NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Technical Memorandum 33-558

A General Purpose Maneuver Turns Computer Program

G. I. Jaivin

JET PROPULSION LABORATORY
CALIFORNIA INSTITUTE OF TECHNOLOGY
PASADENA, CALIFORNIA

August 15, 1972

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Technical Memorandum 33-558

A General Purpose Maneuver Turns Computer Program

G. I. Jaivin

JET PROPULSION LABORATORY
CALIFORNIA INSTITUTE OF TECHNOLOGY
PASADENA, CALIFORNIA

August 15, 1972

PREFACE

The work described in this report was performed by the Guidance and Control Division of the Jet Propulsion Laboratory.

CONTENTS

I. Introduction 1

II. Problem Task Description 2

A. Definition of Coordinate Systems 2

B. Program Objectives and Procedures 3

III. Method of Solution 4

IV. Program Description 10

A. Introduction 10

B. Basic Program Description 10

C. Detailed Output Option 11

D. Specific Turn Option 11

V. Operating Instructions 12

A. Input 12

B. Output 12

Appendix: Subroutine Descriptions and Program Listings 23

TABLES

1. Input symbols 13

2. Definition of integer input parameter TURN 14

3. Definition of integer input parameter TURN1 14

4. Definition of integer input parameter TURN2 15

FIGURES

1. Definition of $(\overline{A}, \overline{B}, \overline{C})$ coordinate system 16

2. Definition of arbitrary vector in terms of clock and cone angles 16

3. Typical relationship between the $(\overline{X}, \overline{Y}, \overline{Z})$ and $(\overline{A}', \overline{B}', \overline{C}')$ coordinate systems. 17

4. General purpose maneuver turns program functional flow diagram. 18

5. Typical program output. 19

6. Program output illustrating the detailed solution option 20

7. Program output illustrating the specified turns option (coordinate transformation calculation) 21

8. Program output illustrating the specified turns option (vector transformation calculation) 22

I. INTRODUCTION

During the course of a spaceflight mission the spacecraft orientation is nominally stabilized by orienting on-board sensors to specific celestial references. At certain times during the mission the spacecraft attitude must be changed to point a spacecraft-fixed piece of hardware in a specified inertially-fixed direction. Examples where such a spacecraft reorientation would be required include the pointing of a rocket motor prior to a trajectory correction maneuver, and the repositioning of a radio antenna toward the Earth to improve the signal strength. This reorientation is accomplished by turning the spacecraft about one or more of its body axes. The choice of turn axes is usually restricted by the capabilities of the on-board attitude control system. Thus, given these requirements and constraints, it becomes necessary to calculate the magnitudes of those turns or combination of turns that correctly achieve the desired realignment of the spacecraft. A computer program has been written to compute the desired maneuver parameters, and a description of what the program is designed to do and how it may be used are contained in the following sections of this report.

II. PROBLEM TASK DESCRIPTION

A. Definition of Coordinate Systems

It becomes necessary at this point to define the orientation of the spacecraft with respect to inertial space. It is assumed that, when the spacecraft is in the cruise mode prior to a maneuver, the attitude control system has acquired its primary and secondary celestial attitude references. The Sun and the star Canopus have often been chosen to serve as these references. However, to retain generality in this discussion no further mention of specific references will be made. A primary reference-probe-secondary reference spacecraft-centered coordinate system ($\overline{A}$, $\overline{B}$, $\overline{C}$) is defined such that $\overline{C}$ is directed along the probe-primary reference direction; $\overline{B} = \overline{C} \times \overline{S}$, where $\overline{S}$ is directed along the probe-secondary reference direction; and $\overline{A} = \overline{B} \times \overline{C}$ completes the right-handed orthogonal coordinate system. This system is shown in Fig. 1. A spacecraft-fixed coordinate system ($\overline{A}'$, $\overline{B}'$, $\overline{C}'$) is defined such that when the spacecraft primary and secondary reference sensors are locked onto their respective celestial objects the ($\overline{A}$, $\overline{B}$, $\overline{C}$) and the ($\overline{A}'$, $\overline{B}'$, $\overline{C}'$) coordinate systems are coincident.

The cone angle of a celestial object is defined as the angle, β (where $0 \leq \beta \leq 180$ deg), from the spacecraft primary reference line to the spacecraft-object line. The clock angle of a celestial object is defined as an angle, α (where $0 \leq \alpha < 360$ deg), between a plane containing the primary reference, the spacecraft, and the secondary reference, and a plane containing the primary reference, the spacecraft, and the object. The clock angle is measured from the primary reference-spacecraft-secondary reference plane and is defined as positive, in the clockwise direction, when looking toward the primary reference from the spacecraft. Figure 2 illustrates how these definitions can be applied to the specification of any arbitrary spacecraft-fixed vector $\overline{V}$.

The spacecraft body axis (pitch, yaw, and roll) coordinate system ($\overline{X}$, $\overline{Y}$, $\overline{Z}$) is defined by expressing the pitch, ($\overline{X}$), and roll, ($\overline{Z}$), axes in terms of clock and cone angles with respect to the ($\overline{A}'$, $\overline{B}'$, $\overline{C}'$) system. The yaw axis, ($\overline{Y}$), defined to be $\overline{Y} = \overline{Z} \times \overline{X}$, completes the definition of this right-handed orthogonal coordinate system. Figure 3 shows such a coordinate system and its relationship to the ($\overline{A}'$, $\overline{B}'$, $\overline{C}'$) system.

B. Program Objectives and Procedures

The major objective of this program is to determine the magnitudes of the maneuver turns required to point a spacecraft-fixed vector $\overline{VT}$ in the direction of an inertially-fixed vector $\overline{VC}$. These vectors are defined by expressing them in terms of clock and cone angles with respect to the ($\overline{A}$, $\overline{B}$, $\overline{C}$) coordinate system. Because the spacecraft is assumed to be in the cruise mode the ($\overline{A}$, $\overline{B}$, $\overline{C}$) and ($\overline{A}'$, $\overline{B}'$, $\overline{C}'$) coordinate axes are coincident. The transformation matrix from the ($\overline{A}'$, $\overline{B}'$, $\overline{C}'$) system to the ($\overline{X}$, $\overline{Y}$, $\overline{Z}$) system is computed and vectors $\overline{VC}$ and $\overline{VT}$ are redefined in terms of the ($\overline{X}$, $\overline{Y}$, $\overline{Z}$) system by the program. The desired two-turn sequence is selected and the required turns needed to point the vector $\overline{VT}$ in the direction of $\overline{VC}$ are computed. In the event that a two-turn sequence cannot achieve the desired pointing a three-turn sequence can be computed if desired. It is also possible to eliminate the computation of a two-turn maneuver and compute only a three-turn sequence if desired. The turn magnitude of the first maneuver of the three-turn sequence is chosen by the user and the second and third maneuver turn angles are computed by the program. The first-turn magnitude is then incremented by the program and the computation is repeated, thus generating a solution grid as a function of the first-turn angle.

The secondary objective of this program is to determine, as a function of the maneuver turns, the coordinates of an inertially-fixed arbitrarily-chosen reference vector $\overline{RV}$. The vector $\overline{RV}$ is defined by expressing it in terms of clock and cone angles with respect to the ($\overline{A}'$, $\overline{B}'$, $\overline{C}'$) coordinate system, and is computed before and after each maneuver turn. Intermediate positions of the vector $\overline{RV}$ throughout each turn can also be computed, if desired, by means of a program option. Moreover, another program option causes the orientation of this vector to be computed before and after each turn of an arbitrarily-specified two- or three-turn maneuver.

At the conclusion of each computation a new set of vectors, $\overline{VC}$, $\overline{VT}$, and $\overline{RV}$, as well as a new set of maneuver turns, can be defined and the computations may be repeated.

III. METHOD OF SOLUTION

To compute the turn magnitudes required to achieve the required alignment of the spacecraft-fixed vector $\overline{VT}$ with the inertially-fixed vector $\overline{VC}$ the following procedure can be followed.

The desired turn sequence is

$$\overline{VT} = \circledR_2 \circledR_1 \overline{VC} \tag{1}$$

where $\circledR_1$ and $\circledR_2$ represent the first- and second-turn transformation matrices respectively. By performing the indicated matrix multiplication and making the appropriate substitutions it is possible to solve for the turn magnitudes that satisfy the desired spacecraft reorientation.

To illustrate this procedure, a specific maneuver, namely, a roll-yaw turn set, will be studied. The extension of this example to any other maneuver turn set should become obvious. Let

$$\overline{VC} = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \quad \text{and} \quad \overline{VT} = \begin{bmatrix} u \\ v \\ w \end{bmatrix}$$

in the $(\bar{X}, \bar{Y}, \bar{Z})$ coordinate system. The desired roll-yaw sequence is given by

$$\overline{VT} = \circledR_y \circledR_r \overline{VC} \tag{2}$$

where $\circledR_y$ is the yaw turn transformation matrix and is given by

$$\circledR_y = \begin{bmatrix} \cos\theta_y & 0 & -\sin\theta_y \\ 0 & 1 & 0 \\ \sin\theta_y & 0 & \cos\theta_y \end{bmatrix} \tag{3}$$

and Θ_r is the roll turn transformation matrix and is given by

$$\Theta_r = \begin{bmatrix} \cos\theta_r & \sin\theta_r & 0 \\ -\sin\theta_r & \cos\theta_r & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4}$$

Note that θ_y and θ_r are the yaw and roll turn magnitudes respectively. Substituting Eqs. (3) and (4) into Eq. (2) and performing the matrix multiplication yields

$$u = a\cos\theta_r\cos\theta_y + b\sin\theta_r\cos\theta_y - c\sin\theta_y \tag{5}$$

$$v = -a\sin\theta_r + b\cos\theta_r \tag{6}$$

$$w = a\cos\theta_r\sin\theta_y + b\sin\theta_r\sin\theta_y + c\cos\theta_y \tag{7}$$

Solving for $\sin\theta_r$ from Eq. (6) gives

$$\sin\theta_r = \frac{-va \pm b\sqrt{a^2 + b^2 - v^2}}{a^2 + b^2} \tag{8}$$

Inserting Eq. (8) into Eq. (6) and solving for $\cos\theta_r$ yields

$$\cos\theta_r = \frac{vb \pm a\sqrt{a^2 + b^2 - v^2}}{a^2 + b^2} \tag{9}$$

Therefore, using Eqs. (8) and (9)

$$\tan\theta_r = \frac{-va \pm b\sqrt{a^2 + b^2 - v^2}}{vb \pm a\sqrt{a^2 + b^2 - v^2}} \tag{10}$$

