5.5 BASIC SUBROUTINES

5.5.1 GENERAL COMMENTS

The basic solar system and conic trajectory subroutines which are used by the various guidance and navigation routines are described in this section.

5.5.1.1 Solar System Subroutines

The subroutines used to determine the translation and rotation of the relevant solar system bodies (earth, moon and sun) are designed specifically for a fourteen day lunar landing mission. The method of computing the position and velocity of the moon and the sun relative to the earth is given in Section 5.5.4. The transformations between the Basic Reference Coordinate System and the Earth- and Moon-fixed Coordinate Systems are described in Section 5.5.2. The procedure for transforming between vectors in the Basic Reference Coordinate System and latitude, longitude, altitude coordinates is given in Section 5.5.3. Although these subroutines are normally used in the lunar landing mission, they are valid for use in any mission of not more than fourteen days duration in earth-moon space.

5.5.1.2 Conic Trajectory Subroutines

This is a description of a group of conic trajectory subroutines which are frequently used by higher level routines and programs in both the Command Module and the Lunar Module computers.

These subroutines, whose block diagrams are presented in Sections 5.5.5 to 5.5.10, provide solutions to the following conic problems. (See nomenclature which follows)

(1) Given $\underline{r}(t_1)$, $\underline{v}(t_1)$, t_D; solve for $\underline{r}(t_2)$, $\underline{v}(t_2)$

(Kepler Subroutine)

(2) Given $\underline{r}(t_1)$, $\underline{r}(t_2)$, t_{D21}, s_G; solve for $\underline{v}(t_1)$

(Lambert Subroutine)

(3) Given $\underline{r}(t_1)$, $\underline{v}(t_1)$, θ; solve for t_{21}, $\underline{r}(t_2)$, $\underline{v}(t_2)$

(Time-Theta Subroutine)

(4) Given $\underline{r}(t_1)$, $\underline{v}(t_1)$, $r(t_2)$, $s_{\dot{r}}$; solve for t_{21}, $\underline{r}(t_2)$, $\underline{v}(t_2)$

(Time-Radius Subroutine)

(5) Given $\underline{r}(t)$, $\underline{v}(t)$; solve for r_P, r_A, e

(Apsides Subroutine)

In addition, the following useful subroutines are provided.

(6) Conic Parameters Subroutine (See Fig. 5.10-1).

(7) Geometric Parameters Subroutine (See Fig. 5.10-2).

(8) Iterator Subroutine (See Fig. 5.10-3).

The solutions to the above set of conic problems have stringent accuracy requirements. Programming the fixed-point Apollo computer introduces two constraints which determine accuracy limitations: the 28 bit double precision word length, and the range of variables which is several orders of magnitude for the Apollo mission.

In order to maintain numerical accuracy when these subroutines are programmed into the Apollo computer, floating point programming techniques must be exercised. The effect is for even a simple equation to require a large number of computer instructions. The alternative to this is to separate the problem into phases, each with a different variable range. This, however, requires an even larger number of instructions. These considerations provide the incentive for efficiently organizing the conic equations as shown in the block diagrams.

In addition to the requirement for accuracy, the solution to the Kepler and Lambert Problems must be accomplished in a minimum of computation time in order that the guidance system operate satisfactorily in real time. This additional constraint dictates that a minimum of computer instructions be performed when solving the problem.

Method of Solution

To minimize the total number of computer instructions, the problems are solved in the "universal" form; i.e. only equations which are equally valid for the ellipse, parabola and hyperbola are used. Also these subroutines can be used with either the earth or the moon as the attracting body.

Kepler's equation, in the universal form, is utilized to relate transfer time to the conic parameters. All other necessary equations are also universal. The Kepler and Lambert problems are solved with a single iteration loop utilizing a simple first-order slope iterator. In the case of the Kepler problem a third order approximation is available to produce the initial guess for the independent variable (See Eq.(2. 2. 4) of Section 5. 2. 2. 2).

Sections 5. 5. 5 thru 5. 5. 10 provide block diagrams of the detailed computational procedures for solving the various problems. The equations are presented in block diagram form with the nomenclature defined below.

Range of Variables

As indicated previously, the programming of the conic subroutines requires a careful balance between accuracy, computational speed and number of instructions. This balance, in the Apollo Guidance Computer, leaves very little margin in any of these areas.

Since the values of problem variables are determined by the solution of the problem being solved and since the problem may originate from the ground system, it is essential that the variable range limitations be defined. The conic routines are incapable of handling problems when the solution lies outside of the range.

The following is a list of the maximum allowable numeric values of the variables. Note that, in addition to fundamental quantities such as position and velocity, there are limitations on intermediate variables and combinations of variables.

Scaling for Conic Subroutines (Sections 5.5.5 to 5.5.10)

Parameter	Maximum Value* Earth Primary Body	Maximum Value* Moon Primary Body
r	2^{29}	2^{27}
v	2^{7}	2^{5}
t	2^{28}	2^{28}
α**	2^{-22}	2^{-20}
α_N**	2^{6}	2^{6}
p_N	2^{4}	2^{4}
$\cot \gamma$	2^{5}	2^{5}
$\cot \frac{\theta}{2}$	2^{5}	2^{5}
x	2^{17}	2^{16}
$\xi = \alpha x^2$***	-50	-50
	$+4\pi^2$	$+4\pi^2$
$c_1 = \underline{r} \cdot \underline{v} / \sqrt{\mu}$	2^{17}	2^{16}
$c_2 = r v^2 / \mu - 1$	2^{6}	2^{6}
$\lambda = r(t_1)/r(t_2)$	2^{7}	2^{7}
$\cos \theta - \lambda$	2^{7}	2^{7}

* All dimensional values are in units of meters and centiseconds

** The maximum absolute value occurs for negative values of this parameter.

***Both the maximum and minimum values are listed since neither may be exceeded.

	Maximum Value*	
Parameter	Earth	Moon
e	2^3	2^3
x^2	2^{34}	2^{32}
$x^2\, c(\xi)$	2^{33}	2^{31}
$x^3\, s(\xi)/\sqrt{\mu}$	2^{28}	2^{28}
$c_1\, x^2\, c(\xi)$	2^{49}	2^{46}
$c_2\, x^2\, s(\xi)$	2^{35}	2^{33}
$x\,[c_2\, x^2\, s(\xi) + r(t_1)]$	2^{49}	2^{46}
$\xi\, s(\xi)$	2^7	2^7
$x^2\, c(\xi)/r$	2^8	2^8
$\sqrt{\mu}\, x\,(\xi\, s(\xi) - 1)/r(t_2)$	2^{15}	2^{13}
$c(\xi)$	2^4	2^4
$s(\xi)$	2^1	2^1

* All dimensional values are in units of meters and centi

Nomenclature for Conic Subroutines (Sections 5.5.5 to 5.5.10)

$\underline{r}(t_1)$	initial position vector
$\underline{v}(t_1)$	initial velocity vector
$\underline{r}(t_2)$	terminal position vector
$\underline{v}(t_2)$	terminal velocity vector
$\underline{u}_N$	unit normal in the direction of the angular momentum vector
α	reciprocal of semi-major axis (negative for hyperbolas)
r_P	radius of pericenter
r_A	radius of apocenter
e	eccentricity
α_N	ratio of magnitude of initial position vector to semi-major axis
p_N	ratio of semi-latus rectum to initial position vector magnitude
γ	inertial flight path angle as measured from vertical
θ	true anomaly difference between $\underline{r}(t_1)$ and $\underline{r}(t_2)$
f	true anomaly of r (t_2)

x	a universal conic parameter equal to the ratio of eccentric anomaly difference to $\sqrt{+\alpha}$ for the ellipse, or the ratio of the hyperbolic analog of eccentric anomaly difference to $\sqrt{-\alpha}$ for the hyperbola
x'	value of x from the previous Kepler solution
t_{21}	computed transfer time from Kepler's equation $(t_2 - t_1)$
t'_{21}	transfer time corresponding to the previous solution of Kepler's equation
t_D	desired transfer time through which the conic update of the state vector is to be made
t_{D21}	desired transfer time to traverse from $\underline{r}(t_1)$ to $\underline{r}(t_2)$
t_{ERR}	error in transfer time
ϵ_t	fraction of desired transfer time to which t_{ERR} must converge
Δx	increment in x which will produce a smaller t_{ERR}
ϵ_x	value of Δx which will produce no significant change in t_{21}
$\Delta \cot \gamma$	increment in $\cot \gamma$ which will decrease the magnitude of t_{ERR}
ϵ_c	value of $\Delta \cot \gamma$ which will produce no significant change in t_{21}

μ	product of universal gravitational constant and mass of the primary attracting body
x_{MAX}	maximum value of x
x_{MIN}	minimum value of x
cot_{MAX}	maximum value of cot γ
cot_{MIN}	minimum value of cot γ
ℓ_{MAX}	upper bound of general independent variable
ℓ_{MIN}	lower bound of general independent variable
x_{MAX1}	absolute upper bound on x with respect to the moon
x_{MAX0}	absolute upper bound on x with respect to the earth
k	a fraction of the full range of the independent variable which determines the increment of the independent variable on the first pass through the iterator
y	general dependent variable
y'	previous value of y
y_{ERR}	error in y
z	general independent variable

Δz	increment in z which will produce a smaller y_{ERR}
s_G	a sign which is plus or minus according to whether the true anomaly difference between $\underline{r}(t_1)$ and $\underline{r}(t_2)$ is to be less than or greater than 180 degrees
$s_{\dot{r}}$	a sign which is plus or minus according to whether the desired radial velocity at $r(t_2)$ is plus or minus
$\underline{\eta}_1$	general vector # 1
$\underline{\eta}_2$	general vector # 2
ϕ	angle between $\underline{\eta}_1$ and $\underline{\eta}_2$
f_1	a switch set to 0 or 1 according to whether a guess of cot γ is available or not
f_2	a switch set to 0 or 1 according to whether Lambert should determine $\underline{u}_N$ from $\underline{r}(t_1)$ and $\underline{r}(t_2)$ or $\underline{u}_N$ is an input
f_3	a tag set to 0 or 1 according to whether the iterator should use the "Regula Falsi" or bias method
f_4	a flag set to 0 or 1 according to whether the iterator is to act as a first order or a second order iterator
f_5	a flag set to 0 or 1 according to whether Lambert converges to a solution or not

f_6	a switch set to 0 or 1 according to whether or not the new state vector is to be an additional output requirement of the Time-Theta or Time-Radius problems.
f_7	a flag set to 1 if the inputs require that the conic trajectory must close through infinity
f_8	a flag set to 1 if the Time-Radius problem was solved for pericenter or apocenter instead of $r(t_2)$
f_9	a flag set to 1 if the input to the Time-Radius Subroutine produced an e less than 2^{-18}.
t_p	period of the orbit
t_R	integral periods subtracted from t_D to produce a t_D less than t_p
x_R	value of x corresponding to t_R
k_1	the minimal acceptance percentage of t_{D21} to which t_{ERR} must converge
n_1	a flag set to 0 or 1 according to whether or not the velocity vector at the terminal position is to be an additional output requirement of the Lambert Subroutine

5.5.2 PLANETARY INERTIAL ORIENTATION SUBROUTINE

This subroutine is used to transform vectors between the Basic Reference Coordinate System and a Planetary (Earth-fixed or Moon-fixed) Coordinate System at a specified time. These three coordinate systems are defined in Section 5.1.4.

Let $\underline{r}$ be a vector in the Basic Reference Coordinate System, $\underline{r}_P$ the same vector expressed in the Planetary Coordinate System, and t the specified ground elapsed time (GET). Then,

$$\underline{r}_P = M(t)\,(\underline{r} - \underline{\ell} \times \underline{r}) \qquad (5.2.1)$$

and

$$\underline{r} = M^T(t)\,(\underline{r}_P + \underline{\ell}_P \times \underline{r}_P) \qquad (5.2.2)$$

where M(t) is a time dependent orthogonal transformation matrix, $\underline{\ell}$ is a small rotation vector in the Basic Reference Coordinate System, and $\underline{\ell}_P$ is the same vector $\underline{\ell}$ expressed in the Planetary Coordinate System. The vector $\underline{\ell}$ is considered constant in one coordinate system for the duration of the mission. The method of computing M(t) and $\underline{\ell}$ depends on whether the relevant planet is the earth or the moon.

Case I - Earth

For the earth, the matrix M(t) describes a rotation about the polar axis of the earth (the Z-axis of the Earth-fixed Coordinate

System), and the vector $\underline{\ell}$ accounts for the precession and nutation of the polar axis (the deviation of the true pole from the mean pole).

Let A_X and A_Y be the small angles about the X- and Y-axes of the Basic Reference Coordinate System, respectively, that describe the precession and nutation of the earth's polar axis. The values of these two angles at the midpoint of the mission are included in the pre-launch erasable data load and are considered constant throughout the flight. Then,

$$\underline{\ell} = \begin{pmatrix} A_X \\ A_Y \\ 0 \end{pmatrix}$$

$$A_Z = A_{Z0} + \omega_E (t + t_0) \qquad (5.2.3)$$

$$M(t) = \begin{pmatrix} \cos A_Z & \sin A_Z & 0 \\ -\sin A_Z & \cos A_Z & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\underline{\ell}_P = M(t)\, \underline{\ell}$$

where A_{Z0} is the angle between the X-axis of the Basic Reference Coordinate System and the X-axis of the Earth-fixed Coordinate System (the intersection of the Greenwich meridian and the equatorial plane of the earth) at July 1.0, 1968 universal time (i.e., midnight at Greenwich just prior to July 1, 1968), t_0 is the elapsed time between July 1.0, 1968 universal time and the time that the computer clock was zeroed, and ω_E is the angular velocity of the earth.

Case II - Moon

For the moon, the matrix M(t) accounts for the difference in orientation of the Basic Reference and Moon-fixed Coordinate Systems in exact accordance with Cassini's laws, and the rotation vector $\underline{\ell}$ corrects for deviations from the above orientation because of physical libration.

