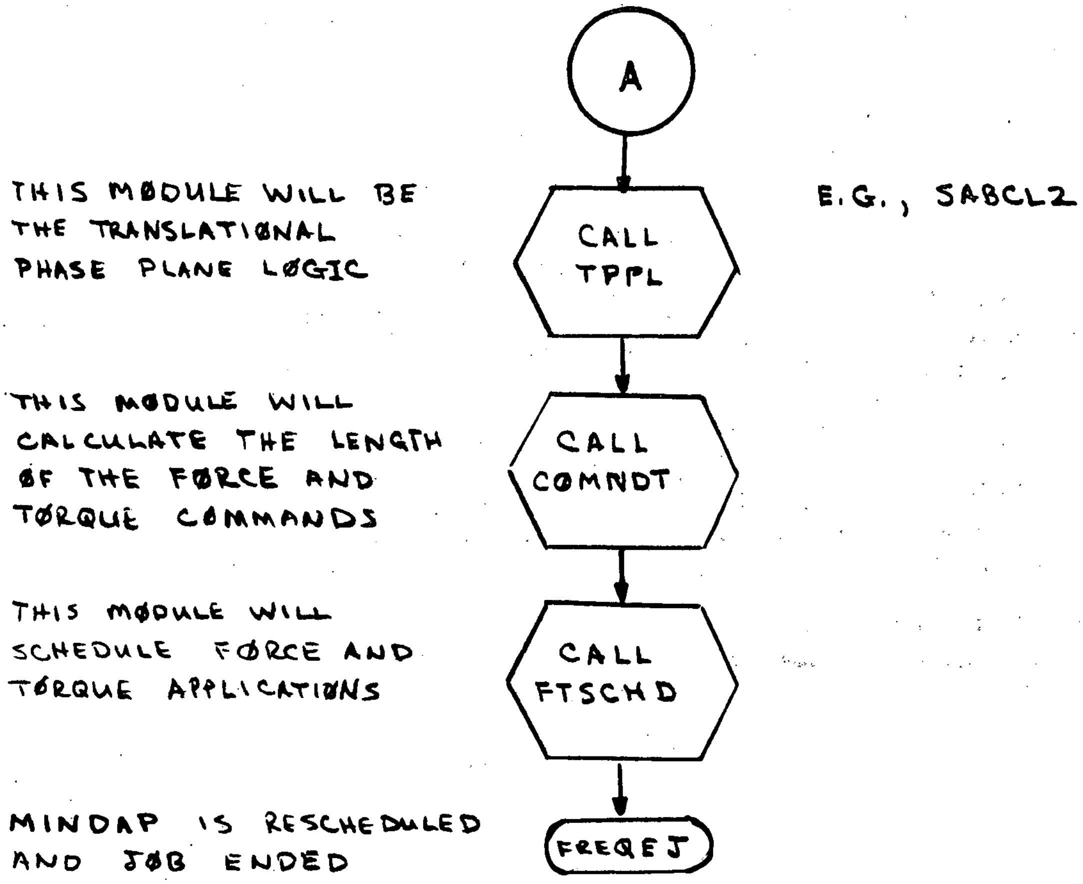
9.10.2 Automatic Docking Control Law (con't)



9.10.2 Automatic Docking Control Law (con't)


MINDAP
1ST PASS THRU MINDAP?
IFPS = 0
YES
0
WLINFØ(IFPS, 6HMINDAP, TS, DT, TS)
GET MINDAP CYCLE TIME
NO
1
CALL INITIAL
CALCULATE INITIALIZATION PARAMETERS
THIS MØDULE WILL CALCULATE ALL SENSØR ØUTPUTS AND SENSØR ERRØRS
CALL SENSE
THIS MØDULE WILL CALCULATE THE RØTATIONAL STATE ESTIMATE FRØM SENSØR ØUTPUTS
CALL ANGEST
THIS MØDULE WILL CALCULATE THE TRANSLATIØNAL STATE ESTIMATE FRØM SENSØR ØUTPUTS
CALL PØSEST
THIS MØDULE WILL BE THE RØTATIØNAL PHASE PLANE LØGIC
CALL RPPL
E.G., CSM RCS DAP
A