Equations (5) and (7) can be rewritten as

$$u = Z \cos \theta_y - c \sin \theta_y \tag{11}$$

$$w = c \cos \theta_y + Z \sin \theta_y \tag{12}$$

where

$$Z = a \cos \theta_r + b \sin \theta_r \tag{13}$$

Using Eqs. (11) and (12) gives

$$\cos \theta_y = \frac{Zu + cw}{Z^2 + c^2} \tag{14}$$

$$\sin \theta_y = \frac{Zw - cw}{Z^2 + c^2} \tag{15}$$

Substituting Eqs. (8) and (9) into Eq. (13) gives

$$Z = \pm \sqrt{a^2 + b^2 - v^2} \tag{16}$$

Substituting Eq. (16) into Eqs. (14) and (15) and rewriting gives

$$\tan \theta_y = \frac{-cu \pm w \sqrt{a^2 + b^2 - v^2}}{cw \pm u \sqrt{a^2 + b^2 - v^2}} \tag{17}$$

The two solutions of Eq. (10) are

$$\theta_{r_1} = \tan^{-1}\left(\frac{-va + b\sqrt{a^2 + b^2 - v^2}}{vb + a\sqrt{a^2 + b^2 - v^2}}\right) \tag{18}$$

and

$$\theta_{r_2} = \tan^{-1}\left(\frac{-va + b\sqrt{a^2 + b^2 - v^2}}{vb + a\sqrt{a^2 + b^2 - v^2}}\right) \tag{19}$$

Two other permissible turns are

$$\theta_{r_3} = \theta_{r_1} - 360°$$

and

$$\theta_{r_4} = \theta_{r_2} - 360°$$

The two solutions of Eq. (17) are

$$\theta_{y_1} = \tan^{-1}\left(\frac{-cu + w\sqrt{a^2 + b^2 - v^2}}{cw + u\sqrt{a^2 + b^2 - v^2}}\right) \tag{20}$$

and

$$\theta_{y_2} = \tan^{-1}\left(\frac{-cu + w\sqrt{a^2 + b^2 - v^2}}{cw + u\sqrt{a^2 + b^2 - v^2}}\right) \tag{21}$$

Two other permissible turns are

$$\theta_{y_3} = \theta_{y_1} - 360°$$

and

$$\theta_{y_4} = \theta_{y_2} - 360°$$

There are eight permissible roll and yaw turn magnitude combinations, namely

θ_{r_1} and θ_{y_1}

θ_{r_1} and θ_{y_3}

θ_{r_3} and θ_{y_1}

θ_{r_3} and θ_{y_3}

θ_{r_2} and θ_{y_2}

θ_{r_2} and θ_{y_4}

θ_{r_4} and θ_{y_2}

θ_{r_4} and θ_{y_4}

Any one of these eight sets of turns can be used to correctly orient the vector $\overline{VT}$ with the vector $\overline{VC}$. All eight sets of turn angles are listed by the program. Expressions similar to Eqs. (10) and (17) can be obtained for any other set of two maneuver turns.

Note that in the roll-yaw turn sequence example no solution exists if $v^2 > (a^2 + b^2)$. Similar geometric constraints exist for other turn sequences as well and thus necessitate the use of three-turn sequences to achieve the required vector pointing. Because a three-turn sequence does not have a unique mathematical solution it must be treated in a parametric fashion as indicated in the following section.

IV. PROGRAM DESCRIPTION

A. Introduction

The program is designed to solve primarily for those turn magnitudes that will point a spacecraft-fixed vector in a specified inertially-fixed direction. The user can arbitrarily select any two- or three-turn combination and all possible turn magnitudes will be generated. One program option outputs the intermediate positions of a spacecraft-fixed reference vector throughout each spacecraft turn. A second option permits the user to select a turn combination and the turn magnitudes, and the program then computes the orientation of a spacecraft-fixed reference vector before and after each turn. The basic program and the two options will be discussed in the next three sections followed by a section on program usage.

B. Basic Program Description

The functional description of the program is best explained by making use of a functional flow diagram, Fig. 4. The cone and clock angles of the pitch and roll axes are read into the program. These data are used to construct the coordinate transformation matrix between the ($\overline{X}$, $\overline{Y}$, $\overline{Z}$) and ($\overline{A}$, $\overline{B}$, $\overline{C}$) coordinate systems as well as the inverse transformation matrix. The clock and cone angles of the spacecraft- and inertially-fixed vectors and the reference vector are read into the program. The desired turn sequence data is then read into the program. If a two-turn sequence was requested and if a solution can be found it is printed out and a new set of vectors and another turn sequence can be selected.

The solution consists of a listing of the eight possible sets of turn magnitudes that will correctly orient the spacecraft. The position of the reference vector before and after each turn is also shown. If a two-turn sequence is not possible and/or an alternative three-turn sequence was specified it will be computed; otherwise a new set of data is read in.

If a three-turn sequence was requested the user must specify the magnitude of the first turn. This is done by giving an initial turn magnitude, a final turn magnitude, and the amount to be used in incrementing the turn magnitude after each solution. The program solves for the second and third turns after turning the spacecraft the amount specified for the

first turn. After printing the solution the first turn magnitude is incremented and the process repeated until the specified final value for the first turn is reached. The computations are then terminated and a new set of data may then be read into the program to start another computation.

C. Detailed Output Option

After having found a maneuver turn solution a program option may be exercised that solves for and prints intermediate positions of the reference vector as the spacecraft moves through each turn. The option is selected by specifying in the input data the increment to be used in stepping the spacecraft through each turn. There is no restriction on the size or polarity of the increment.

D. Specific Turn Option

If a particular set of maneuver turn magnitudes is specified in the input data a program option is exercised that will compute and print the orientation of the reference vector before and after each of the turns. Any two- or three-turn sequence can be selected. The main body of the program is bypassed and after printing the solution a new set of data may be selected. By user option, two types of turns can be computed: a coordinate transformation may be chosen wherein the position of the fixed reference vector is computed relative to a rotated coordinate system, or a vector transformation can be selected wherein the reference vector is rotated relative to a fixed coordinate system.

V. OPERATING INSTRUCTIONS

A. Input

Data input to the program is through formatted cards. A set of three cards constitutes the initial data set. At the conclusion of a computation a new case may be run by specifying new data as indicated on cards 2 and 3.

Card 1	AX, BX, AZ, BZ
Format	4F8.2
Card 2	AVC, BVC, AT, BT, ARV, BRV, Z1, Z2, Z3, VTURN
Format	9F8.2, I1
Card 3	TURN, TURN1, TURN2, ANG1, ANG2, DELT, DELANG
Format	I1, 1X, 2I1, 2X, 4F8.2

The symbols are defined in Table 1, which describes the parameters used on the input cards of the program. The user has the option of selecting any one of six two-turn maneuver sequences to accomplish the spacecraft reorientation or of omitting the two-turn sequence completely. The integer input parameter TURN must be set in accordance with Table 2 to obtain the desired sequence.

The user has the option of arbitrarily selecting any three-turn maneuver sequence to accomplish the spacecraft reorientation. Two input parameters must be set. The first integer parameter, TURN1, establishes the first turn in the sequence as indicated in Table 3. The second integer parameter, TURN2, establishes the second and third turns of the sequence as indicated in Table 4. If TURN1 is set equal to zero no three-turn sequence is performed.

B. Output

The program output consists of a tabulation of the input parameters followed by a listing of the maneuver solution. If the detailed solution option is selected additional printed output is generated that traces the reference vector through each turn. The specific turn option suppresses the

Table 1. Input symbols

Name	Type	Description
ANG1	REAL	Initial value of angle used for first turn of three-turn sequence (deg)
ANG2	REAL	Final value of angle used for first turn of three-turn sequence (deg)
ARV	REAL	Clock angle of vector $\overline{RV}$ (deg)
AT	REAL	Clock angle of vector $\overline{VT}$ (deg)
AVC	REAL	Clock angle of vector $\overline{VC}$ (deg)
AX	REAL	Clock angle of pitch axis (deg)
AZ	REAL	Clock angle of roll axis (deg)
BRV	REAL	Cone angle of vector $\overline{RV}$ (deg)
BT	REAL	Cone angle of vector $\overline{VT}$ (deg)
BVC	REAL	Cone angle of vector $\overline{VC}$ (deg)
BX	REAL	Cone angle of pitch axis (deg)
BZ	REAL	Cone angle of roll axis (deg)
DELANG	REAL	Angle used to increment the turn magnitude when generating detailed maneuver turn output (deg). No detailed output is generated if DELANG = 0
DELT	REAL	Angle by which ANG1 is increased for next three-turn maneuver sequence calculation (deg)
TURN	INTEGER	Use is defined in Table 2
TURN1	INTEGER	Use is defined in Table 3
TURN2	INTEGER	Use is defined in Table 4
VTURN	INTEGER	Used if specific turn option is selected; set VTURN = 0 if coordinate transformation is desired, set VTURN = 1 if vector transformation is desired
Z1	REAL	First maneuver turn magnitude, if specific turn option is desired. Option is selected only if Z1 $\neq$ 0 (deg)
Z2	REAL	Second maneuver turn magnitude if specific turn option is desired; otherwise set Z2 = 0 (deg)
Z3	REAL	Third maneuver turn magnitude if specific turn option is desired; otherwise set Z3 = 0 (deg)

Table 2. Definition of integer input parameter TURN

First turn	Second turn	Input parameter TURN
Yaw	Roll	1
Roll	Yaw	2
Pitch	Roll	3
Roll	Pitch	4
Yaw	Pitch	5
Pitch	Yaw	6
Omit two-turn sequence		7

maneuver solution and lists the reference vector position before and after the specified turns. These three basic types of output are illustrated in Figs. 5 through 8.

Several diagnostic messages may also be printed by the program. If the inverse coordinate transformation matrix cannot be found a message indicating this fact is produced (See Appendix Subsection F-4 for details). If, because of geometric constraints, a requested turn sequence is not feasible it is so stated in a message (See Appendix Subsection C-4 for details).