Define the following three angles which are functions of time:

B = the obliquity, the angle between the mean earth equatorial plane and the plane of the ecliptic.

Ω_I = the longitude of the node of the moon's orbit measured from the X-axis of the Basic Reference Coordinate System.

F = the angle from the mean ascending node of the moon's orbit to the mean moon.

Let I be the constant angle between the mean lunar equatorial plane and the plane of the ecliptic (1° 32.1'). Then, the sequence of rotations which brings the Basic Reference Coordinate System into coincidence with the Moon-fixed Coordinate System (neglecting libration) is as follows:

Rotation	Axis of Rotation	Angle of Rotation
1	X	B
2	Z	Ω_I
3	X	-I
4	Z	$\pi + F$

The transformation matrices for these rotations are, respectively,

$$M_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos B & \sin B \\ 0 & -\sin B & \cos B \end{pmatrix}$$

$$M_2 = \begin{pmatrix} \cos \Omega_I & \sin \Omega_I & 0 \\ -\sin \Omega_I & \cos \Omega_I & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$M_3 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos I & -\sin I \\ 0 & \sin I & \cos I \end{pmatrix} \qquad (5.2.4)$$

$$M_4 = \begin{pmatrix} -\cos F & -\sin F & 0 \\ \sin F & -\cos F & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

The matrix M(t) is then given by

$$M(t) = M_4 \, M_3 \, M_2 \, M_1 \qquad (5.2.5)$$

The following approximate method is used to determine the transformation between the Basic Reference and Moon-fixed Coordinate Systems.

The angles B, Ω_I and F are computed as linear functions of time. Let $\underline{\ell}_M$ be the value of the vector libration $\underline{\ell}_P$ (expressed in the Moon-fixed Coordinate System) at the midpoint of the mission. The vector $\underline{\ell}_M$ is included in the pre-launch erasable data load and is considered constant throughout the flight. Then,

$$\begin{aligned}
\underline{\ell}_P &= \underline{\ell}_M \\
t_M &= t + t_0 \\
B &= B_0 + \dot{B}\, t_M \\
\Omega_I &= \Omega_{I0} + \dot{\Omega}_I\, t_M \\
F &= F_0 + \dot{F}\, t_M \\
\underline{a} &= \begin{pmatrix} \cos \Omega_I \\ \cos B \, \sin \Omega_I \\ \sin B \, \sin \Omega_I \end{pmatrix} \\
\underline{b} &= \begin{pmatrix} -\sin \Omega_I \\ \cos B \, \cos \Omega_I \\ \sin B \, \cos \Omega_I \end{pmatrix}
\end{aligned} \qquad (5.2.6)$$

$$
\begin{aligned}
\underline{c} &= \begin{pmatrix} 0 \\ -\sin B \\ \cos B \end{pmatrix} \\
\underline{d} &= \underline{b}\, C_I - \underline{c}\, S_I \\
\underline{m}_2 &= \underline{b}\, S_I + \underline{c}\, C_I \\
\underline{m}_0 &= -\underline{a} \cos F - \underline{d} \sin F \qquad\qquad (5.2.6)\ \text{(cont.)} \\
\underline{m}_1 &= \underline{a} \sin F - \underline{d} \cos F \\
M(t) &= \begin{pmatrix} \underline{m}_0^T \\ \underline{m}_1^T \\ \underline{m}_2^T \end{pmatrix} \\
\underline{\ell} &= M^T(t)\, \underline{\ell}_P
\end{aligned}
$$

where B_0, Ω_{I0}, and F_0 are the values of the angles B, Ω_I and F, respectively, at July 1.0, 1968 universal time; $\dot{B}$, $\dot{\Omega}_I$ and $\dot{F}$ are the rates of change of these angles; and C_I and S_I are the cosine and sine, respectively, of the angle I. Time t_M is defined in Section 5.5.4.

5.5.3 LATITUDE-LONGITUDE SUBROUTINE

For display and data load purposes, the latitude, longitude, and altitude of a point near the surface of the earth or the moon are more meaningful and more convenient to use than the components of a position vector. This subroutine is used to transform position vectors between the Basic Reference Coordinate System and Geographic or Selenographic latitude, longitude, altitude at a specified time.

In the case of the moon, the altitude is computed above either the landing site radius, r_{LS}, or the mean lunar radius, r_M. For the earth, the altitude is defined with respect to either the launch pad radius, r_{LP}, or the radius of the Fischer ellipsoid, r_F, which is computed from

$$r_F^2 = \frac{b^2}{1 - (1 - \frac{b^2}{a^2})(1 - SINL^2)} \qquad (5.3.1)$$

where a and b are the semi-major and semi-minor axes of the Fischer ellipsoid, respectively, and SINL is the sine of the geocentric latitude.

The computational procedures are illustrated in Figs. 5.3-1, 5.3-2, and 5.3-3. The calling program must specify either a vector $\underline{r}$ or latitude (Lat), longitude (Long), and altitude (Alt). In addition, the program must set the time t and the two indicators P and F where

$$P = \begin{cases} 0 & \text{for earth} \\ 1 & \text{for moon} \end{cases}$$

$$F = \begin{cases} 1 \text{ for Fischer ellipsoid or mean lunar radius} \\ 0 \text{ for launch pad or landing site radius} \end{cases}$$

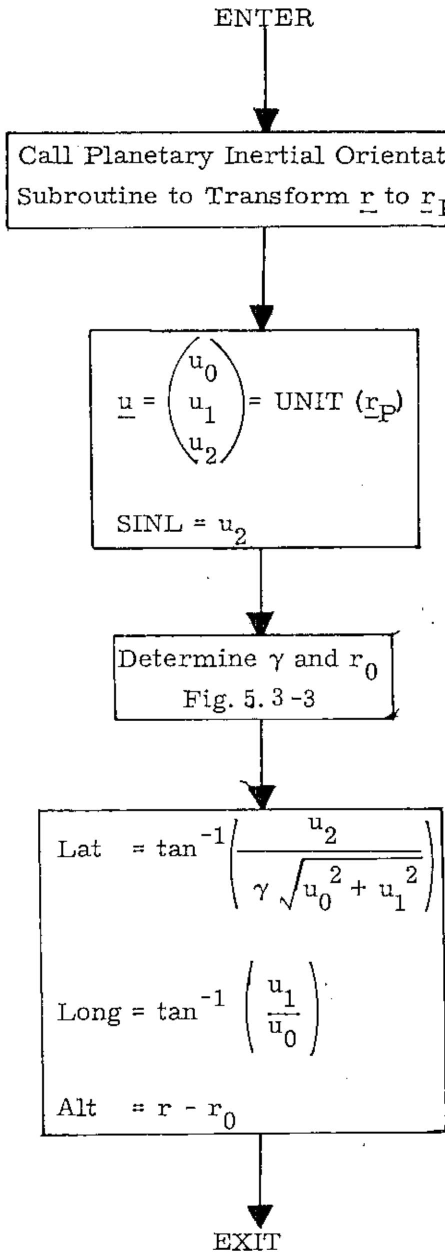


Fig. 5.3-1 Vector to Latitude, Longitude, Altitude Computation Logic Diagram

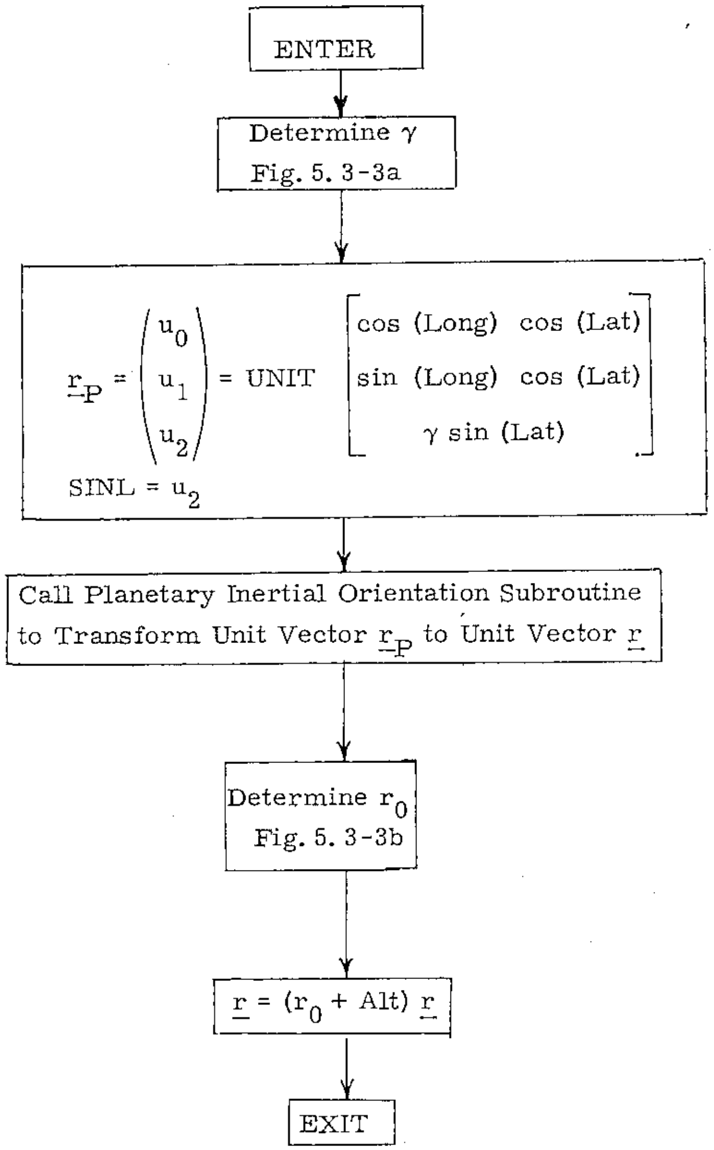


Fig. 5.3-2 Latitude, Longitude, Altitude to Vector Computation Logic Diagram

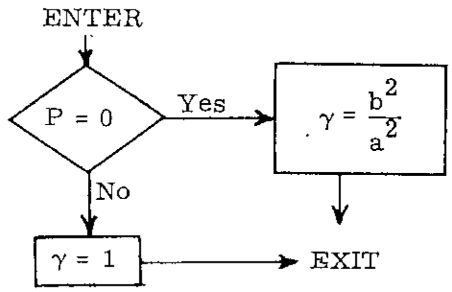


Figure 5.3-3a Determination of γ

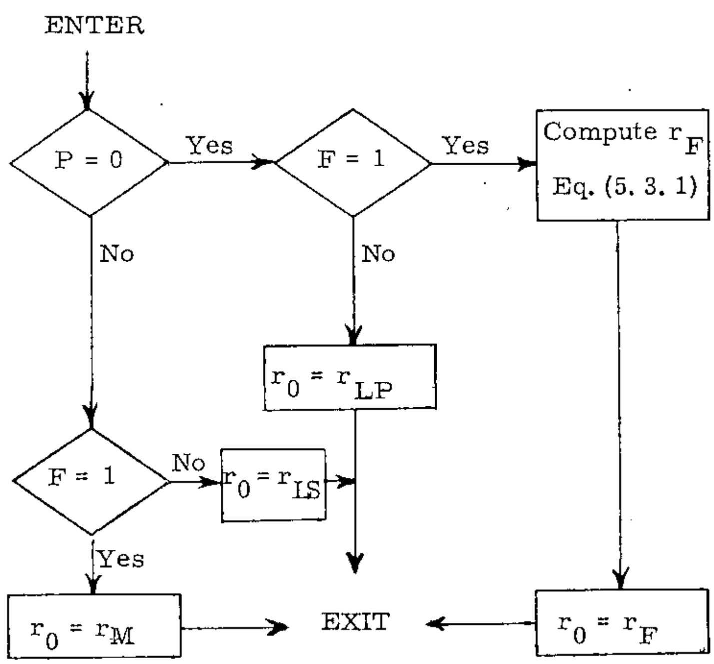


Figure 5.3-3b Determination of r_0

5.5.4 LUNAR AND SOLAR EPHEMERIDES

This subroutine is used to determine the position and velocity vectors of the sun and the moon relative to the earth. The position vectors of the moon and the sun are needed by the Coasting Integration Routine to compute gravity perturbations (Section 5.2.2.3). The velocity of the moon is used by the Coasting Integration Routine when a change in the origin of the coordinate system is performed at the sphere of influence of the moon (Fig. 2.2-3). The velocity of the sun is required, but not very accurately, to compute aberration corrections to optical sightings.

The position of the moon is stored in the computer in the form of a ninth-degree polynominal approximation which is valid over a 14.5 day interval beginning at noon ephemeris time on the day of the launch. The following parameters are included in the pre-launch erasable data load:

t_{M0} = the elapsed time between July 1.0, 1968 universal time and the time at the center of the range over which the lunar-position polynominal is valid. The value of t_{M0} will be an integral number of quarter days minus the difference between ephemeris time and universal time.

$\underline{c}_0$ to $\underline{c}_9$ = vector coefficients

Let t be the specified ground elapsed time (GET), and t_0 be the elapsed time between July 1.0, 1968 universal time and the time that the computer clock was zeroed. Then, the approximate position and velocity of the moon are computed from

$$t_M = t + t_0 - t_{M0} \qquad (5.4.1)$$

$$\underline{r}_{EM} = \sum_{i=0}^{9} \underline{c}_i \, t_M^{\,i} \qquad (5.4.2)$$

$$\underline{v}_{EM} = \sum_{i=1}^{9} i \, \underline{c}_i \, t_M^{\,i-1} \qquad (5.4.3)$$

The approximate position and velocity of the sun are computed from the following items which are included in the pre-launch erasable data load:

$\underline{r}_{ES0}$, $\underline{v}_{ES0}$ = the position and velocity vectors of the sun relative to the earth at time t_{M0}.