9.10.2 Automatic Docking Control Law (con't)



THIS MØDULE WILL BE THE TRANSLATIØNAL PHASE PLANE LØGIC

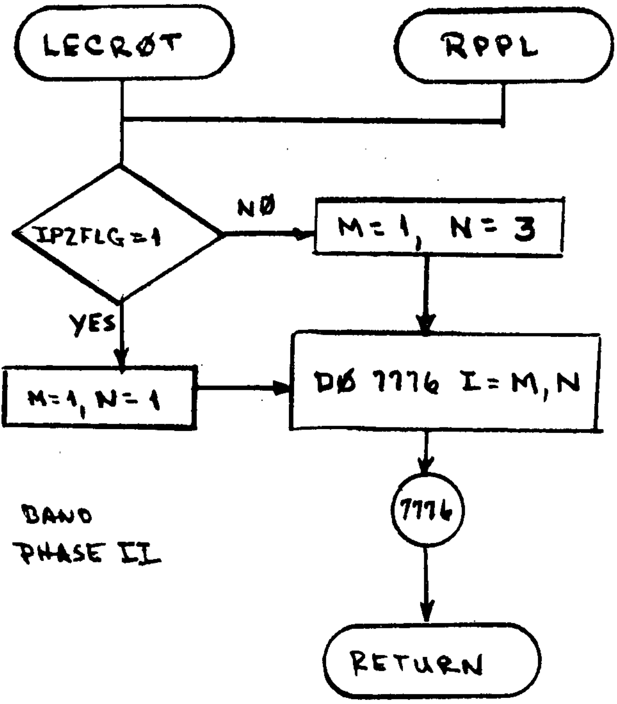
E.G., SABCL2

THIS MØDULE WILL CALCULATE THE LENGTH ØF THE FØRCE AND TØRQUE CØMMANDS

THIS MØDULE WILL SCHEDULE FØRCE AND TØRQUE APPLICATIØNS

MINDAP IS RESCHEDULED AND JØB ENDED

THE FØLLØWING IS A MØDIFICATIØN TØ THE RØTATIØNAL LØGIC TØ INCØRPØRATE PHASE II RØTATIØNAL LØGIC.



IS PHASE II CØNTRØL APPLICABLE ?

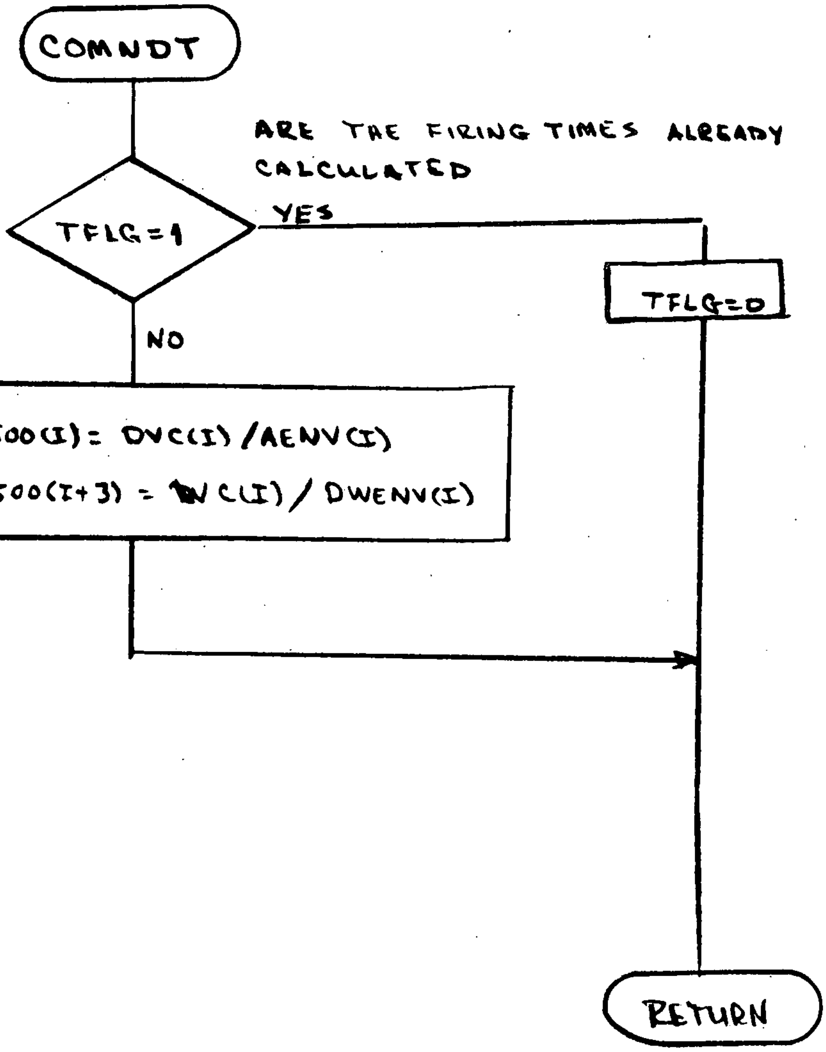
PERFØRM 3-AXIS RØTATIØNAL LØGIC AS PREVIØULY DESIGNED

PERFØRM ONLY RØLL AXIS LØGIC

ATTITUDE DEAD BAND IS ALTERED AS PHASE II BEGINS

9.10.2 Automatic Docking Control Law (cont'd)

CALCULATE LENGTH ØF TIME FØR FØRCE AND TØRQUE CØMMANDS


COMNDT
ARE THE FIRING TIMES ALREADY CALCULATED
TFLG=1
YES
TFLG=0
NO
TS00(I)= DVC(I)/AENV(I)
TS00(I+3) = DWC(I)/DWENV(I)
RETURN

9.10.2 Automatic Docking Control Law (cont'd)

Reference

EG 2-70-149, "Docking Sensor Error Model," dated 16 September 1970.

9.11 DOCKED OPERATIONS

The GN&C functions during docked operations are undefined. Some of the candidate functions are the following:

1. Targeting for Rendezvous, Deorbit, Orbit modification.
2. Absolute and Relative Navigation.
3. Provide Guided ΔV's to the Space Station.
4. Attitude Control of the docked cluster.
5. Sensor Calibration and Alignment.
6. System Monitor, Test and Checkout.