Table 3. Definition of integer input parameter TURN1

First turn	Input parameter TURN1
Omit three-turn sequence	0
Roll	1
Pitch	2
Yaw	3

Table 4. Definition of integer input parameter TURN2

Second turn	Third turn	Input parameter TURN2
Yaw	Roll	1
Roll	Yaw	2
Pitch	Roll	3
Roll	Pitch	4
Yaw	Pitch	5
Pitch	Yaw	6

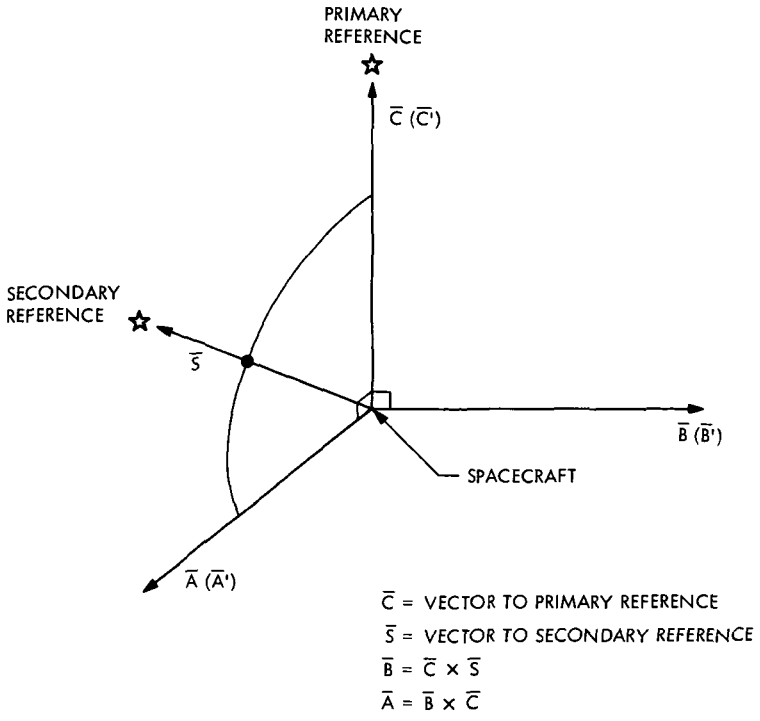


Fig. 1. Definition of $(\overline{A}, \overline{B}, \overline{C})$ coordinate system

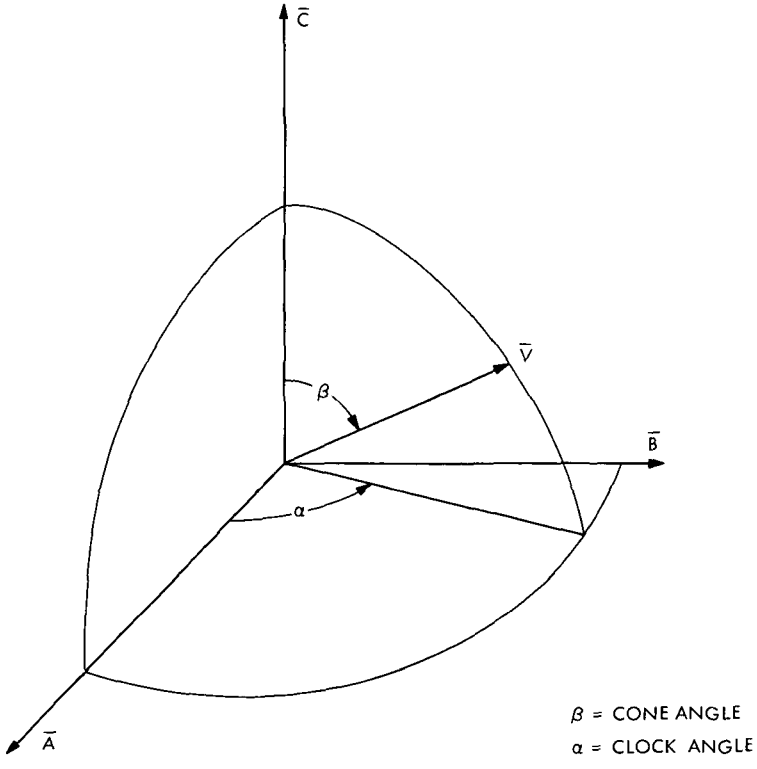


Fig. 2. Definition of arbitrary vector in terms of clock and cone angles

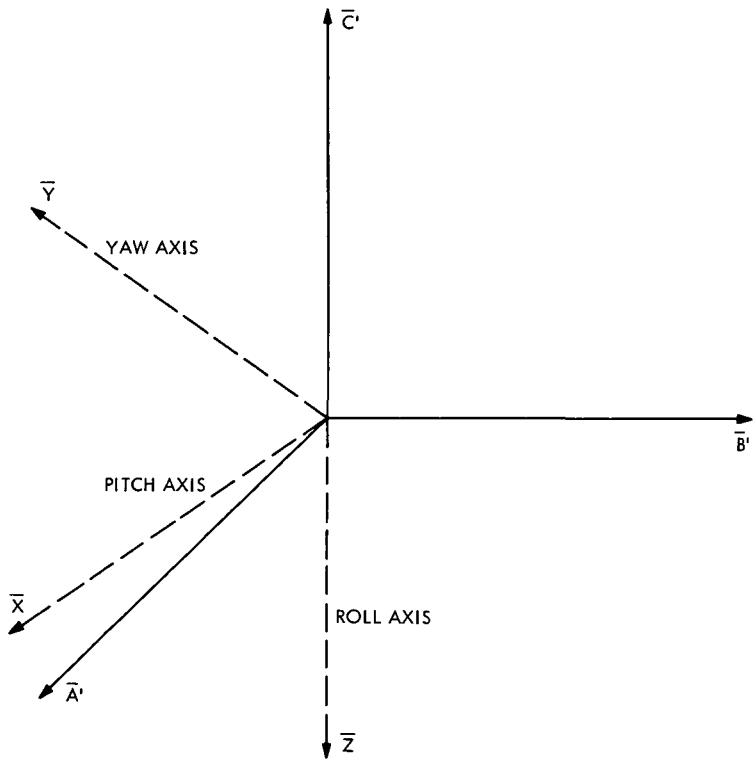


Fig. 3. Typical relationship between the $(\overline{X}, \overline{Y}, \overline{Z})$ and $(\overline{A}', \overline{B}', \overline{C}')$ coordinate systems

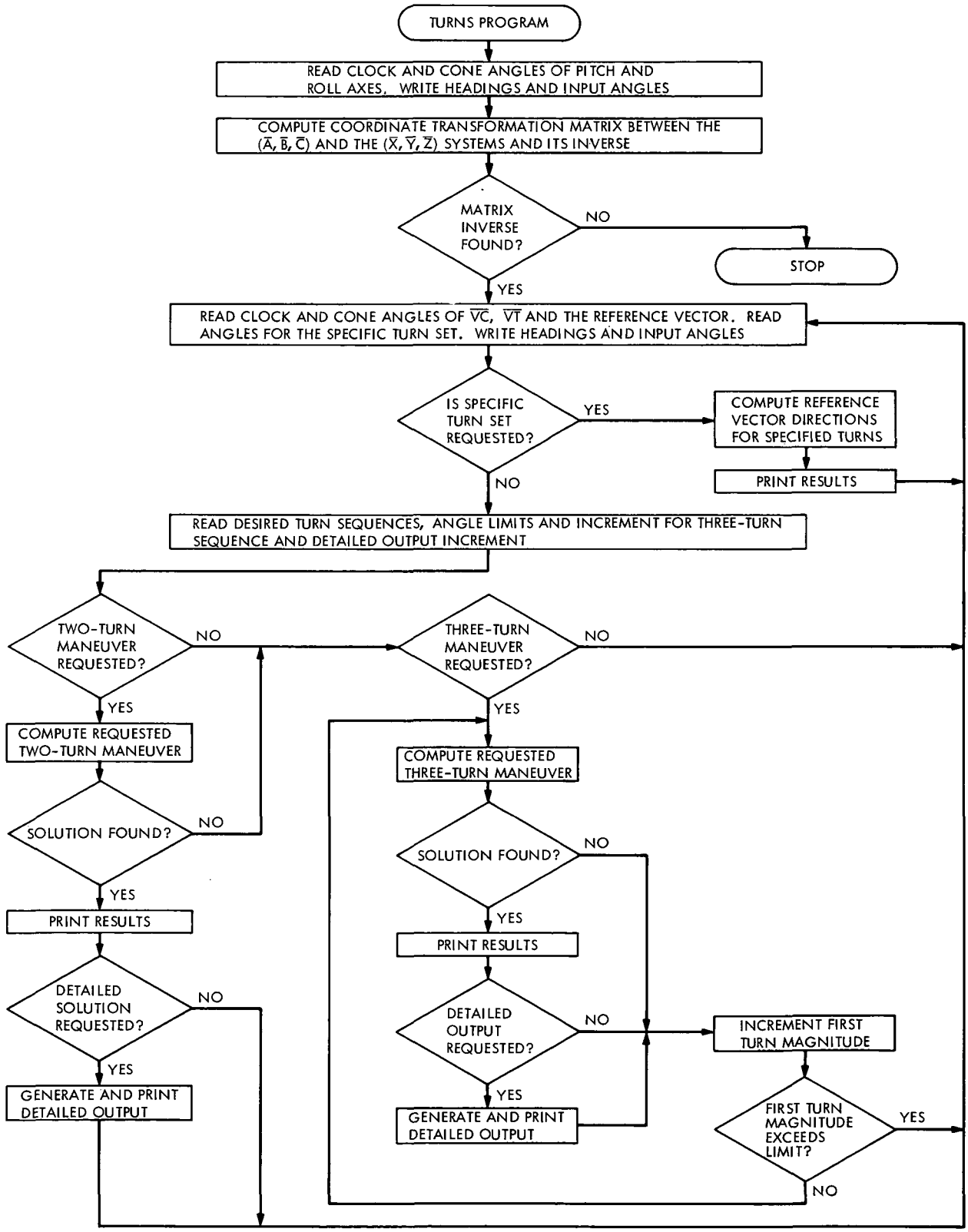


Fig. 4. General purpose maneuver turns program functional flow diagram

PROGRAM TO COMPUTE MANEUVER TURNS TO ROTATE A VECTOR VT TO THE DIRECTION OF A VECTOR VC

PITCH AXIS CLOCK ANGLE	PITCH AXIS CONE ANGLE	ROLL AXIS CLOCK ANGLE	ROLL AXIS CONE ANGLE
-32.20	90.00	.00	180.00

VECTOR VC CLOCK ANGLE	VECTOR VC CONE ANGLE	VECTOR VT CLOCK ANGLE	VECTOR VT CONE ANGLE	REFERENCE VECTOR CLOCK ANGLE	REFERENCE VECTOR CONE ANGLE
90.00	30.00	150.00	70.00	80.00	120.00