ω_{ES} = the angular velocity of the vector $\underline{r}_{ES0}$ at time t_{M0}

Then,

$$\underline{r}_{ES} = \underline{r}_{ES0} \cos(\omega_{ES} \, t_M) + \left[\underline{r}_{ES0} \times \text{UNIT}(\underline{v}_{ES0} \times \underline{r}_{ES0}) \right] \sin(\omega_{ES} \, t_M) \qquad (5.4.4)$$

$$\underline{v}_{ES} = \underline{v}_{ES0} \qquad (5.4.5)$$

5.5.5 KEPLER SUBROUTINE

The Kepler Subroutine solves for the two body position and velocity vectors at the terminal position given the initial position and velocity vectors and a transfer time to the terminal position.

This section contains information to aid the reader in understanding the less obvious aspects of the Kepler Subroutine block diagram depicted in Figs. 5.5-1 thru 5.5-3. The subroutines referred to in these figures are presented in Section 5.5.10. Nomenclature is found in Section 5.5.1.2.

Prior to entering the Kepler Subroutine an initial estimate of x can be generated via Eq. (2.2.4) of Section 5.2.2.2 with $\frac{\Delta t}{2} = t_D - t_{21}$ and $\tau = t_D$. However, x' and t_{21} are non-zero only if the subroutine is being used repetitively.

Although, theoretically, there is no upper bound on x, the practical bound is set to x_{MAX0} or x_{MAX1} to eliminate non-feasible trajectories and increase the accuracy to which x can be computed. In addition, αx^2 has a practical range of $-50 < \alpha x^2 < (2\pi)^2$ which determines an independent upper bound on x. The x_{MAX} used, then, corresponds to the smaller of the two values.

The transfer time convergence criterion is approximately the same as the granularity of the time input. Since, for some of the problems to be solved, the sensitivity of time to x is so large that the granularity in x, ϵ_x, produces a change in time which exceeds the granularity in time, it is necessary to introduce ϵ_x as a redundant convergence criterion.

The Kepler Subroutine, provided the parameter range constraints are satisfied will always produce a solution.

A negative value of t_D will cause the subroutine to update the state vector backward in time (i.e. backdate the state vector). The subroutine may be called to update or backdate for any amount of time; there are no restrictions on whether the time t_D is less than a period.

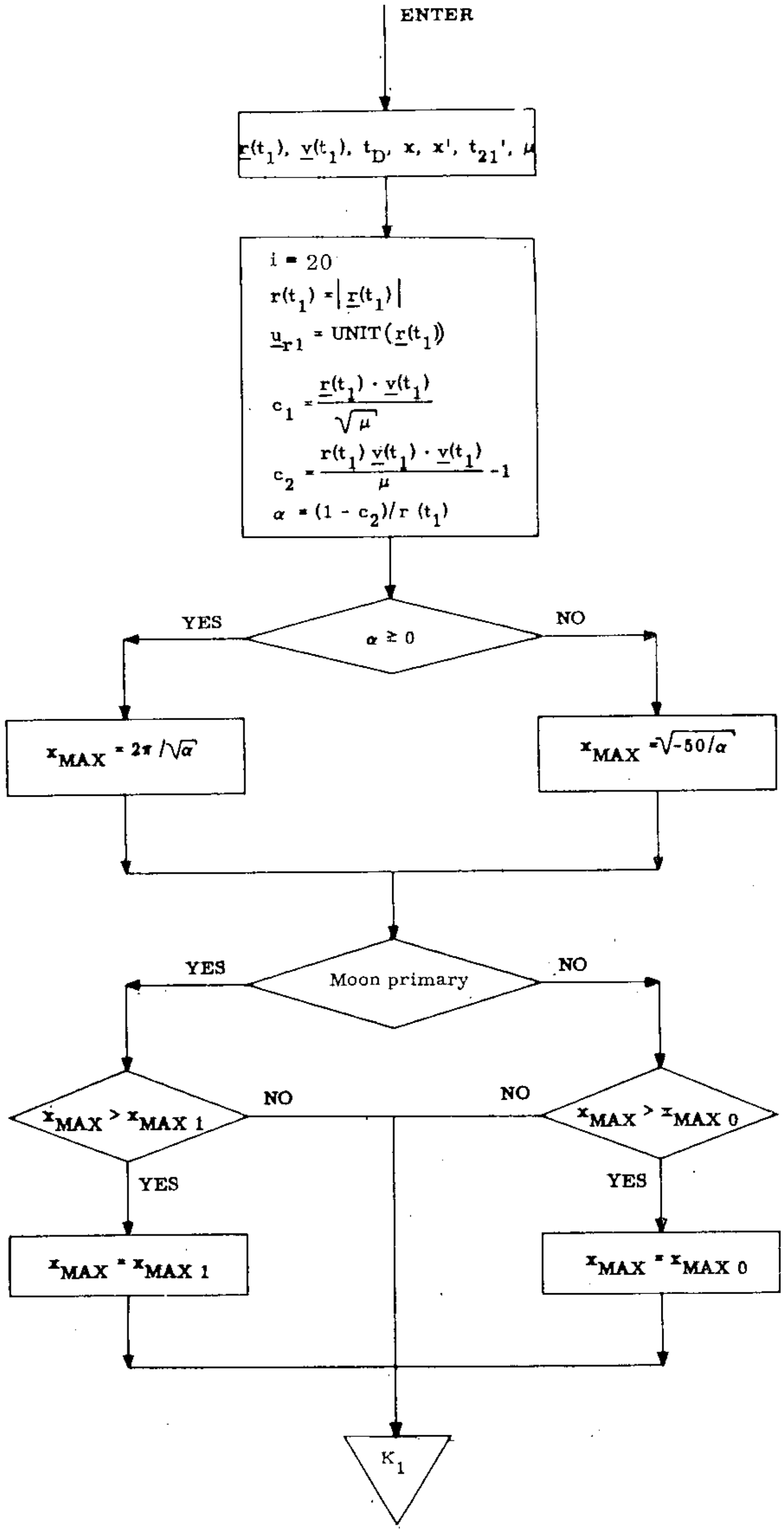


Figure 5. 5-1 Kepler Subroutine

☑ Revised COLOSSUS
☐ Added GSOP # R-577 PCR # 659. 1 Rev. 5 Date 12-5-68

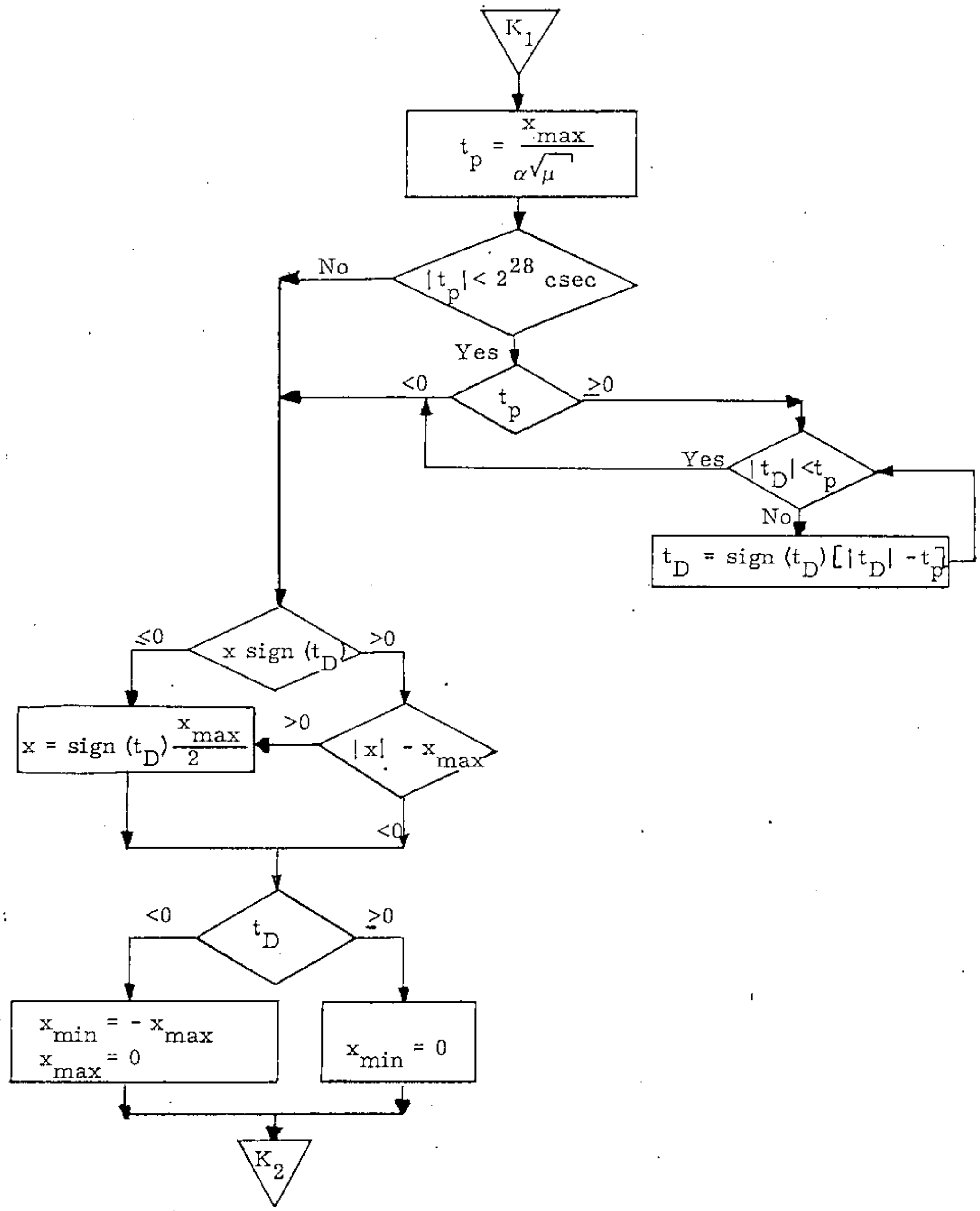


Fig. 5.5-2 Kepler Subroutine

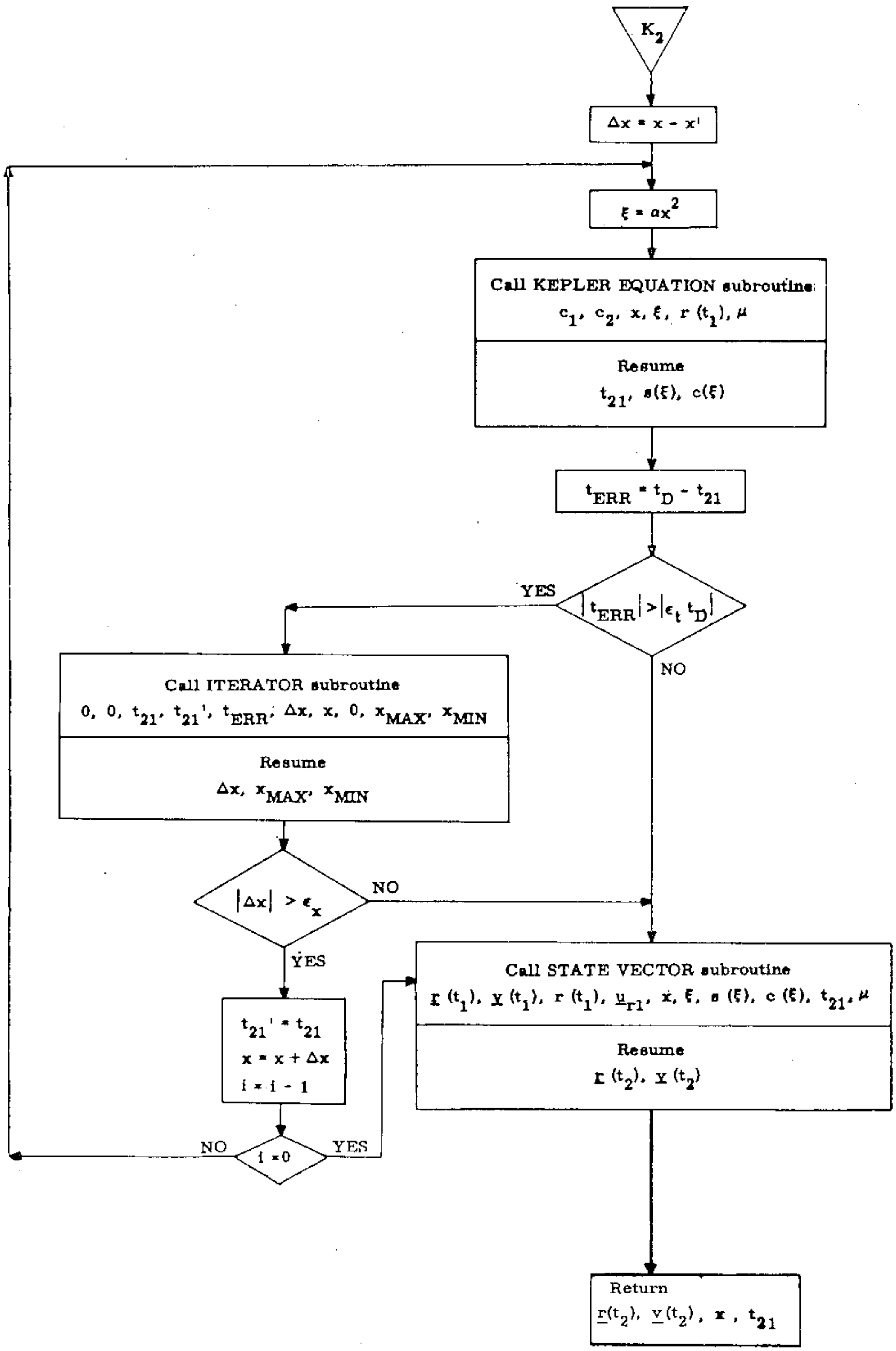


Figure 5.5-3 Kepler Subroutine

5.5.6 LAMBERT SUBROUTINE

The Lambert Subroutine solves for the two body initial velocity vector given the initial and terminal position vectors and a transfer time between the two.

This section contains information to aid the reader in understanding the less obvious aspects of the Lambert Subroutine block diagrams depicted in Figs. 5.6-1 and 5.6-2. The subroutines referred to in these figures are presented in Section 5.5.10 and the nomenclature is found in Section 5.5.1.2.

If the Lambert Subroutine is used repetitively and rapid computation is required, the previous value of the independent variable, cot γ, can be used as a starting point for the new iteration. Flag f_1 provides this option.

The Lambert Subroutine computes the normal to the trajectory, $\underline{u}_N$, using the two input position vectors. If these vectors are nearly colinear, it is desirable to specify the normal as an input rather than rely on the ill-defined normal based on the two input position vectors. Flag f_2 provides this option. The presence of the inputs in parentheses, therefore, is contingent upon the setting of these flags.

The theoretical bounds on the independent variable, cot γ, correspond to the infinite energy hyperbolic path and the parabolic path which closes through infinity. These bounds are dynamically reset by the iterator to provide a more efficient iteration scheme. In addition, if during the course of the iteration, cot γ causes a parameter of the problem to exceed its maximum as determined by its allowable range, the appropriate bound is reset and the iterator continues trying to find an acceptable solution. (This logic does not appear in Figs. 5.6-1 and 2 as it is pertinent only to fixed-point programming). If no acceptable solution is reached, the transfer time input was too small to produce a practical trajectory between the input position vectors. When this happens, $\Delta \cot \gamma$ approaches its granularity limit ϵ_c before time converges to within a fraction ϵ_t of the desired time. However, this same granularity condition exists when the sensitivity problem described in the Kepler Subroutine, Section 5.5.5., occurs. In this case an acceptable solution does exist. This dual situation is resolved via a third convergence criterion. If the error in transfer time is greater than the usual fraction ϵ_t of the desired transfer time, but still less than a slightly larger fraction k_1 of the desired transfer time and $\Delta \cot \gamma$ is less than ϵ_c, then the solution is deemed acceptable and the required velocity is computed.

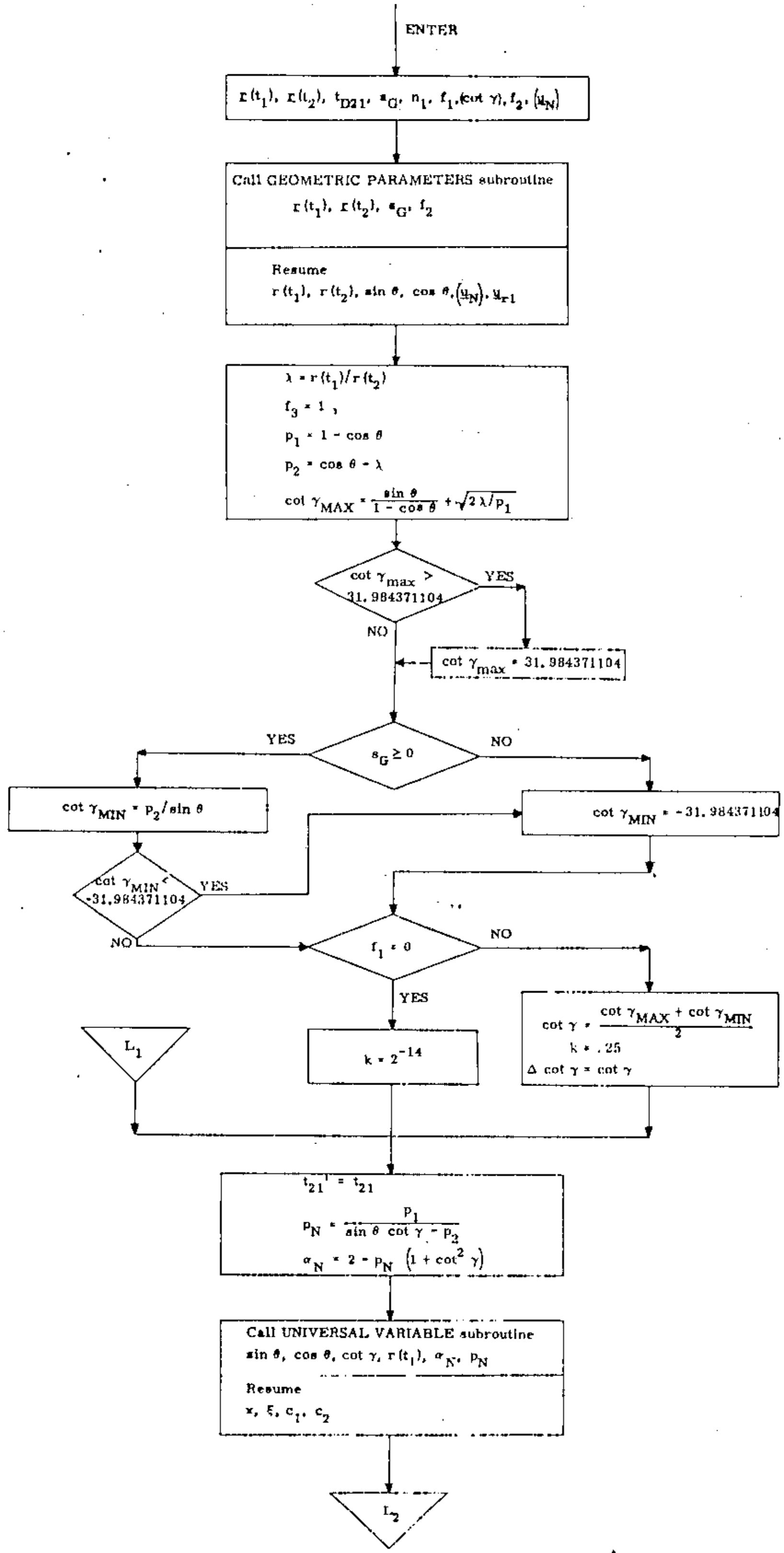
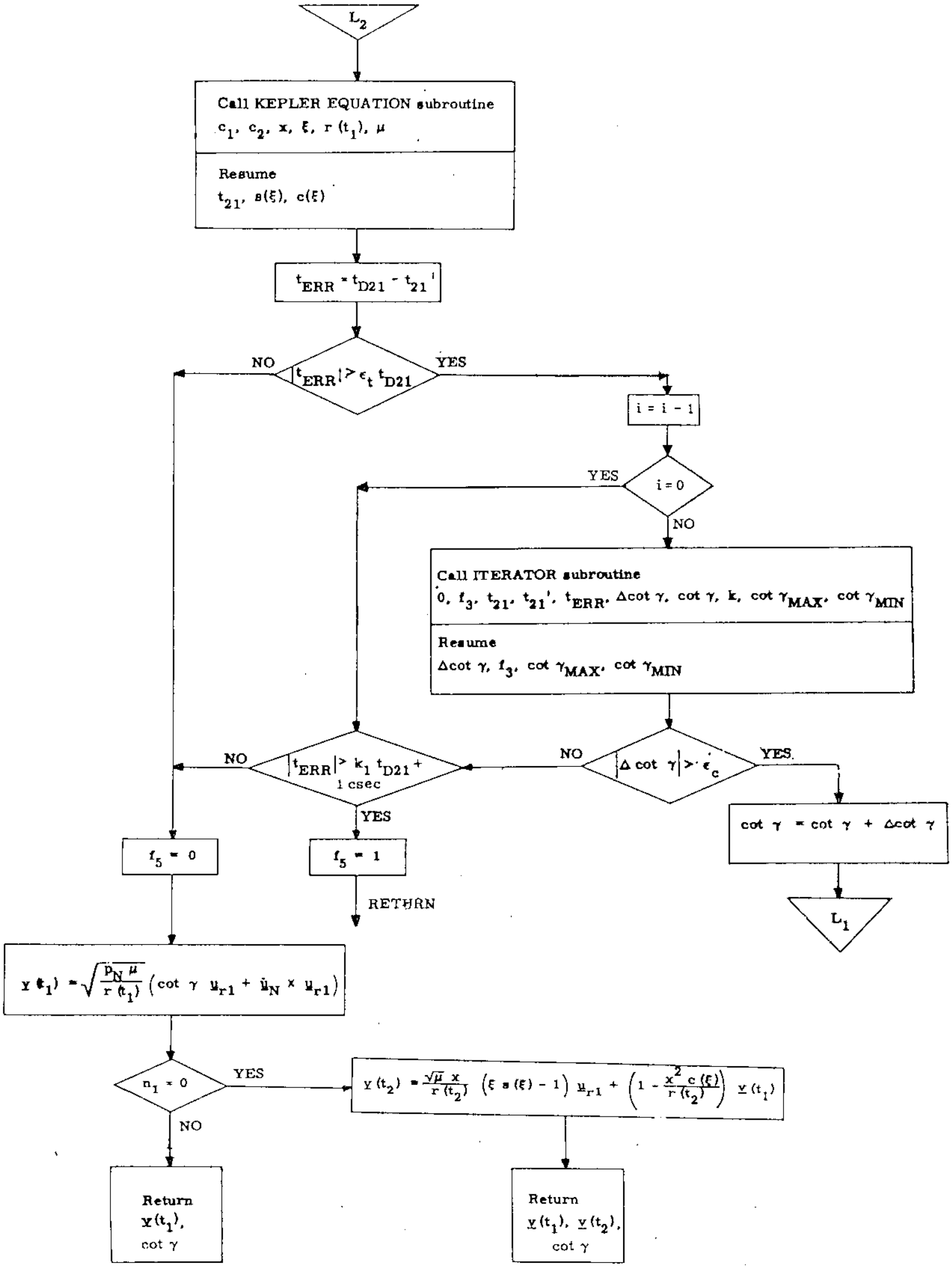


Figure 5.6-1 Lambert Subroutine



$$\underline{v}(t_1) = \sqrt{\frac{p_N \mu}{r(t_1)}} \left(\cot\gamma \ \underline{u}_{r1} + \underline{u}_N \times \underline{u}_{r1} \right)$$

$$\underline{v}(t_2) = \frac{\sqrt{\mu}\, x}{r(t_2)} \left(\xi\, s(\xi) - 1 \right) \underline{u}_{r1} + \left(1 - \frac{x^2 c(\xi)}{r(t_2)} \right) \underline{v}(t_1)$$

Figure 5.6-2 Lambert Subroutine

5.5.7 TIME-THETA SUBROUTINE

The Time-Theta Subroutine solves for the two body transfer time given the initial position and velocity vectors and the true anomaly difference (transfer angle) to the terminal position.

This section contains information to aid the reader in understanding the less obvious aspects of the Time-Theta Subroutine block diagram depicted in Fig. 5.7-1. The subroutines referred to in this figure are presented in Section 5.5.10 and the nomenclature is found in Section 5.5.1.2.

The flag f_6 must be zero if the user desires computation of the terminal state vector in addition to the transfer time.

If the conic trajectory is a parabola or hyperbola and the desired transfer angle, θ, lies beyond the asymptote of the conic, f_7 will be set indicating that no solution is possible.*

In addition to the parameter range constraints imposed on Kepler's equation, the additional restriction on Time-Theta that the trajectory must not be near rectilinear is indicated by the range of cot γ.*

The Time-Theta problem is not well defined for near rectilinear trajectories, i e. the transfer angle θ is no longer a meaningful problem parameter. This will not cause difficulties provided the input variables are within the specified ranges.

---

*If the Time-Theta Routine is called with inputs for which no solution is possible (for either or both of these two reasons), then the routine will abort with an alarm code of 00607.

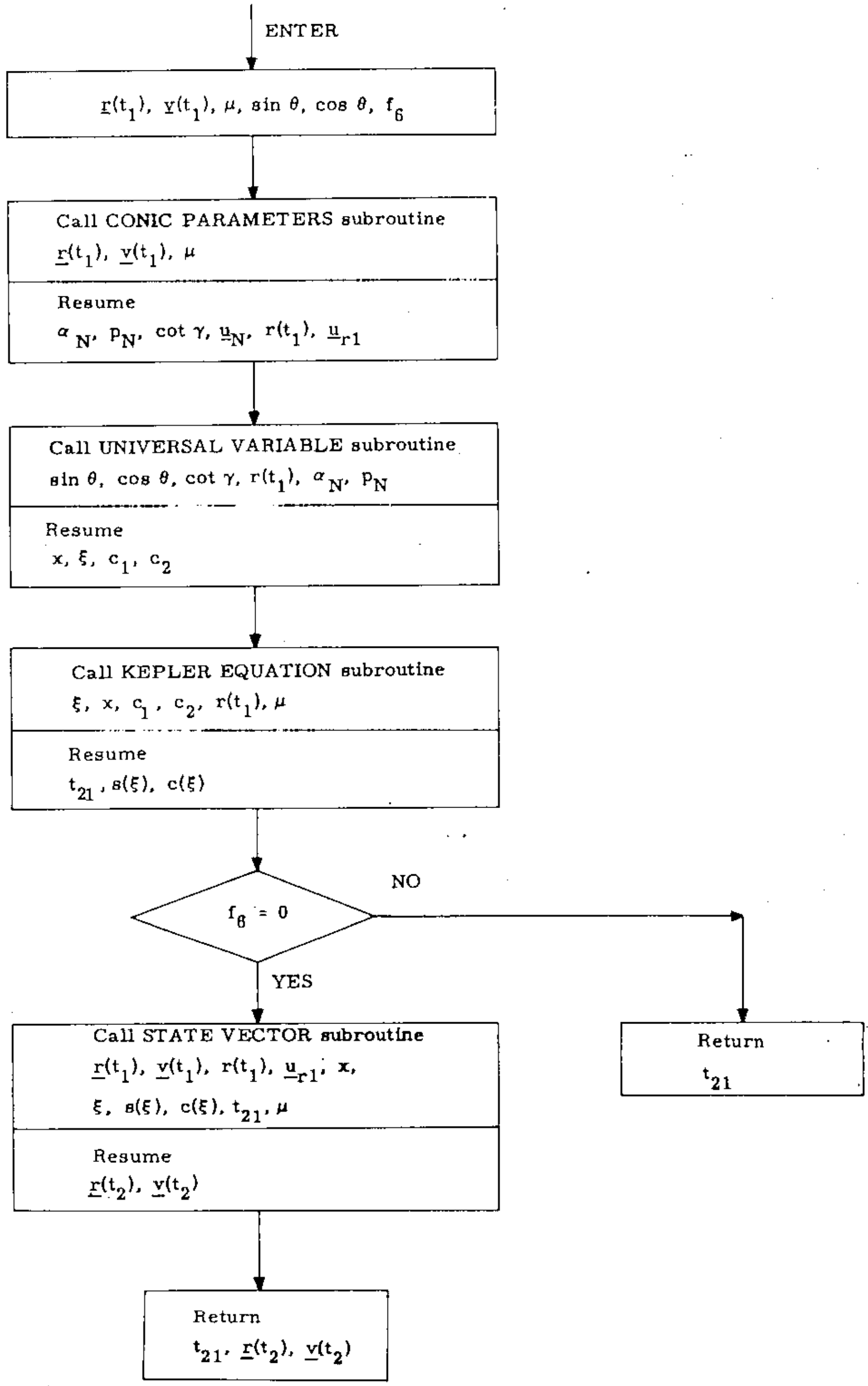


Figure 5.7-1 Time-Theta Subroutine

5.5.8 TIME-RADIUS SUBROUTINE

The Time-Radius Subroutine solves for the two body transfer time to a specified radius given in the initial position and velocity vectors and the radius magnitude.