PITCH ANGLE	YAW ANGLE	ROLL ANGLE	REF. VECTOR BEFORE 1ST TURN CLOCK ANGLE	REF. VECTOR BEFORE 1ST TURN CONE ANGLE	REF. VECTOR BEFORE 2ND TURN CLOCK ANGLE	REF. VECTOR BEFORE 2ND TURN CONE ANGLE	REF. VECTOR BEFORE 3RD TURN CLOCK ANGLE	REF. VECTOR BEFORE 3RD TURN CONE ANGLE	REF. VECTOR AFTER 3RD TURN CLOCK ANGLE	REF. VECTOR AFTER 3RD TURN CONE ANGLE
10.00	313.41	39.32	80.00	120.00	82.77	129.17	39.38	132.20	78.70	132.20
10.00	313.41	-320.68	80.00	120.00	82.77	129.17	39.38	132.20	78.70	132.20
10.00	-46.59	39.32	80.00	120.00	82.77	129.17	39.38	132.20	78.70	132.20
10.00	-46.59	-320.68	80.00	120.00	82.77	129.17	39.38	132.20	78.70	132.20
10.00	84.34	145.08	80.00	120.00	82.77	129.17	101.04	74.73	246.12	74.73
10.00	84.34	-214.92	80.00	120.00	82.77	129.17	101.04	74.73	246.12	74.73
10.00	-275.66	145.08	80.00	120.00	82.77	129.17	101.04	74.73	246.12	74.73
10.00	-275.66	-214.92	80.00	120.00	82.77	129.17	101.04	74.73	246.12	74.73

Fig. 5. Typical program output

PROGRAM TO COMPUTE MANEUVER TURNS TO ROTATE A VECTOR VT TO THE DIRECTION OF A VECTOR VC

PITCH AXIS CLOCK ANGLE	PITCH AXIS CONE ANGLE	ROLL AXIS CLOCK ANGLE	ROLL AXIS CONE ANGLE
-32.20	90.00	.00	180.00

VECTOR VC CLOCK ANGLE	VECTOR VC CONE ANGLE	VECTOR VT CLOCK ANGLE	VECTOR VT CONE ANGLE	REFERENCE VECTOR CLOCK ANGLE	REFERENCE VECTOR CONE ANGLE
90.00	30.00	150.00	70.00	80.00	120.00

ROLL ANGLE	YAW ANGLE	REFERENCE VECTOR BEFORE FIRST TURN CLOCK ANGLE	REFERENCE VECTOR BEFORE FIRST TURN CONE ANGLE	REFERENCE VECTOR BEFORE SECOND TURN CLOCK ANGLE	REFERENCE VECTOR BEFORE SECOND TURN CONE ANGLE	REFERENCE VECTOR AFTER SECOND TURN CLOCK ANGLE	REFERENCE VECTOR AFTER SECOND TURN CONE ANGLE
233.66	260.08	80.00	120.00	313.66	120.00	296.49	24.02
233.66	-99.92	80.00	120.00	313.66	120.00	296.49	24.02
-126.34	260.08	80.00	120.00	313.66	120.00	296.49	24.02
-126.34	-99.92	80.00	120.00	313.66	120.00	296.49	24.02
61.94	319.95	80.00	120.00	141.94	120.00	133.12	159.57
61.94	-40.05	80.00	120.00	141.94	120.00	133.12	159.57
-298.06	319.95	80.00	120.00	141.94	120.00	133.12	159.57
-298.06	-40.05	80.00	120.00	141.94	120.00	133.12	159.57

DETAIL MAPPING OF REFERENCE VECTOR THROUGH THE MANEUVER TURNS

ROLL TURN

ROLL ANGLE	CLOCK ANGLE	CONE ANGLE
.00	80.00	120.00
90.00	170.00	120.00
180.00	260.00	120.00
233.66	313.66	120.00

YAW TURN

YAW ANGLE	CLOCK ANGLE	CONE ANGLE
.00	313.66	120.00
90.00	170.73	147.12
180.00	161.94	60.00
260.08	296.49	24.02

ROLL TURN

ROLL ANGLE	CLOCK ANGLE	CONE ANGLE
.00	80.00	120.00
61.94	141.94	120.00

YAW TURN

YAW ANGLE	CLOCK ANGLE	CONE ANGLE
.00	141.94	120.00
90.00	137.77	30.52
180.00	333.66	60.00
270.00	337.83	149.48
319.95	133.12	159.57

ROLL TURN

ROLL ANGLE	CLOCK ANGLE	CONE ANGLE
.00	80.00	120.00
-90.00	350.00	120.00
-126.34	313.66	120.00

YAW TURN

YAW ANGLE	CLOCK ANGLE	CONE ANGLE
.00	313.66	120.00
-90.00	304.87	32.88
-99.92	296.49	24.02

ROLL TURN

ROLL ANGLE	CLOCK ANGLE	CONE ANGLE
.00	80.00	120.00
-90.00	350.00	120.00
-180.00	260.00	120.00
-270.00	170.00	120.00
-298.06	141.94	120.00

YAW TURN

YAW ANGLE	CLOCK ANGLE	CONE ANGLE
.00	141.94	120.00
-40.05	133.12	159.57

Fig. 6. Program output illustrating the detailed solution option

PROGRAM TO COMPUTE MANEUVER TURNS TO ROTATE A VECTOR VT TO THE DIRECTION OF A VECTOR VC

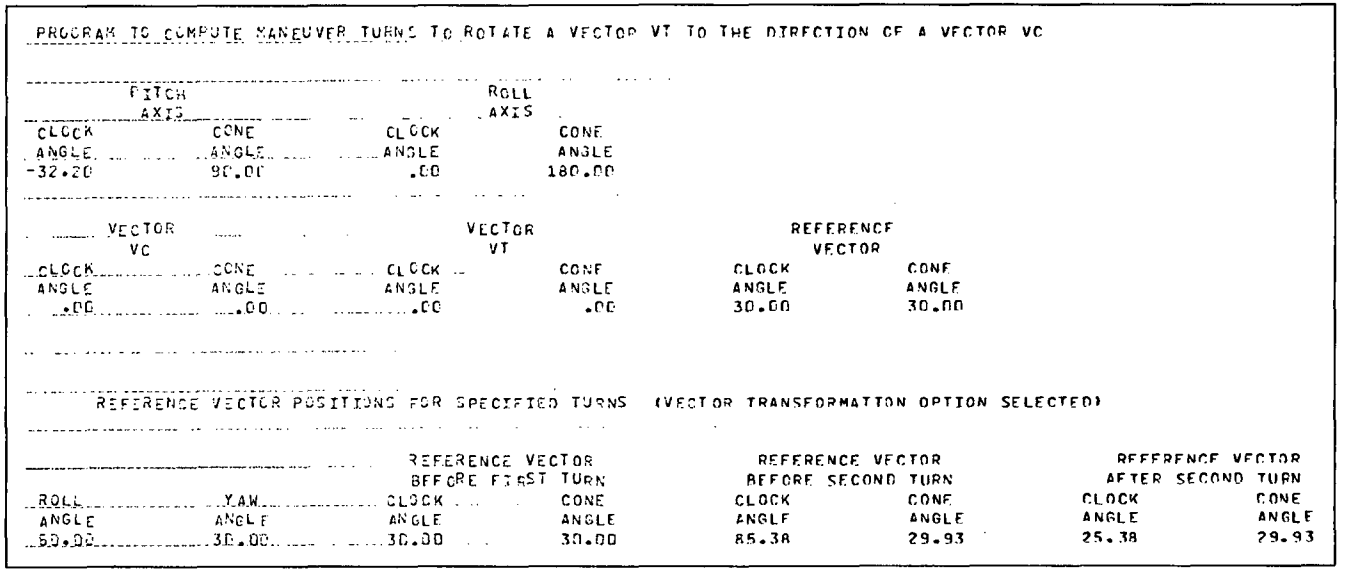
PITCH AXIS		ROLL AXIS	
CLOCK ANGLE	CONE ANGLE	CLOCK ANGLE	CONE ANGLE
-32.20	90.00	.00	180.00

VECTOR VC		VECTOR VT		REFERENCE VECTOR	
CLOCK ANGLE	CONE ANGLE	CLOCK ANGLE	CONE ANGLE	CLOCK ANGLE	CONE ANGLE
.00	.00	.00	.00	30.00	30.00

REFERENCE VECTOR POSITIONS FOR SPECIFIED TURNS (COORDINATE TRANSFORMATION OPTION SELECTED)

		REFERENCE VECTOR BEFORE FIRST TURN		REFERENCE VECTOR BEFORE SECOND TURN		REFERENCE VECTOR AFTER SECOND TURN	
ROLL ANGLE	YAW ANGLE	CLOCK ANGLE	CONE ANGLE	CLOCK ANGLE	CONE ANGLE	CLOCK ANGLE	CONE ANGLE
60.00	30.00	30.00	30.00	90.00	30.00	32.25	27.97

Fig. 7. Program output illustrating the specified turns option (coordinate transformation calculation)



PROGRAM TO COMPUTE MANEUVER TURNS TO ROTATE A VECTOR VT TO THE DIRECTION OF A VECTOR VC

PITCH AXIS		ROLL AXIS	
CLOCK ANGLE	CONE ANGLE	CLOCK ANGLE	CONE ANGLE
-32.20	90.00	.00	180.00

VECTOR VC		VECTOR VT		REFERENCE VECTOR	
CLOCK ANGLE	CONE ANGLE	CLOCK ANGLE	CONE ANGLE	CLOCK ANGLE	CONE ANGLE
.00	.00	.00	.00	30.00	30.00

REFERENCE VECTOR POSITIONS FOR SPECIFIED TURNS (VECTOR TRANSFORMATION OPTION SELECTED)

		REFERENCE VECTOR BEFORE FIRST TURN		REFERENCE VECTOR BEFORE SECOND TURN		REFERENCE VECTOR AFTER SECOND TURN	
ROLL ANGLE	YAW ANGLE	CLOCK ANGLE	CONE ANGLE	CLOCK ANGLE	CONE ANGLE	CLOCK ANGLE	CONE ANGLE
60.00	30.00	30.00	30.00	85.38	29.93	25.38	29.93

Fig. 8. Program output illustrating the specified turns option (vector transformation calculation)

APPENDIX
SUBROUTINE DESCRIPTIONS AND PROGRAM LISTINGS

The following sections provide a detailed description of the main program and the nine subroutines that comprise the general purpose maneuver turns program. Listings of the main program and each subroutine are also given. The entire program is coded in Fortran V for use on the JPL Scientific Computing Facility Univac 1108 computer.