This section contains information to aid the reader in understanding the less obvious aspects of the Time-Radius Subroutine block diagrams depicted in Figs. 5.8-1 and 5.8-2. The subroutines referred to in this figure are presented in Section 5.5.10 and the nomenclature is found in Section 5.5.1.2.

Paragraphs 3, 4 and 5 of Section 5.5.7 apply to the Time-Radius Subroutine as well.*

Since an inherent singularity is present for the circular orbit case, near-circular orbits result in a loss of accuracy in computing both the transfer time, t_{21}, and the final state vector. This is caused by the increasing sensitivity of t_{21} to $r(t_2)$ as the circular orbit is approached. In the extreme case that eccentricity is less than 2^{-18}, the problem is undefined and the subroutine will exit without a solution, setting flag f_9 to indicate this.*

If $r(t_2)$ is less than the radius of pericenter or greater than the radius of apocenter, then $r(t_2)$ will be ignored and the pericenter or apocenter solution, respectively, will be computed. A flag, f_8, will be set to indicate this.

---

*If the Time-Radius Routine is called with inputs for which no solution is possible (for any one or more of the reasons given in paragraphs 4 or 5 of Section 5.5.7 or paragraph 4 above), the routine will abort with an alarm code of 00607.

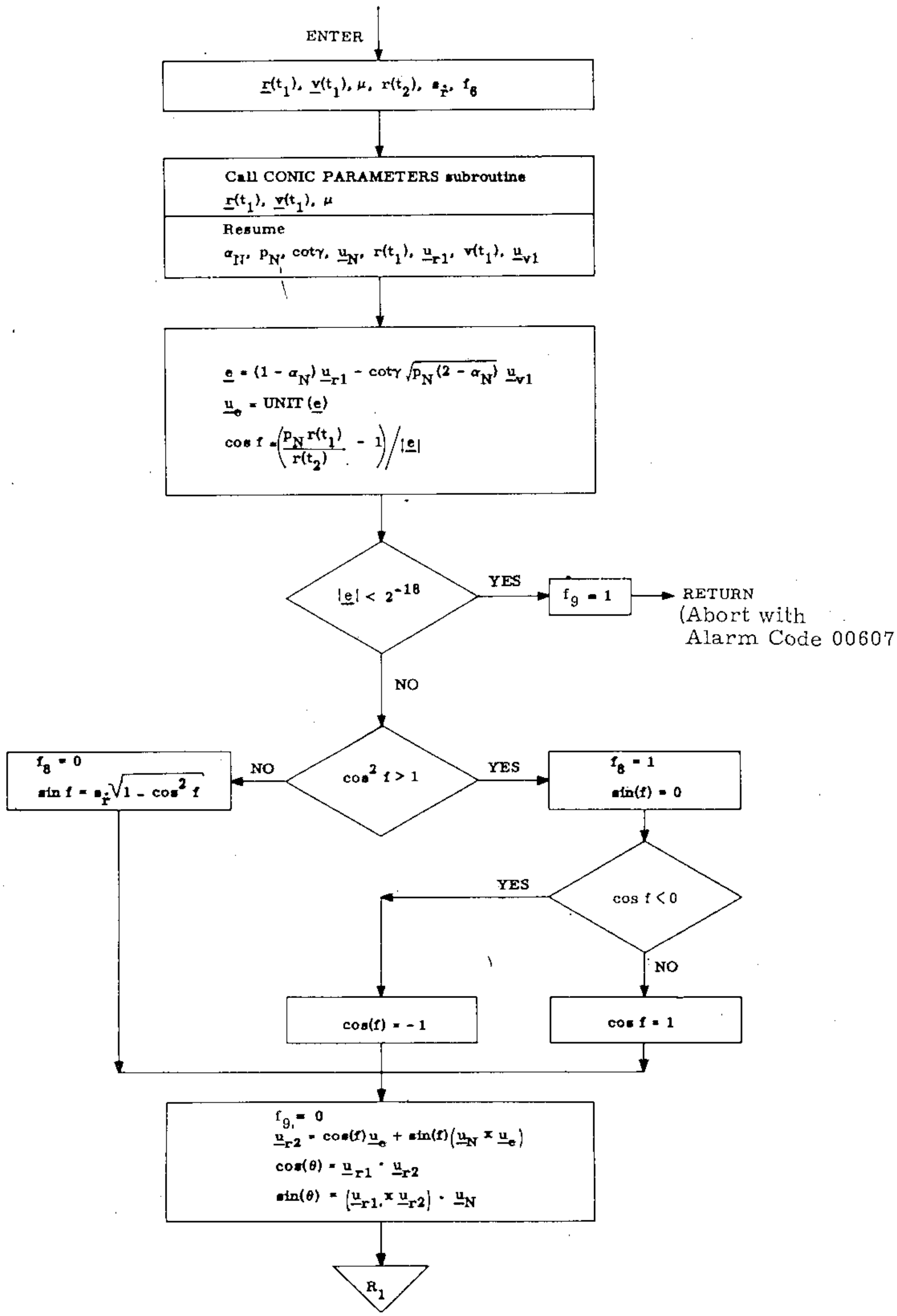


Figure 5.8-1 Time-Radius Subroutine

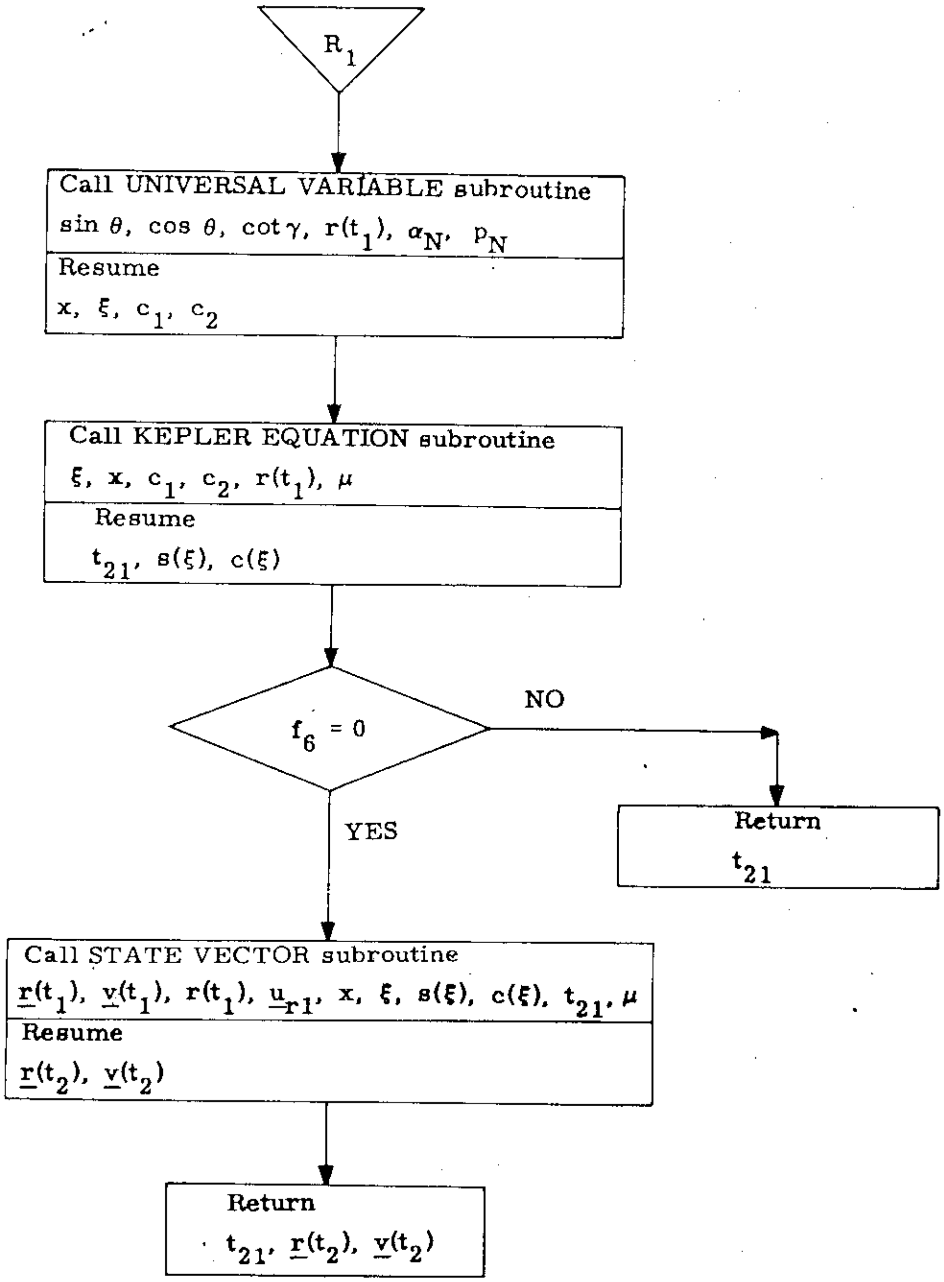


Figure 5.8-2 Time-Radius Subroutine

5.5.9 APSIDES SUBROUTINE

The Apsides Subroutine solves for the two body radii of apocenter and pericenter and the eccentricity of the trajectory given the position and velocity vectors for a point on the trajectory.

This subroutine is depicted in Fig. 5.9-1. The subroutines referred to in this figure are presented in Section 5.5.10. Nomenclature is found in Section 5.5.1.2.

It is characteristic of this computation that the apsides become undefined as the conic approaches a circle. This is manifested by decreasing accuracy. When the conic is nearly parabolic, or hyperbolic, the radius of apocenter is not defined. In this event the radius of apocenter will be set to the maximum positive value allowed by the computer.

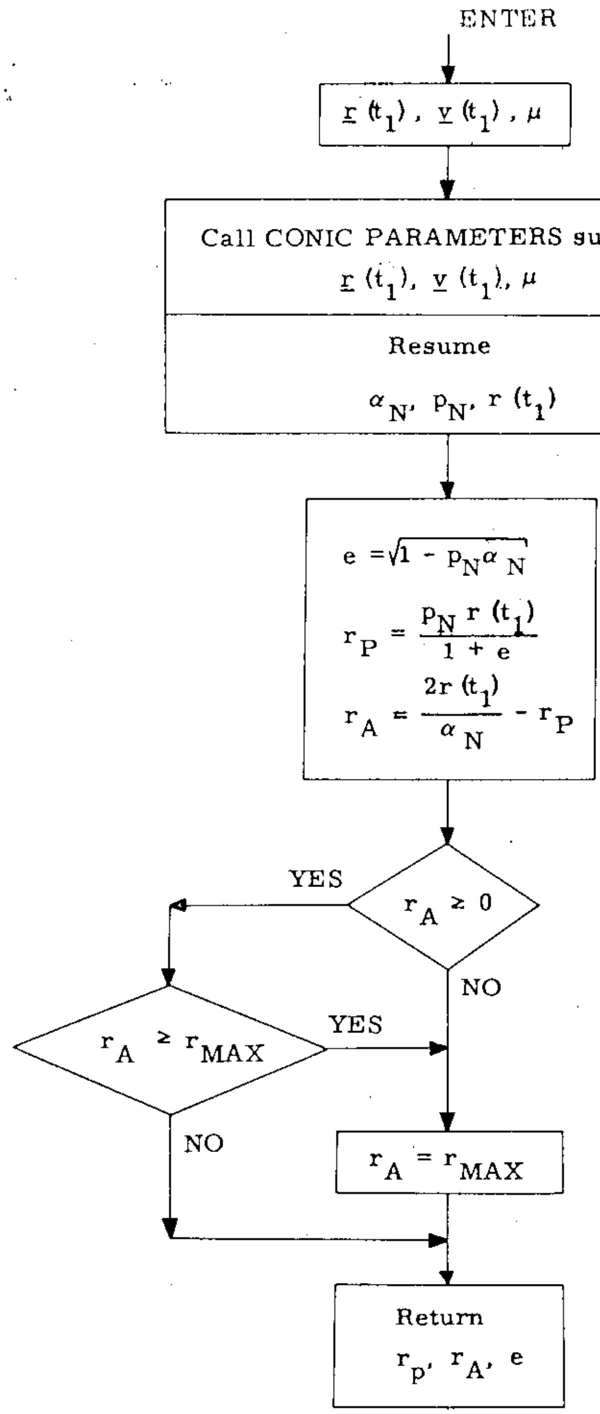


Figure 5.9-1 Apsides Subroutine

5.5.10 MISCELLANEOUS SUBROUTINES

There are, as part of the Conic Trajectory Subroutines, three subroutines which are useful in their own right. These are the Conic Parameters, the Geometric Parameters and the Iterator Subroutines which are depicted in Figs. 5.10-1, 5.10-2 and 5.10-3, respectively.