A. Description of Main Program

1. Identification.

Program TURNS
FORTRAN V

2. Purpose. This is a main (or driver) program that accepts input data, controls the sequence of operations as specified by the data, and prints the output results.

3. Restrictions.

(1)	COMMON used:	COM1, COM2, COM3, COM4, COM5, COM6, COM7, COM9, COM11, COM12
(2)	Subroutines called:	TMATRX, CCLABC, TRANS, YRTURN, RYTURN, PRTURN, RPTURN, YPTURN, PYTURN, RTURN, PTURN, YTURN, CONV, DETAIL, XYZCCL
(3)	Files used:	Unit 5 System input file (READ$) Unit 6 System output file (PRINT$)

4. Method. The general operating scheme of the program is described in Fig. 4, which illustrates the functional modes of the program.

5. Use.

(1) Input

Through files:

Unit 5 System input file (READ$)
Input parameters as described in Subsection V-A, Input

(2) Output

Through files:

Unit 6 System output file (PRINT$)
Output parameters as described in Section V-B, Output

Through COMMON blocks:

/COM1/
As described in the description of subroutine TMATRX

/COM2/
As described in the description of subroutine YRTURN

/COM3/
As described in the description of subroutine TMATRX

/COM4/
As described in the description of subroutine TMATRX

/COM5/
As described in the description of subroutine TMATRX

/COM6/
As described in the description of subroutine DETAIL

/COM7/
As described in the description of subroutine DETAIL

/COM9/
As described in the description of subroutine DETAIL

/COM11/
As described in the description of subroutine YAWTRN

/COM12/
As described in the description of subroutine DETAIL

6. Fortran V Listing of Main (Driver) Program.

```
C      GENERAL PURPOSE MANEUVER TURNS PROGRAM
       INTEGER TURN,TURN1,TURN2,VTURN,TRN,TRNS
       DIMENSION V(3),VT(3),TR(4),TY(4),TP(4),TI(3,3),VABC(3),VTABC(3),RV
      1ABC(3),RV(3),RV11(3),RV12(3),RV21(3),RV22(3),XX(4),YY(4),V1(3),RV1
      2(3),RSAVE(3),VSAVE(3),Q(3)
       COMMON/COM1/AX,BX,AZ,BZ/COM2/V,VT,TY,TR,TP/COM3/DEG/COM4/TI/COM5/K
      1EY/COM6/RV11,RV12,RV21,RV22,A11,A12,A21,A22,B11,B12,B21,B22
       COMMON/COM7/ITRIG,ANG,TURN1,TURN2,T1,T2,T3/COM9/NUMTRN,TURN,RV/COM
      111/DELANG/COM12/RSAVE,RV1
       DATA DEG/57.295779/T1,T2,T3/-ROLL -,-PITCH-,- YAW -/DELT,DELANG,LK
      1EY,KEY/1.,0.,0,0/
C      READ CLOCK AND CONE OF PITCH AND ROLL AXES
       READ(5,1000)AX,BX,AZ,BZ
       WRITE(6,3500)
       WRITE(6,3600)
       WRITE(6,3000)
       WRITE(6,4000)
       WRITE(6,4100)
       WRITE(6,4200)
       WRITE(6,4300)
       WRITE(6,2000)AX,BX,AZ,BZ
C      COMPUTE TRANSFORMATION MATRIX AND ITS INVERSE
       CALL TMATRX
       IF(KEY.EQ.1) STOP
C      READ CLOCK AND CONE OF VC,VT AND REFERENCE VECTORS
C      Z1 IS THE MAGNITUDE OF THE 1ST TURN OF A SPECIFIC TURN SET
C      Z2 IS THE MAGNITUDE OF THE 2ND TURN OF A SPECIFIC TURN SET
C      Z3 IS THE MAGNITUDE OF THE 3RD TURN OF A SPECIFIC TURN SET
1      READ(5,1000,END=500) AVC,BVC,AT,BT,ARV,BRV,Z1,Z2,Z3,VTURN
       WRITE(6,3000)
       WRITE(6,4600)
       WRITE(6,4700)
       WRITE(6,4250)
       WRITE(6,4400)
       WRITE(6,2000)AVC,BVC,AT,BT,ARV,BRV
       WRITE(6,3000)
       WRITE(6,3000)
       KKEY=0
       KEY=0
       ITRIG=0
       IF(ABS(Z1).GT.0.) GO TO 200
C      CHANGE CONE AND CLOCK TO ABC SYSTEM
       CALL CCLABC(VABC,AVC,BVC)
       CALL CCLABC(VTABC,AT,BT)
       CALL CCLABC(RVABC,ARV,BRV)
C      CHANGE FROM ABC SYSTEM TO XYZ SYSTEM
       CALL TRANS(VABC,V)
       CALL TRANS(VTABC,VT)
       CALL TRANS(RVABC,RV)
       DO 4 I=1,3
       VSAVE(I)=V(I)
4      RSAVE(I)=RV(I)
       DO 5 I=1,4
       XX(I)=0.
5      YY(I)=0.
C      READ DESIRED TURN SEQUENCE
C      DELANG IS THE ANGLE STEP SIZE FOR DETAIL OUTPUT
6      READ(5,1500) TURN,TURN1,TURN2,ANG1,ANG2,DELT,DELANG
       IF(ABS(Z1).GT.0.) GO TO 202
C      COMPUTE TWO-TURN SEQUENCE
```

```
9       GO TO (10,20,30,40,50,60,110), TURN
10      CALL YRTURN
        IF(KEY.GT.1) GO TO 110
        GO TO 80
20      CALL RYTURN
        IF(KEY.GT.1) GO TO 110
        GO TO 70
30      CALL PRTURN
        IF(KEY.GT.1) GO TO 110
        GO TO 90
40      CALL RPTURN
        IF(KEY.GT.1) GO TO 110
        GO TO 70
50      CALL YPTURN
        IF(KEY.GT.1) GO TO 110
        GO TO 80
60      CALL PYTURN
        IF(KEY.GT.1) GO TO 110
        GO TO 90
70      CALL RTURN(RV,RV11,Q,TR(1))
        CALL RTURN(RV,RV12,Q,TR(2))
        DO 71 I=1,4
71      XX(I)=TR(I)
        IF(TURN.EQ.2) GO TO 72
        IF(TURN.EQ.4) GO TO 74
72      CALL YTURN(RV11,RV21,Q,TY(1))
        CALL YTURN(RV12,RV22,Q,TY(2))
        DO 73 I=1,4
73      YY(I)=TY(I)
        GO TO 100
74      CALL PTURN(RV11,RV21,Q,TP(1))
        CALL PTURN(RV12,RV22,Q,TP(2))
        DO 75 I=1,4
75      YY(I)=TP(I)
        GO TO 100
80      CALL YTURN(RV,RV11,Q,TY(1))
        CALL YTURN(RV,RV12,Q,TY(2))
        DO 81 I=1,4
81      XX(I)=TY(I)
        IF(TURN.EQ.1) GO TO 82
        IF(TURN.EQ.5) GO TO 84
82      CALL RTURN(RV11,RV21,Q,TR(1))
        CALL RTURN(RV12,RV22,Q,TR(2))
        DO 83 I=1,4
83      YY(I)=TR(I)
        GO TO 100
84      CALL PTURN(RV11,RV21,Q,TP(1))
        CALL PTURN(RV12,RV22,Q,TP(2))
        DO 85 I=1,4
85      YY(I)=TP(I)
        GO TO 100
90      CALL PTURN(RV,RV11,Q,TP(1))
        CALL PTURN(RV,RV12,Q,TP(2))
        DO 91 I=1,4
91      XX(I)=TP(I)
        IF(TURN.EQ.3) GO TO 92
        IF(TURN.EQ.6) GO TO  94
92      CALL RTURN(RV11,RV21,Q,TR(1))
        CALL RTURN(RV12,RV22,Q,TR(2))
        DO 93 I=1,4
```

```
93    YY(I)=TR(I)
      GO TO 100
94    CALL YTURN(RV11,RV21,Q,TY(1))
      CALL YTURN(RV12,RV22,Q,TY(2))
      DO 95 I=1,4
95    YY(I)=TY(I)
100   IF(KKEY.EQ.1) GO TO 105
101   IF(ABS(Z1).GT.0..AND.VTURN.EQ.0) WRITE(6,5500)
      IF(ABS(Z1).GT.0..AND.VTURN.EQ.1) WRITE(6,5600)
      WRITE(6,5000)
      WRITE(6,4800)
      IF(TURN.EQ.1) WRITE(6,4900) T3,T1
      IF(TURN.EQ.2) WRITE(6,4900) T1,T3
      IF(TURN.EQ.3) WRITE(6,4900) T2,T1
      IF(TURN.EQ.4) WRITE(6,4900) T1,T2
      IF(TURN.EQ.5) WRITE(6,4900) T3,T2
      IF(TURN.EQ.6) WRITE(6,4900) T2,T3
      WRITE(6,4500)
      IF(ABS(Z1).GT.0.) GO TO 210
      CALL CONV
      WRITE(6,2000)XX(1),YY(1),ARV,BRV,A11,B11,A21,B21
      WRITE(6,2000)XX(1),YY(3),ARV,BRV,A11,B11,A21,B21
      WRITE(6,2000)XX(3),YY(1),ARV,BRV,A11,B11,A21,B21
      WRITE(6,2000)XX(3),YY(3),ARV,BRV,A11,B11,A21,B21
      WRITE(6,2000)XX(2),YY(2),ARV,BRV,A12,B12,A22,B22
      WRITE(6,2000)XX(2),YY(4),ARV,BRV,A12,B12,A22,B22
      WRITE(6,2000)XX(4),YY(2),ARV,BRV,A12,B12,A22,B22
      WRITE(6,2000)XX(4),YY(4),ARV,BRV,A12,B12,A22,B22
      WRITE(6,3000)
      NUMTRN=2
      IF(DELANG.GT.0.) CALL DETAIL
      GO TO 1
105   IF(ABS(Z1).GT.0..AND.VTURN.EQ.0) WRITE(6,5500)