The Conic Parameters and Geometric Parameters Subroutines are self explanatory.

The Iterator Subroutine serves several purposes. It is used when flag f_4 is set to zero to solve for the value of the independent variable which drives the error in the dependent variable to zero, provided the function is monotonically increasing. To improve convergence for functions whose derivative changes rapidly, the limits are reset as shown in the block diagram.

With f_4 set to 1, the Iterator seeks a minimum of the function, provided the first derivative is single-valued between the limits. The inputs are redefined so that "y" is the derivative of the independent variable with respect to the dependent variable, and "x" is the value at which the derivative was computed or approximated. Since the desired value of y is zero, $y_{ERR} = -y$.

Since the Iterator uses the "Regula Falsi" technique, it requires two sets of variables to begin iteration. If only one set is available, flag f_3 must be set to 1, causing the iterator to generate the independent variable increment from a percentage of the full range.

In addition to the above subroutines there are three other subroutines of primary interest to the five basic conic subroutines described in Sections 5.5.5 to 5.5.9. These are the Universal Variable Subroutine, the Kepler Equation Subroutine and the State Vector Subroutine shown in Figs. 5.10-4, 5.10-5 and 5.10-6, respectively.

The Universal Variable Subroutine is utilized by the Lambert, the Time-Theta and the Time-Radius Subroutines to compute the universal parameter x required for the time equation. There are two different formulations required according to the size of the parameter w.

If the input to the subroutine requires the physically impossible solution that the trajectory "close through infinity", the problem will be aborted, setting flag f_7.

The Kepler Equation Subroutine computes the transfer time given the variable x and the conic parameters.

The State Vector subroutine computes the position and velocity vectors at a point along the trajectory given an initial state vector, the variable x and the transfer time.

The final miscellaneous subroutine, the SETMU Subroutine, is depicted in Fig. 5.10-7. It sets μ to the appropriate primary body gravitational constant consistent with the estimated CSM or LM state vector as defined in Section 5.2.2.6.