      IF(ABS(Z1).GT.0..AND.VTURN.EQ.1) WRITE(6,5600)
      WRITE(6,5400)
      WRITE(6,5300)
      IF(TURN1.EQ.1.AND.TURN2.EQ.1)WRITE(6,5100)T1,T3,T1
      IF(TURN1.EQ.1.AND.TURN2.EQ.2)WRITE(6,5100)T1,T1,T3
      IF(TURN1.EQ.1.AND.TURN2.EQ.3)WRITE(6,5100)T1,T2,T1
      IF(TURN1.EQ.1.AND.TURN2.EQ.4)WRITE(6,5100)T1,T1,T2
      IF(TURN1.EQ.1.AND.TURN2.EQ.5)WRITE(6,5100)T1,T3,T2
      IF(TURN1.EQ.1.AND.TURN2.EQ.6)WRITE(6,5100)T1,T2,T3
      IF(TURN1.EQ.2.AND.TURN2.EQ.1)WRITE(6,5100)T2,T3,T1
      IF(TURN1.EQ.2.AND.TURN2.EQ.2)WRITE(6,5100)T2,T1,T3
      IF(TURN1.EQ.2.AND.TURN2.EQ.3)WRITE(6,5100)T2,T2,T1
      IF(TURN1.EQ.2.AND.TURN2.EQ.4)WRITE(6,5100)T2,T1,T2
      IF(TURN1.EQ.2.AND.TURN2.EQ.5)WRITE(6,5100)T2,T3,T2
      IF(TURN1.EQ.2.AND.TURN2.EQ.6)WRITE(6,5100)T2,T2,T3
      IF(TURN1.EQ.3.AND.TURN2.EQ.1)WRITE(6,5100)T3,T3,T1
      IF(TURN1.EQ.3.AND.TURN2.EQ.2)WRITE(6,5100)T3,T1,T3
      IF(TURN1.EQ.3.AND.TURN2.EQ.3)WRITE(6,5100)T3,T2,T1
      IF(TURN1.EQ.3.AND.TURN2.EQ.4)WRITE(6,5100)T3,T1,T2
      IF(TURN1.EQ.3.AND.TURN2.EQ.5)WRITE(6,5100)T3,T3,T2
      IF(TURN1.EQ.3.AND.TURN2.EQ.6)WRITE(6,5100)T3,T2,T3
      WRITE(6,5200)
      IF(ABS(Z1).GT.0.) GO TO 220
      CALL CONV
      WRITE(6,2500)ANG,XX(1),YY(1),ARV,BRV,A1,B1,A11,B11,A21,B21
      WRITE(6,2500)ANG,XX(1),YY(3),ARV,BRV,A1,B1,A11,B11,A21,B21
      WRITE(6,2500)ANG,XX(3),YY(1),ARV,BRV,A1,B1,A11,B11,A21,B21
```

```
      WRITE(6,2500)ANG,XX(3),YY(3),ARV,BRV,A1,B1,A11,B11,A21,B21
      WRITE(6,2500)ANG,XX(2),YY(2),ARV,BRV,A1,B1,A12,B12,A22,B22
      WRITE(6,2500)ANG,XX(2),YY(4),ARV,BRV,A1,B1,A12,B12,A22,B22
      WRITE(6,2500)ANG,XX(4),YY(2),ARV,BRV,A1,B1,A12,B12,A22,B22
      WRITE(6,2500)ANG,XX(4),YY(4),ARV,BRV,A1,B1,A12,B12,A22,B22
      WRITE(6,3000)
      NUMTRN=3
      IF(DELANG.GT.0.) CALL DETAIL
106   ANG=ANG+DELT
      IF(ANG.GT.ANG2) GO TO 1
      DO 107 I=1,3
      RV(I)=RSAVE(I)
107   V(I)=VSAVE(I)
      GO TO 115
C     COMPUTE THREE-TURN SEQUENCE IF REQUIRED
110   IF(TURN1.EQ.0) GO TO 1
      KEY=0
      KKEY=1
      ITRIG=ITRIG+1
      IF(ITRIG.GT.1) GO TO 106
      ANG=ANG1
      TURN=TURN2
115   GO TO (120,130,140),TURN1
120   CALL RTURN(V,V1,Q,ANG)
      CALL RTURN(RV,RV1,Q,ANG)
      GO TO 150
130   CALL PTURN(V,V1,Q,ANG)
      CALL PTURN(RV,RV1,Q,ANG)
      GO TO 150
140   CALL YTURN(V,V1,Q,ANG)
      CALL YTURN(RV,RV1,Q,ANG)
150   DO 151 I=1,3
      V(I)=V1(I)
151   RV(I)=RV1(I)
      CALL XYZCCL(RV1,A1,B1)
      GO TO 9
C     COMPUTES REFERENCE VECTOR DIRECTIONS FOR SPECIFIED TURN ANGLES
200   CALL CCLABC(RVABC,ARV,BRV)
      CALL TRANS(RVABC,RV)
      DO 201 I=1,3
201   RSAVE(I)=RV(I)
      GO TO 6
202   IF(TURN.EQ.7) GO TO 105
      GO TO 101
210   IF(ABS(Z1).GT.0..AND.VTURN.EQ.1) GO TO 300
      GO TO (211,212,213,214,215,216),TURN
211   CALL YTURN(RSAVE,RV11,Q,Z1)
      CALL RTURN(RV11,RV22,Q,Z2)
      GO TO 217
212   CALL RTURN(RSAVE,RV11,Q,Z1)
      CALL YTURN(RV11,RV22,Q,Z2)
      GO TO 217
213   CALL PTURN(RSAVE,RV11,Q,Z1)
      CALL RTURN(RV11,RV22,Q,Z2)
      GO TO 217
214   CALL RTURN(RSAVE,RV11,Q,Z1)
      CALL PTURN(RV11,RV22,Q,Z2)
      GO TO 217
215   CALL YTURN(RSAVE,RV11,Q,Z1)
      CALL PTURN(RV11,RV22,Q,Z2)
      GO TO 217
```

```
216   CALL PTURN(RSAVE,RV11,Q,Z1)
      CALL YTURN(RV11,RV22,Q,Z2)
217   IF(LKEY.EQ.1) GO TO 225
      CALL XYZCCL(RV11,D1,D2)
      CALL XYZCCL(RV22,D3,D4)
      WRITE(6,2000) Z1,Z2,ARV,BRV,D1,D2,D3,D4
      GO TO 1
220   TURN=TURN2
      LKEY=1
      IF(ABS(Z1).GT.0..AND.VTURN.EQ.1) GO TO 310
      GO TO(221,222,223),TURN1
221   CALL RTURN(RV,RSAVE,Q,Z1)
      GO TO 224
222   CALL PTURN(RV,RSAVE,Q,Z1)
      GO TO 224
223   CALL YTURN(RV,RSAVE,Q,Z1)
224   Z4=Z1
      Z1=Z2
      Z2=Z3
      GO TO 210
225   LKEY=0
      IF(ABS(Z1).GT.0..AND.VTURN.EQ.1) Z1=Z2
      IF(ABS(Z1).GT.0..AND.VTURN.EQ.1) Z2=Z3
      CALL XYZCCL(RSAVE,D1,D2)
      CALL XYZCCL(RV11,D3,D4)
      CALL XYZCCL(RV22,D5,D6)
      WRITE(6,2500) Z4,Z1,Z2,ARV,BRV,D1,D2,D3,D4,D5,D6
      GO TO 1
300   TRN=TURN
301   GO TO (302,303,304,305,306,307), TRN
302   CALL RTURN(RSAVE,Q,RV11,Z2)
      CALL YTURN(RV11,Q,RV22,Z1)
      GO TO 217
303   CALL YTURN(RSAVE,Q,RV11,Z2)
      CALL RTURN(RV11,Q,RV22,Z1)
      GO TO 217
304   CALL RTURN(RSAVE,Q,RV11,Z2)
      CALL PTURN(RV11,Q,RV22,Z1)
      GO TO 217
305   CALL PTURN(RSAVE,Q,RV11,Z2)
      CALL RTURN(RV11,Q,RV22,Z1)
      GO TO 217
306   CALL PTURN(RSAVE,Q,RV11,Z2)
      CALL YTURN(RV11,Q,RV22,Z1)
      GO TO 217
307   CALL YTURN(RSAVE,Q,RV11,Z2)
      CALL PTURN(RV11,Q,RV22,Z1)
      GO TO 217
310   IF(TURN1.EQ.3.AND.(TURN2.EQ.2.OR.TURN2.EQ.4))TRN=1
      IF(TURN1.EQ.1.AND.(TURN2.EQ.1.OR.TURN2.EQ.5))TRN=2
      IF(TURN1.EQ.2.AND.(TURN2.EQ.2.OR.TURN2.EQ.4))TRN=3
      IF(TURN1.EQ.1.AND.(TURN2.EQ.3.OR.TURN2.EQ.6))TRN=4
      IF(TURN1.EQ.3.AND.(TURN2.EQ.3.OR.TURN2.EQ.6))TRN=5
      IF(TURN1.EQ.2.AND.(TURN2.EQ.1.OR.TURN2.EQ.5))TRN=6
      IF(TURN2.EQ.1.OR.TURN2.EQ.3) TRNS=1
      IF(TURN2.EQ.2.OR.TURN2.EQ.6) TRNS=2
      IF(TURN2.EQ.4.OR.TURN2.EQ.5) TRNS=3
      LKEY=1
      GO TO (311,312,313), TRNS
311   CALL RTURN(RV,Q,RSAVE,Z3)
      GO TO 314
```

```
312   CALL YTURN(RV,Q,RSAVE,Z3)
      GO TO 314
313   CALL PTURN(RV,Q,RSAVE,Z3)
314   Z4=Z1
      GO TO 301
500   STOP
1000  FORMAT(9F8.2,I1)
1500  FORMAT(I1,1X,2I1,2X,4F8.1)
2000  FORMAT(1H ,8F15.2)
2500  FORMAT(1H ,11F10.2)
3000  FORMAT(1H0)
3500  FORMAT(1H1)
3600  FORMAT(1H ,10X,-PROGRAM TO COMPUTE MANEUVER TURNS TO ROTATE A VECT
     1OR VT TO THE DIRECTION OF A VECTOR VC-)
4000  FORMAT(1H ,18X,-PITCH-,26X,-ROLL-)
4100  FORMAT(1H ,19X,-AXIS-,26X,-AXIS-)
4200  FORMAT(1H ,2(10X,-CLOCK-,10X,-CONE-,1X))
4250  FORMAT(1H ,3(10X,-CLOCK-,10X,-CONE-,1X))
4300  FORMAT(1H ,4(10X,-ANGLE-))
4400  FORMAT(1H ,6(10X,-ANGLE-))
4500  FORMAT(1H ,8(10X,-ANGLE-))
4600  FORMAT(1H ,16X,-VECTOR-,25X,-VECTOR-,22X,-REFERENCE-)
4700  FORMAT(1H ,18X,-VC-,29X,-VT-,26X,-VECTOR-)
4800  FORMAT(1H ,42X,-BEFORE FIRST TURN-,12X,-BEFORE SECOND TURN-,13X,-A
     1FTER SECOND TURN-)
4900  FORMAT(1H ,2(10X,A5),10X,-CLOCK-,10X,-CONE-,2(11X,-CLOCK-,10X,-CON
     1E-))
5000  FORMAT(1H ,42X,-REFERENCE VECTOR-,14X,-REFERENCE VECTOR-,15X,-REFE
     1RENCE VECTOR-)
5100  FORMAT(1H ,3(5X,A5),4(5X,-CLOCK-,5X,-CONE-,1X))
5200  FORMAT(1H ,11(5X,-ANGLE-))
5300  FORMAT(1H ,35X,-BEFORE 1ST TURN-,5X,-BEFORE 2ND TURN-,5X,-BEFORE 3
     1RD TURN-,6X,-AFTER 3RD TURN-)
5400  FORMAT(1H ,37X,-REF. VECTOR-,3(9X,-REF. VECTOR-))
5500  FORMAT(1H ,15X,-REFERENCE VECTOR POSITIONS FOR SPECIFIED TURNS  (C
     1OORDINATE TRANSFORMATION OPTION SELECTED)-//)
5600  FORMAT(1H ,15X,-REFERENCE VECTOR POSITIONS FOR SPECIFIED TURNS  (V
     1ECTOR TRANSFORMATION OPTION SELECTED)-//)
      END
```

B. Description of Subroutine CONV

1. Identification.

Subroutine CONV
FORTRAN V

2. Purpose. This subroutine computes the cone and clock angles for the reference vector after the turns.

3. Restrictions.

(1) COMMON used: COM6

(2) Subroutines called: XYZCCL

4. Method. This subroutine makes successive calls to XYZCCL using appropriate values for the reference vector.

5. Use.

(1) Calling sequence: CALL CONV

(2) Input

Through COMMON blocks:

/COM6/

RV11(3), RV12(3)	REAL reference vector after the first turn
RV21(3), RV22(3)	REAL reference vector after the second turn

(3) Output

Through COMMON blocks:

/COM6/

A11, A12	REAL clock angles of reference vector after the first turn (deg)
A21, A22	REAL clock angles of reference vector after the second turn (deg)
B11, B12	REAL cone angles of reference vector after the first turn (deg)
B21, B22	REAL cone angles of reference vector after the second turn (deg)

6. Fortran V Listing of Subroutine CONV.

```
      SUBROUTINE CONV
C     COMPUTES CONE AND CLOCK ANGLES FOR REFERENCE VECTOR AFTER TURNS
      DIMENSION RV11(3),RV12(3),RV21(3),RV22(3)
      COMMON/COM6/RV11,RV12,RV21,RV22,A11,A12,A21,A22,B11,B12,B21,B22
      CALL XYZCCL(RV11,A11,B11)
      CALL XYZCCL(RV12,A12,B12)
      CALL XYZCCL(RV21,A21,B21)
      CALL XYZCCL(RV22,A22,B22)
      RETURN
      END
```

C. Description of Subroutine YRTURN

1. Identification.

Subroutine YRTURN
FORTRAN V

2. Purpose. This subroutine computes the turn magnitudes corresponding to any two-turn maneuver.

3. Restrictions.

(1)	COMMON used:	COM2, COM5, COM7
(2)	Subroutine called:	SQRT, ATAN36
(3)	Files used:	Unit 6 System output file (PRINT$)
(4)	Entry points:	RYTURN, PRTURN, RPTURN, YPTURN, PYTURN

4. Method. Before computing the turn magnitudes a check is made of the sign of the quantity under the radical in the solution. (See, for example, Eqs. (18) and (20) in Section III.) If it is negative an error message is printed saying,

A ** TURN SEQUENCE CANNOT BE USED TO ACCOMPLISH
THE DESIRED MANEUVER

or

AN INITIAL ** TURN OF ** DEGREES AND A ** TURN
SEQUENCE CANNOT BE USED TO ACCOMPLISH
THE DESIRED MANEUVER

The spaces marked with ** are filled in by the subroutine with the appropriate words and/or numbers. If it were necessary to print either of the above error messages a trigger parameter, KEY, is set to 2 and a return to the main (driver) program is made.

The subroutine nominally computes the turn magnitudes and then returns to the main (driver) program.

5. Use.

(1) Calling sequence:

```
CALL YRTURN
CALL RYTURN
CALL PRTURN
CALL RPTURN
CALL YPTURN
CALL PYTURN
```

(2) Input

Through COMMON blocks:

/COM2/

V(3)	REAL vector $\overline{VC}$
VT(3)	REAL vector $\overline{VT}$

/COM7/

ITRIG	INTEGER trigger set to indicate two- or three-turn maneuver ITRIG = 0 means two turns; ITRIG > 0 means three turns

(3) Output

Through files:

UNIT 6 System output file (PRINT$)
Error messages

Through COMMON blocks:

/COM2/

TR(4)	REAL roll turn magnitudes (deg)
TP(4)	REAL pitch turn magnitudes (deg)
TY(4)	REAL yaw turn magnitudes (deg)

/COM5/

KEY	INTEGER trigger. When set equal to 2 it indicates that the desired turn sequence cannot be used