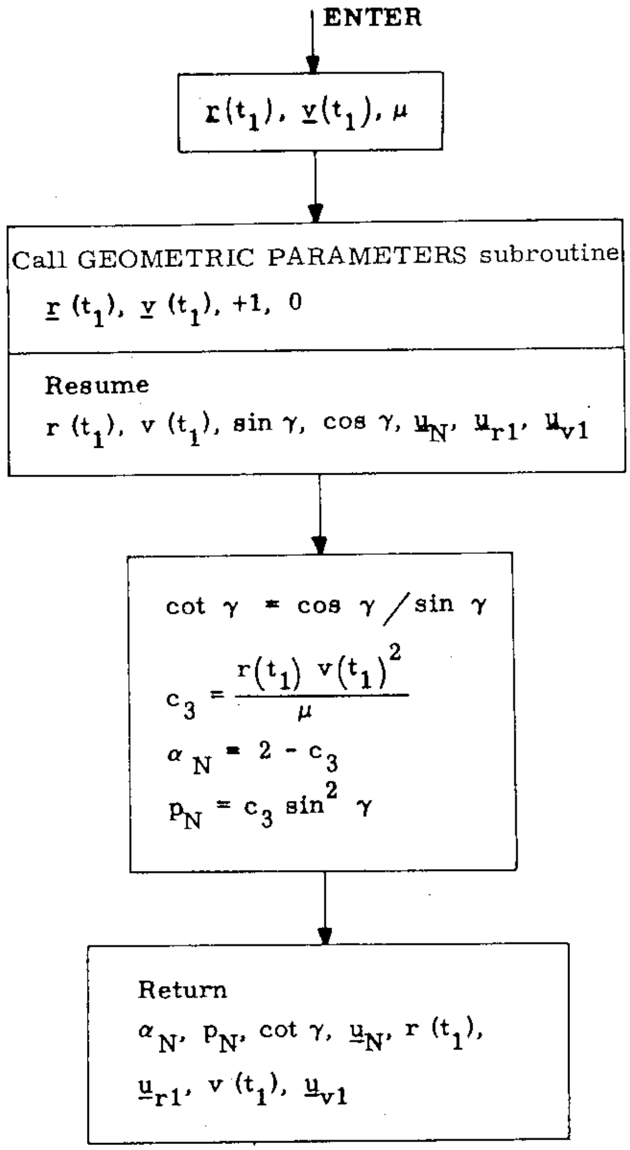


Figure 5.10-1 Conic Parameters Subroutine

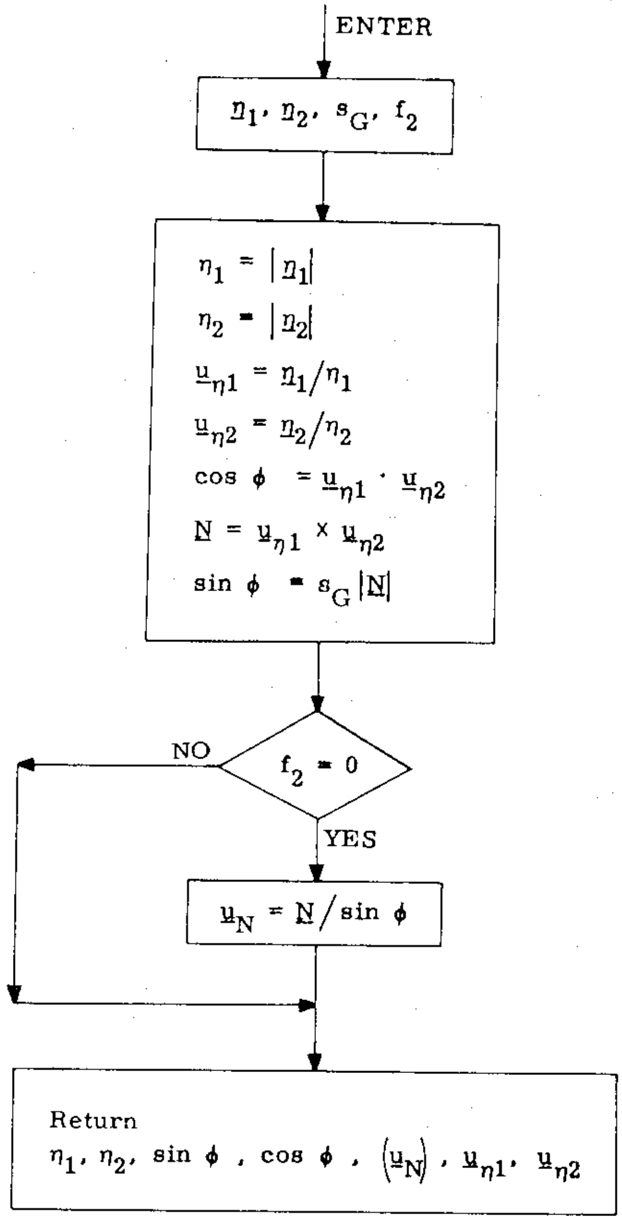


Figure 5.10-2 Geometric Parameters Subroutine

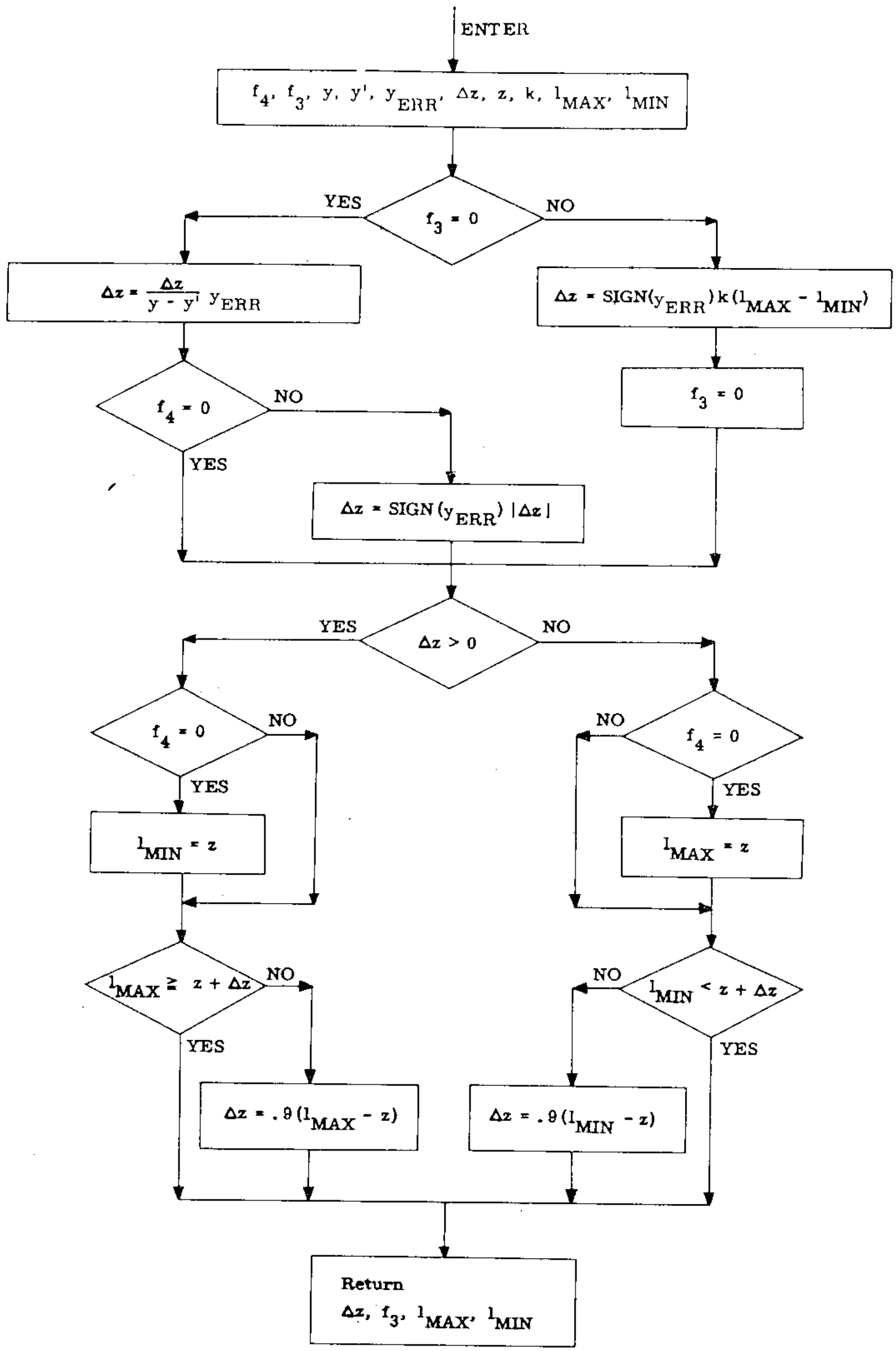


Figure 5.10-3 Iterator Subroutine

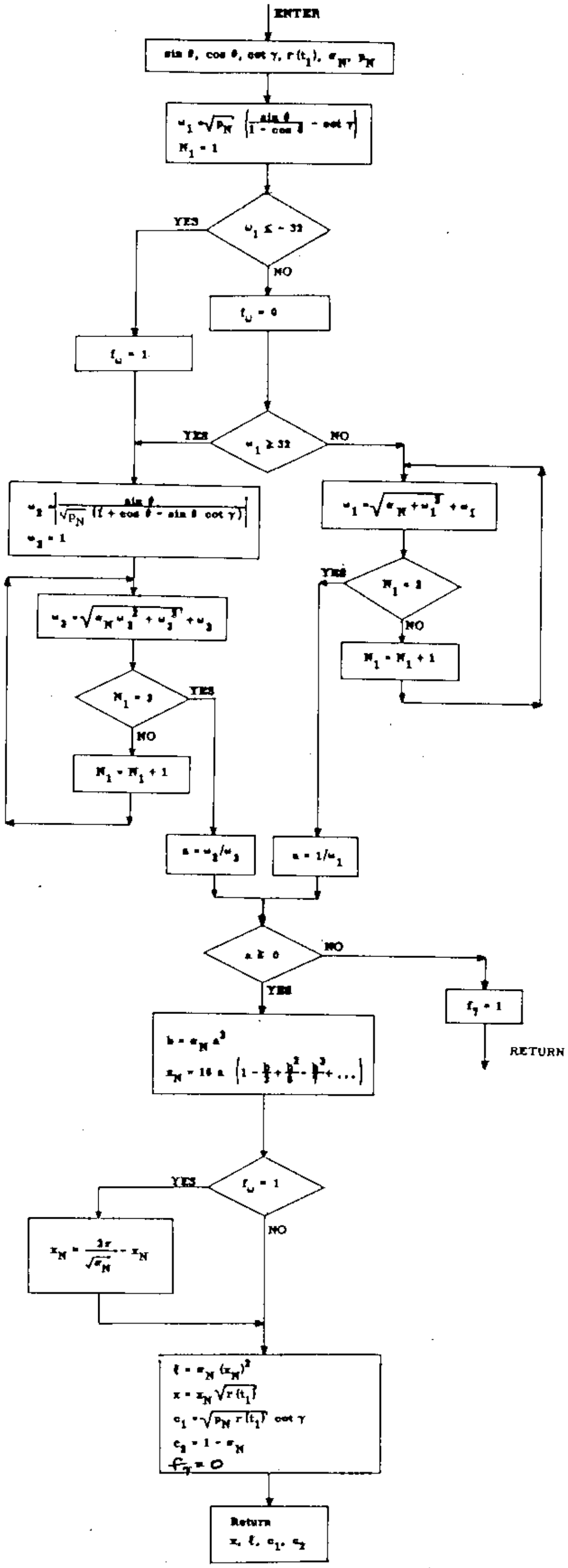


Figure 5.10-4 Universal Variable Subroutine

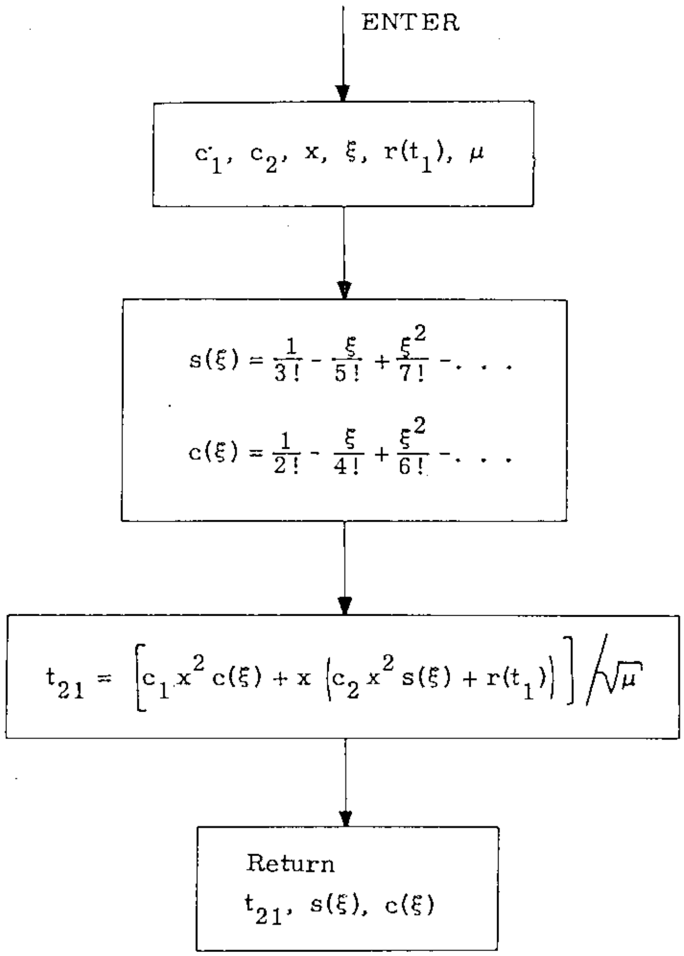


Figure 5.10-5 Kepler Equation Subroutine

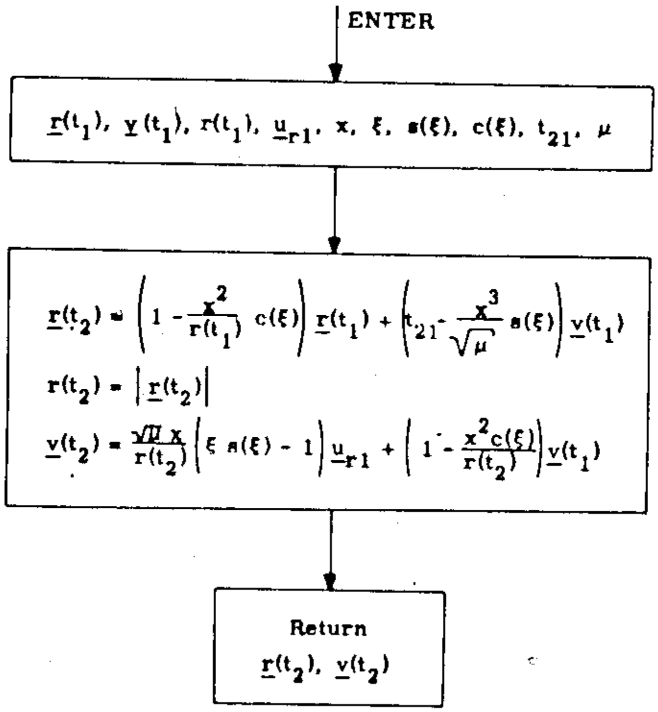


Figure 5.10-6 State Vector Subroutine

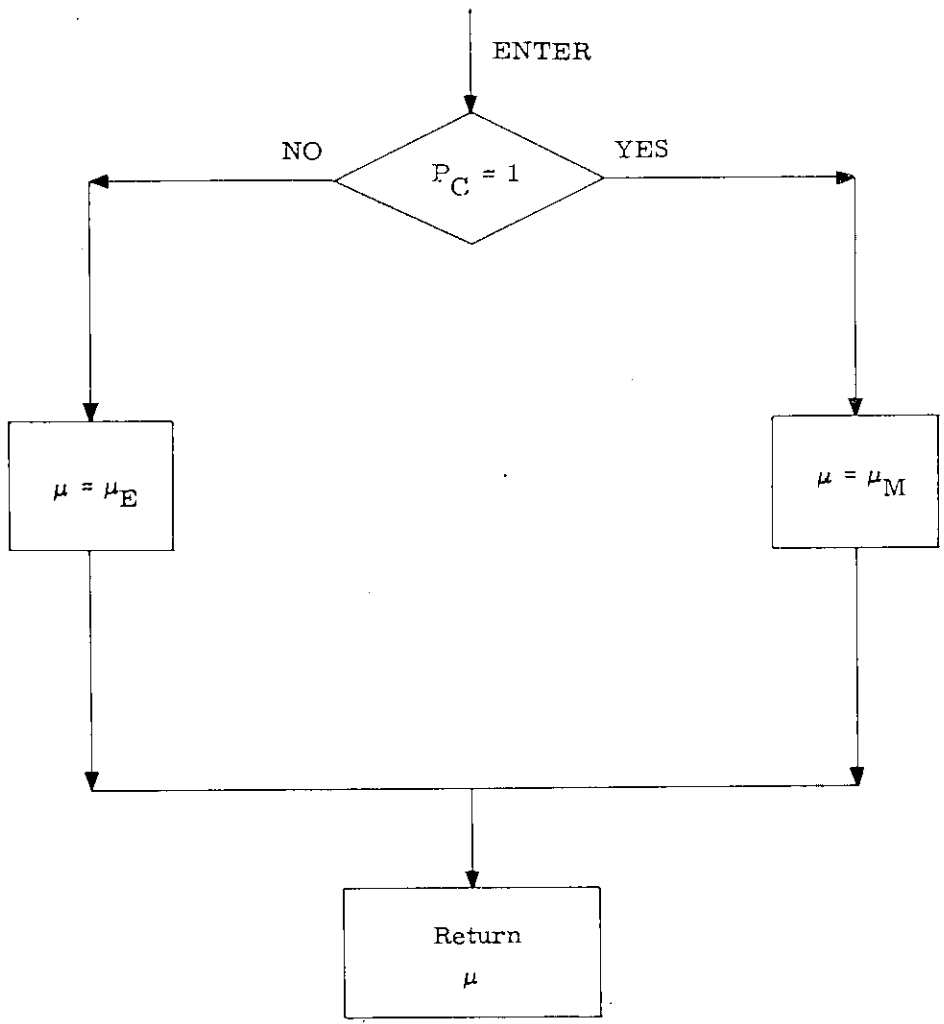


Figure 5.10-7 SETMU Subroutine

5.5.11 INITIAL VELOCITY SUBROUTINE

The Initial Velocity Subroutine computes the required initial velocity vector for a trajectory of specified transfer time between specified initial and target position vectors. The trajectory may be either conic or precision depending on an input parameter (namely, number of offsets). In addition, in the precision trajectory case, the subroutine also computes an "offset target vector", to be used during pure-conic cross-product steering. The offset target vector is the terminal position vector of a conic trajectory which has the same initial state as a precision trajectory whose terminal position vector is the specified target vector.

In order to avoid the inherent singularities in the 180° transfer case when the (true or offset) target vector may be slightly out of the orbital plane, the Initial Velocity Subroutine rotates this vector into a plane defined by the input initial position vector and another input vector (usually the initial velocity vector), whenever the input target vector lies inside a cone whose vertex is the origin of coordinates, whose axis is the 180° transfer direction, and whose cone angle is specified by the user.

The Initial Velocity Subroutine is depicted in Fig. 5.11-1. The Lambert Subroutine, Section 5.5.6, is utilized for the conic computations; and the Coasting Integration Subroutine, Section 5.2.2, is utilized for the precision trajectory computations.

Nomenclature for the Initial Velocity Subroutine

$\underline{r}(t_1)$ — Initial position vector.

$\underline{v}(t_1)$ — Vector (usually the actual initial velocity vector) used to determine whether the transfer from the initial position vector to the target vector is through a central angle of less or greater than 180^o, and also used in certain cases to specify the transfer plane (see text).

$\underline{r}_T(t_2)$ — Target Vector (True target vector if $N_1 > 0$, or Offset target vector if $N_1 = 0$).

t_D — Desired transfer time from initial position vector to target vector.

N_1 — Number of offsets to be used in calculating the offset target vector from the true target vector. ($N_1 = 0$ implies conic calculations only with offset target vector input).

ϵ — Cone Angle of a cone whose vertex is the coordinate origin and whose axis is the 180^o transfer direction (i. e., the negative initial position direction). The cone angle ϵ is measured from the axis to the side of the cone.

f_1 — Switch set to 0 or 1 according to whether a guess of cot γ is input or not.

[cot γ] — Guess of cot γ.

$\underline{v}_T(t_1)$ — Required initial velocity vector of a precision [a conic] trajectory which passes through the true [or offset] target vector, or the rotated true [or offset] target vector if the original target vector was in the cone, at the end of the desired transfer time, if $N_1 > 0$ [or $N_1 = 0$].

$\underline{r}(t_2)$ — Computed offset target vector.

$\underline{v}_T(t_2)$ — Final precision [conic] velocity vector resulting from a precision [conic] update of the initial position vector and the required initial velocity vector $\underline{v}_T(t_1)$, if $N_1 > 0$, [or $N_1 = 0$, respectively].

$\underline{r}_T(t_2)$ — Final precision position vector.

cot γ — Value to which the Lambert Subroutine converged (for later use as guess to minimize computation time).

f_2 — Switch set to 0 or 1 according to whether the input (true or offset) target vector was not or was in the cone, and consequently was not or was rotated into the plane.

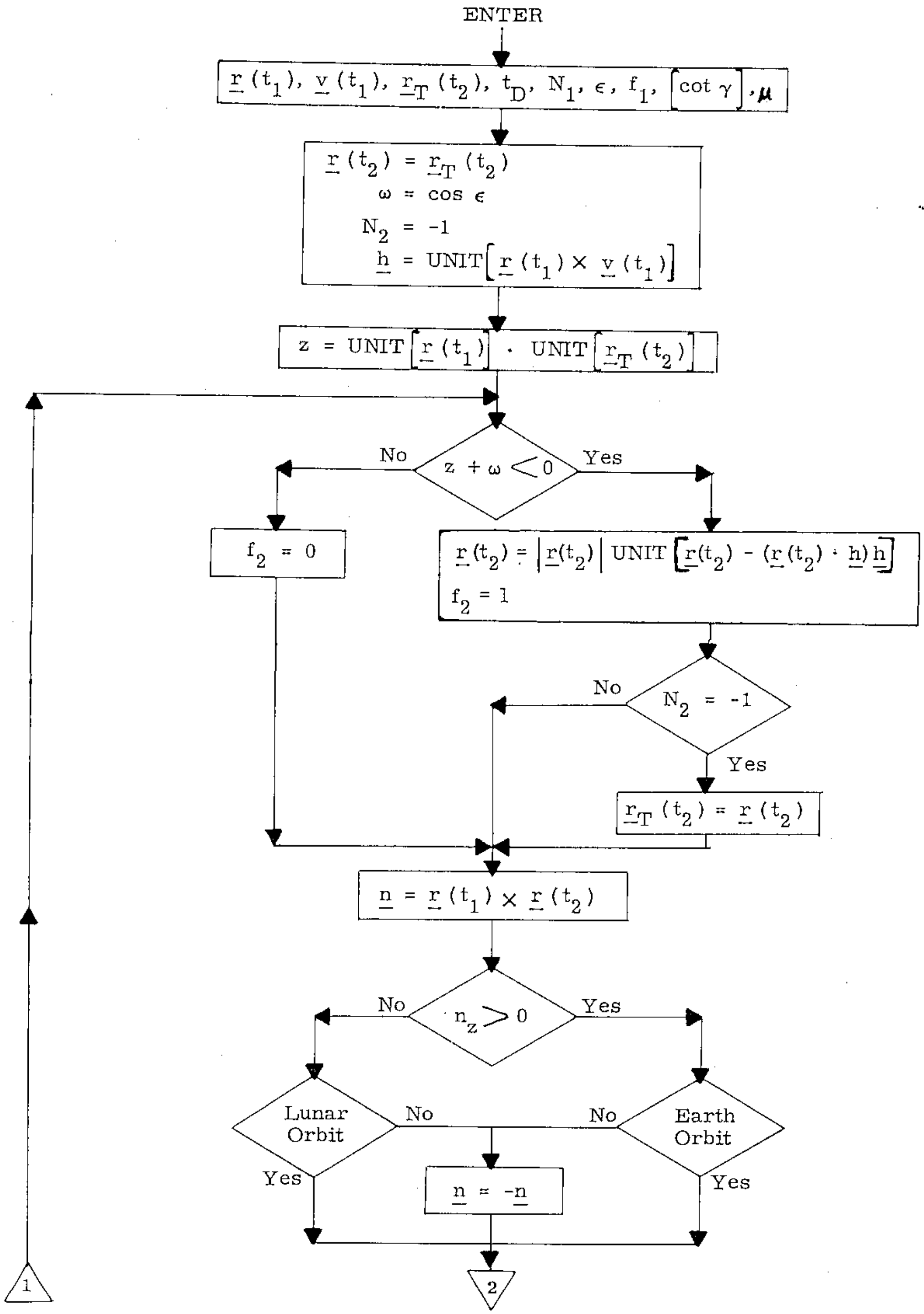


Figure 5.11-1 Initial Velocity Subroutine
(page 1 of 2)

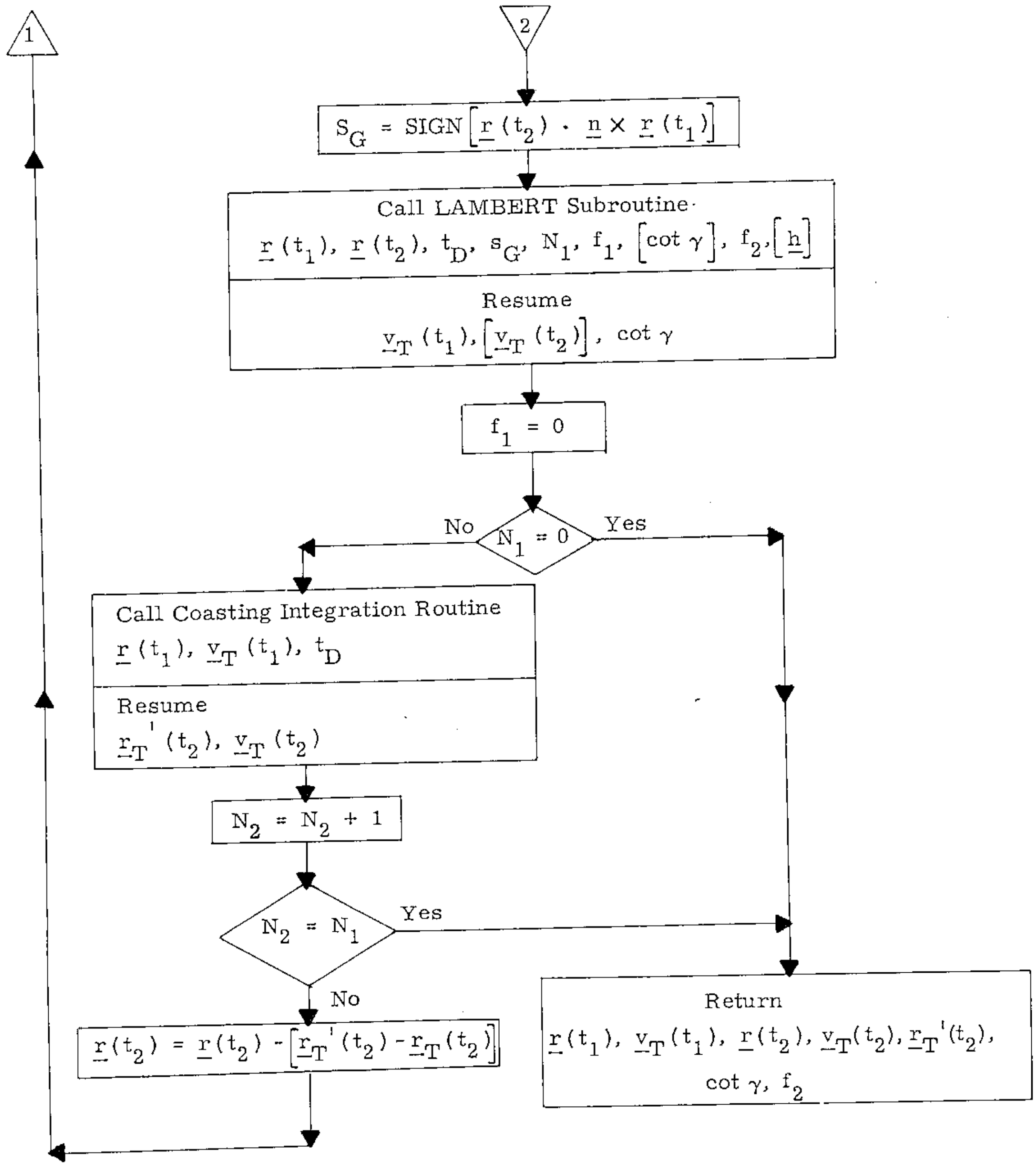


Figure 5.11-1 Initial Velocity Subroutine
(page 2 of 2)

5.5.12 TRANSFER ANGLE SUBROUTINE

The computation of orbital transfer trajectories generally requires a time of flight, t_F. However, for purposes of communication between the astronaut and the guidance computer, ωt, the central angle of travel of the passive vehicle during the transfer is more convenient. This subroutine is used to convert from t_F to ωt. The conversion is approximate because it is based on the mean motion of the passive vehicle.

The equations and logic used for the conversion are shown in Fig. 5.12-1.

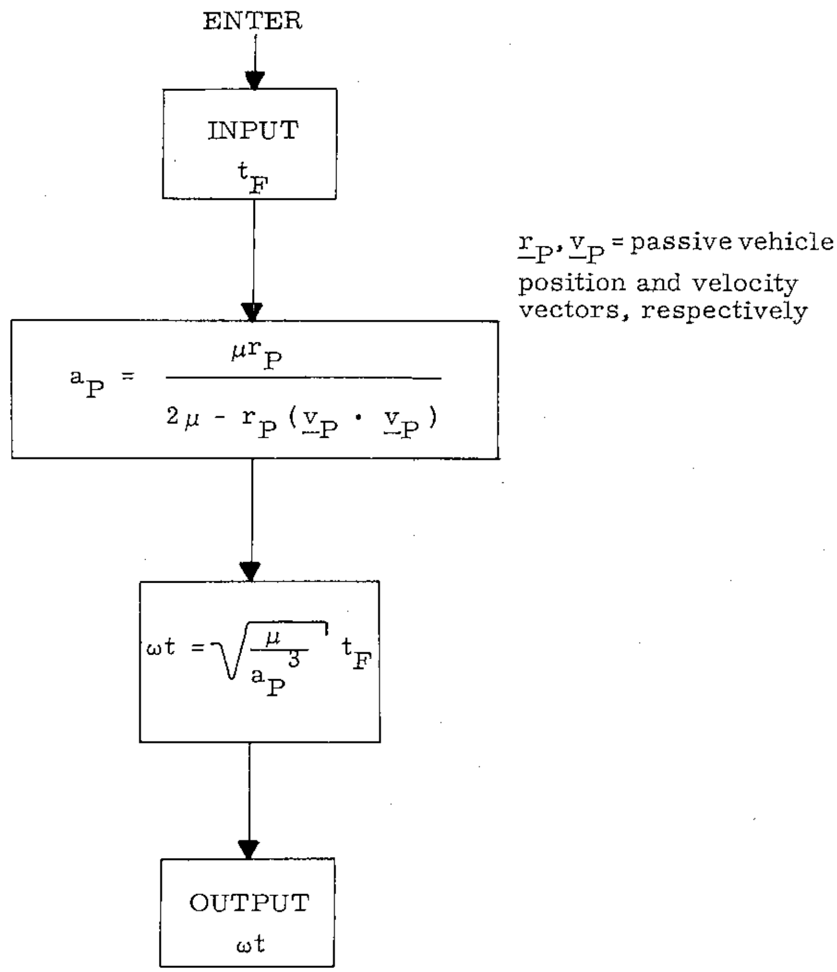


Fig. 5.12-1 Transfer Angle Subroutine

5.5.13 LOSSEM SUBROUTINE

The LOSSEM Subroutine computes the lines-of-sight of the Sun, Earth, and Moon with respect to the spacecraft in the Basic Reference Coordinate System. This data is used by the IMU alignment programs whenever the astronaut elects to sight on the Sun, Earth, or Moon instead of a star for purposes of IMU alignment. The data is also used by the Star Selection Routine (Section 5.6.4) when testing for star occultation. In addition, this subroutine computes the sizes of the occultation cones used in the Star Selection Routine and the correction for aberration of light which is applied in the IMU alignment programs to the line-of-sight unit vector of a star stored in basic reference coordinates.

The unit vectors $\underline{u}_S$, $\underline{u}_E$, and $\underline{u}_M$ specifying the lines-of-sight to the Sun, Earth, and Moon respectively, in the Basic Reference Coordinate System are computed as follows:

$$\underline{u}_S = \begin{cases} \text{UNIT}\ (\underline{r}_{ES}) & \text{if } P = E \\ \text{UNIT}\ (\underline{r}_{ES} - \underline{r}_{EM}) & \text{if } P = M \end{cases} \tag{5.13.1}$$

$$\underline{u}_E = \begin{cases} -\text{UNIT}\ (\underline{r}_C) & \text{if } P = E \\ -\text{UNIT}\ (\underline{r}_{EM} + \underline{r}_C) & \text{if } P = M \end{cases} \tag{5.13.2}$$

$$\underline{u}_M = \begin{cases} \text{UNIT}\ (\underline{r}_{EM} - \underline{r}_C) & \text{if } P = E \\ -\text{UNIT}\ (\underline{r}_C) & \text{if } P = M \end{cases} \tag{5.13.3}$$

where P, E, M, and S respectively denote the primary body, Earth, Moon, and Sun, $\underline{r}_C$ is the position vector of the CSM with respect to the primary body, and $\underline{r}_{EM}$ and $\underline{r}_{ES}$ are the position vectors of the Moon and Sun with respect to the Earth obtained from the Lunar and Solar Ephemerides Subroutine of Section 5.5.4. The line-of-sight vectors are determined for a time specified by the calling program or routine.

The occultation cones used in the Star Selection Routine for the Sun, Earth, and Moon are computed as follows:

$$c_S = \cos 15^\circ \tag{5.13.4}$$

$$c_E = \begin{cases} \cos\left[5^\circ + \sin^{-1}\left(\dfrac{R_E}{r_C}\right)\right] & \text{if } P = E \\ \cos 5^\circ & \text{if } P = M \end{cases} \tag{5.13.5}$$

$$c_M = \begin{cases} \cos 5^\circ & \text{if } P = E \\ \cos\left[5^\circ + \sin^{-1}\left(\dfrac{R_M}{r_C}\right)\right] & \text{if } P = M \end{cases} \tag{5.13.6}$$

where c is the cosine of one half the total angular dimension of a cone and represents a more convenient way of treating the dimension of a cone in the Star Selection Routine, r_C is the magnitude of the CSM position vector, R_E is the equatorial radius (6378.166 km) of the Earth, and R_M is the mean radius (1738.09 km) of the Moon.

The vector $\underline{a}$ which is used by the IMU alignment programs to correct the stored star vectors for aberration of light is determined as follows:

$$\underline{a} = \frac{\underline{v}_C - \underline{v}_{ES}}{c} \tag{5.13.7}$$

where $\underline{v}_C$ is the velocity of the CSM with respect to the primary body, $\underline{v}_{ES}$ is the velocity of the Sun relative to the Earth, and c is the speed of light. Note that this correction does not consider the velocity of the Moon relative to the Earth when the primary body is the Moon since the contribution from this source is considered negligible.

5.5.14 PERICENTER-APOCENTER (PERIAPO) SUBROUTINE

The Pericenter - Apocenter Subroutine computes the two body apocenter and pericenter altitudes given the position and velocity vectors for a point on the trajectory and the primary body.

This subroutine is depicted in Fig. 5.14-1.

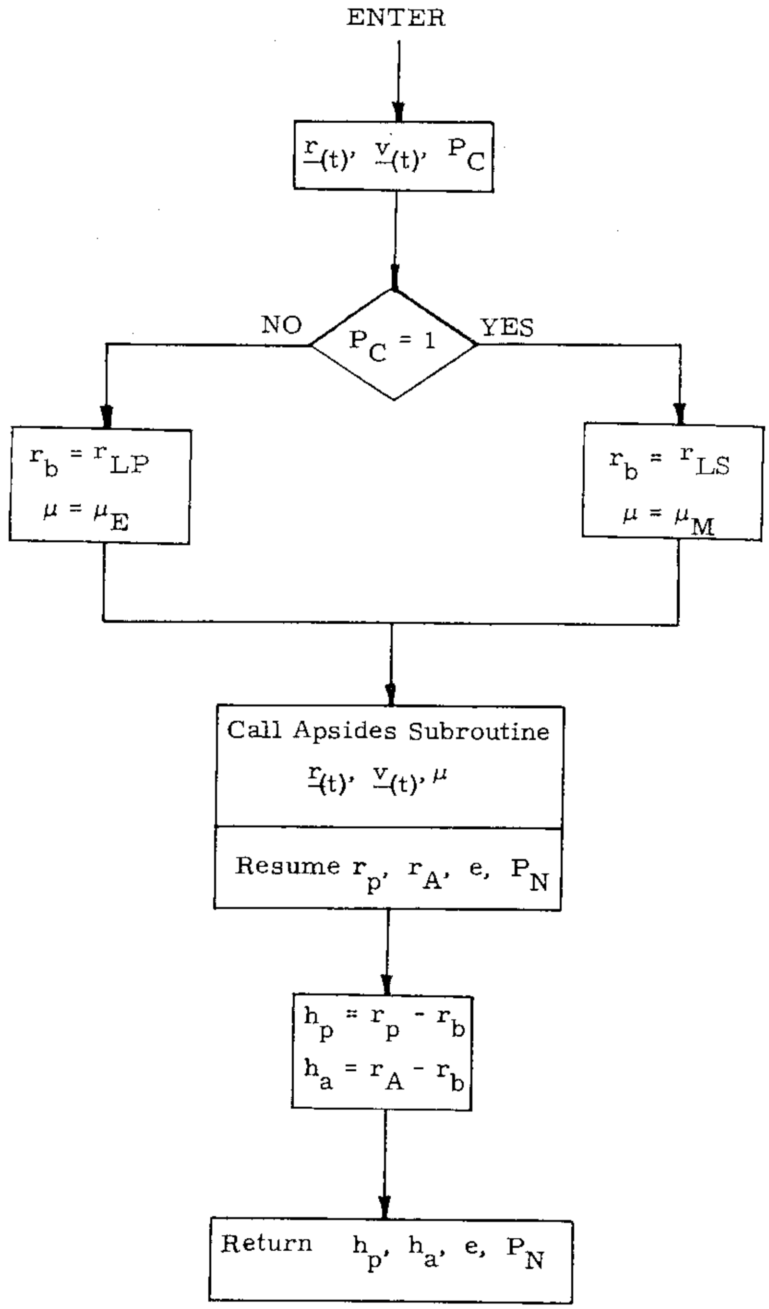


Figure 5.14-1 PERICENTER - APOCENTER SUBROUTINE