6. Fortran V Listing of Subroutine YRTURN.

```
      SUBROUTINE YRTURN
C     THIS PROGRAM COMPUTES COMBINATION TURNS
C     COMPUTES YAW-ROLL TURN
      INTEGER TURN1,TURN2
      DIMENSION V(3),VT(3),TR(4),TY(4),TP(4)
      COMMON/COM2/V,VT,TY,TR,TP/COM5/KEY
      COMMON/COM7/ITRIG,ANG,TURN1,TURN2,T1,T2,T3
      DATA T41,T42/- YAW-ROLL -/T51,T52/- ROLL-YAW -/
      DATA T61,T62/-PITCH-ROLL-/T71,T72/-ROLL-PITCH-/
      DATA T81,T82/-YAW-PITCH -/T91,T92/-PITCH-YAW -/
      EQUIVALENCE (V(1),A),(V(2),B),(V(3),C),(VT(1),X),(VT(2),Y),(VT(3),
     1Z)
      IF(ABS(A*A+C*C-Z*Z).LT.1.E-7) GO TO 1
      IF((A*A+C*C-Z*Z).LT.0.) GO TO 10
      Q=SQRT(A*A+C*C-Z*Z)
      GO TO 2
1     Q=0.
2     TY(1)=ATAN36((Z*A+C*Q),(Z*C-A*Q))
      TY(2)=ATAN36((Z*A-C*Q),(Z*C+A*Q))
      TY(3)=TY(1)-360.
      TY(4)=TY(2)-360.
      TR(1)=ATAN36((X*B+Y*Q),(B*Y-X*Q))
      TR(2)=ATAN36((X*B-Y*Q),(B*Y+X*Q))
      TR(3)=TR(1)-360.
      TR(4)=TR(2)-360.
      RETURN
C     COMPUTES ROLL-YAW TURN
      ENTRY RYTURN
      IF(ABS(A*A+B*B-Y*Y).LT.1.E-7) GO TO 3
      IF((A*A+B*B-Y*Y).LT.0.) GO TO 20
      Q=SQRT(A*A+B*B-Y*Y)
      GO TO 4
3     Q=0.
4     TR(1)=ATAN36((-Y*A+B*Q),(Y*B+A*Q))
      TR(2)=ATAN36((-Y*A-B*Q),(Y*B-A*Q))
      TR(3)=TR(1)-360.
      TR(4)=TR(2)-360.
      TY(1)=ATAN36((-C*X+Z*Q),(C*Z+X*Q))
      TY(2)=ATAN36((-C*X-Z*Q),(C*Z-X*Q))
      TY(3)=TY(1)-360.
      TY(4)=TY(2)-360.
      RETURN
C     COMPUTES PITCH-ROLL TURN
      ENTRY PRTURN
      IF(ABS(B*B+C*C-Z*Z).LT.1.E-7) GO TO 5
      IF((B*B+C*C-Z*Z).LT.0.) GO TO 30
      Q=SQRT(B*B+C*C-Z*Z)
      GO TO 6
5     Q=0.
6     TP(1)=ATAN36((-B*Z+C*Q),(C*Z+B*Q))
      TP(2)=ATAN36((-B*Z-C*Q),(C*Z-B*Q))
      TP(3)=TP(1)-360.
      TP(4)=TP(2)-360.
      TR(1)=ATAN36((-A*Y+X*Q),(A*X+Y*Q))
      TR(2)=ATAN36((-A*Y-X*Q),(A*X-Y*Q))
      TR(3)=TR(1)-360.
      TR(4)=TR(2)-360.
      RETURN
C     COMPUTES ROLL-PITCH TURN
      ENTRY RPTURN
      IF(ABS(A*A+B*B-X*X).LT.1.E-7) GO TO 7
```

```
      IF((A*A+B*B-X*X).LT.0.) GO TO 40
      Q=SQRT(A*A+B*B-X*X)
      GO TO 8
7     Q=0.
8     TR(1)=ATAN36((X*B+A*Q),(X*A-B*Q))
      TR(2)=ATAN36((X*B-A*Q),(X*A+B*Q))
      TR(3)=TR(1)-360.
      TR(4)=TR(2)-360.
      TP(1)=ATAN36((Y*C+Z*Q),(C*Z-Y*Q))
      TP(2)=ATAN36((Y*C-Z*Q),(C*Z+Y*Q))
      TP(3)=TP(1)-360.
      TP(4)=TP(2)-360.
      RETURN
C     COMPUTES YAW-PITCH TURN
      ENTRY YPTURN
      IF(ABS(A*A+C*C-X*X).LT.1.E-7) GO TO 9
      IF((A*A+C*C-X*X).LT.0.) GO TO 50
      Q=SQRT(A*A+C*C-X*X)
      GO TO 11
9     Q=0.
11    TY(1)=ATAN36((-X*C+A*Q),(X*A+C*Q))
      TY(2)=ATAN36((-X*C-A*Q),(X*A-C*Q))
      TY(3)=TY(1)-360.
      TY(4)=TY(2)-360.
      TP(1)=ATAN36((-Z*B+Y*Q),(Y*B+Z*Q))
      TP(2)=ATAN36((-Z*B-Y*Q),(Y*B-Z*Q))
      TP(3)=TP(1)-360.
      TP(4)=TP(2)-360.
      RETURN
C     COMPUTES PITCH-YAW TURN
      ENTRY PYTURN
      IF(ABS(B*B+C*C-Y*Y).LT.1.E-7) GO TO 12
      IF((B*B+C*C-Y*Y).LT.0.) GO TO 60
      Q=SQRT(B*B+C*C-Y*Y)
      GO TO 13
12    Q=0.
13    TP(1)=ATAN36((Y*C+B*Q),(Y*B-C*Q))
      TP(2)=ATAN36((Y*C-B*Q),(Y*B+C*Q))
      TP(3)=TP(1)-360.
      TP(4)=TP(2)-360.
      TY(1)=ATAN36((A*Z+X*Q),(X*A-Z*Q))
      TY(2)=ATAN36((A*Z-X*Q),(X*A+Z*Q))
      TY(3)=TY(1)-360.
      TY(4)=TY(2)-360.
      RETURN
10    IF(ITRIG.GE.1) GO TO 70
      WRITE(6,500) T41,T42
      WRITE(6,1000)
      KEY=2
      RETURN
20    IF(ITRIG.GE.1) GO TO 70
      WRITE(6,500) T51,T52
      WRITE(6,1000)
      KEY=2
      RETURN
30    IF(ITRIG.GE.1) GO TO 70
      WRITE(6,500) T61,T62
      WRITE(6,1000)
      KEY=2
      RETURN
```

```
40    IF(ITRIG.GE.1) GO TO 70
      WRITE(6,500) T71,T72
      WRITE(6,1000)
      KEY=2
      RETURN
50    IF(ITRIG.GE.1) GO TO 70
      WRITE(6,500) T81,T82
      WRITE(6,1000)
      KEY=2
      RETURN
60    IF(ITRIG.GE.1) GO TO 70
      WRITE(6,500) T91,T92
      WRITE(6,1000)
      KEY=2
      RETURN
70    IF(TURN1.EQ.1.AND.TURN2.EQ.1) WRITE(6,700) T1,ANG,T41,T42
      IF(TURN1.EQ.1.AND.TURN2.EQ.2) WRITE(6,700) T1,ANG,T51,T52
      IF(TURN1.EQ.1.AND.TURN2.EQ.3) WRITE(6,700) T1,ANG,T61,T62
      IF(TURN1.EQ.1.AND.TURN2.EQ.4) WRITE(6,700) T1,ANG,T71,T72
      IF(TURN1.EQ.1.AND.TURN2.EQ.5) WRITE(6,700) T1,ANG,T81,T82
      IF(TURN1.EQ.1.AND.TURN2.EQ.6) WRITE(6,700) T1,ANG,T91,T92
      IF(TURN1.EQ.2.AND.TURN2.EQ.1) WRITE(6,700) T2,ANG,T41,T42
      IF(TURN1.EQ.2.AND.TURN2.EQ.2) WRITE(6,700) T2,ANG,T51,T52
      IF(TURN1.EQ.2.AND.TURN2.EQ.3) WRITE(6,700) T2,ANG,T61,T62
      IF(TURN1.EQ.2.AND.TURN2.EQ.4) WRITE(6,700) T2,ANG,T71,T72
      IF(TURN1.EQ.2.AND.TURN2.EQ.5) WRITE(6,700) T2,ANG,T81,T82
      IF(TURN1.EQ.2.AND.TURN2.EQ.6) WRITE(6,700) T2,ANG,T91,T92
      IF(TURN1.EQ.3.AND.TURN2.EQ.1) WRITE(6,700) T3,ANG,T41,T42
      IF(TURN1.EQ.3.AND.TURN2.EQ.2) WRITE(6,700) T3,ANG,T51,T52
      IF(TURN1.EQ.3.AND.TURN2.EQ.3) WRITE(6,700) T3,ANG,T61,T62
      IF(TURN1.EQ.3.AND.TURN2.EQ.4) WRITE(6,700) T3,ANG,T71,T72
      IF(TURN1.EQ.3.AND.TURN2.EQ.5) WRITE(6,700) T3,ANG,T81,T82
      IF(TURN1.EQ.3.AND.TURN2.EQ.6) WRITE(6,700) T3,ANG,T91,T92
      WRITE(6,1000)
      KEY=2
      RETURN
500   FORMAT(1H0,-A-,1X,A6,A4,1X,-TURN SEQUENCE CANNOT BE USED TO ACCOMP
     1LISH THE DESIRED MANEUVER-)
700   FORMAT(1H ,-AN INITIAL-,1X,A5,1X,-TURN OF-,1X,F6.1,1X,-DEGREES AND
     1 A-,1X,A6,A4,1X,-TURN SEQUENCE CANNOT BE USED TO ACCOMPLISH THE DE
     2SIRED MANEUVER-)
1000  FORMAT(1H0)
      END
```

D. Description of Subroutine XYZCCL

1. Identification.

Subroutine XYZCCL
FORTRAN V

2. Purpose. This subroutine expresses a vector in terms of clock and cone angles.

3. Restrictions.

(1) COMMON used: COM3, COM4

(2) Subroutines called: ATAN36, ACOS

4. Method. The vector is transformed from the $(\bar{X}, \bar{Y}, \bar{Z})$ coordinate system to the $(\bar{A}, \bar{B}, \bar{C})$ coordinate system. The clock and cone angles are then computed from the following expressions:

$$\text{Clock angle } \alpha = \tan^{-1} \frac{V_B}{V_A}$$

$$\text{Cone angle } \beta = \cos^{-1} V_C$$

where V_A, V_B, and V_C are the vector components expressed in the $(\bar{A}, \bar{B}, \bar{C})$ coordinate system.

5. Use.

(1) Calling sequence: CALL XYZCCL(X, A, B)

(2) Input

Through calling sequence:

X(3)	REAL vector in the $(\bar{X}, \bar{Y}, \bar{Z})$ coordinate system

Through COMMON blocks:

/COM3/

DEG	REAL conversion factor from radians to degrees

/COM4/

TI(3, 3) REAL transformation matrix from ($\overline{X}$, $\overline{Y}$, $\overline{Z}$) to ($\overline{A}$, $\overline{B}$, $\overline{C}$) coordinate systems

(3) Output

Through calling sequence:

A REAL clock angle (deg)
B REAL cone angle (deg)

6. Fortran V Listing of Subroutine XYZCCL.

```
      SUBROUTINE XYZCCL(X,A,B)
C     THIS PROGRAM CONVERTS FROM XYZ SYSTEM TO CONE AND CLOCK ANGLES
      DIMENSION TI(3,3),X(3),Y(3)
      COMMON/COM4/TI/COM3/DEG
      DO 10 I=1,3
10    Y(I)=0.
      DO 20 I=1,3
      DO 20 J=1,3
20    Y(I)=Y(I)+TI(I,J)*X(J)
      A=ATAN36(Y(2),Y(1))
      B=DEG*ACOS(Y(3))
      IF(ABS(Y(3)-1.).LE.1.E-7) B=0.
      IF(ABS(Y(3)).LE.1.E-7) B=90.
      IF(ABS(Y(3)+1.).LE.1.E-7) B=180.
      RETURN
      END
```

E. Description of Subroutine CCLABC

1. Identification.

Subroutine CCLABC
FORTRAN V

2. Purpose. This subroutine expresses vector defined by clock and cone angles in the ($\bar{A}$, $\bar{B}$, $\bar{C}$) coordinate system.

3. Restrictions.

(1) COMMON used: COM3

(2) Subroutines called: SIN, COS

4. Method. The following expressions are used to define the vector in the ($\bar{A}$, $\bar{B}$, $\bar{C}$) coordinate system:

$$V_A = \sin\beta \cos\alpha$$

$$V_B = \sin\beta \sin\alpha$$

$$V_C = \cos\beta$$

where α = clock angle and β = cone angle.

5. Use.

(1) Calling sequence: CALL CCLABC (X, A, B)

(2) Input

Through calling sequence:

A	REAL clock angle (deg)
B	REAL cone angle (deg)

Through COMMON blocks:

/COM3/

DEG	REAL conversion factor from radians to degrees

(3) Output

Through calling sequence:

X(3) REAL vector expressed in ($\bar{A}$, $\bar{B}$, $\bar{C}$) coordinate system

6. Fortran V Listing of Subroutine CCLABC.

```
      SUBROUTINE CCLABC(X,A,B)
C     THIS PROGRAM CONVERTS FROM CONE AND CLOCK ANGLES TO ABC SYSTEM
      DIMENSION X(3)
      COMMON/COM3/DEG
      AA=A/DEG
      BB=B/DEG
      X(1)=SIN(BB)*COS(AA)
      X(2)=SIN(BB)*SIN(AA)
      X(3)=COS(BB)
      RETURN
      END
```

F. Description of Subroutine TMATRX

1. Identification.

Subroutine TMATRX
FORTRAN V

2. Purpose. This subroutine computes the transformation matrices between the ($\bar{X}$, $\bar{Y}$, $\bar{Z}$) and ($\bar{A}$, $\bar{B}$, $\bar{C}$) coordinate systems. It also transforms a vector from the ($\bar{A}$, $\bar{B}$, $\bar{C}$) to the ($\bar{X}$, $\bar{Y}$, $\bar{Z}$) coordinate system.

3. Restrictions.

(1) COMMON used: COM1, COM3, COM4, COM5

(2) Subroutine called: SIN, COS, AINVR

(3) File used: Unit 6 System output file (PRINT$)

(4) Entry points: TRANS

4. Method. Given the clock and cone angles of the pitch and roll axes the transformation matrix from the (A, B, C) to the (X, Y, Z) coordinate systems is constructed. The inverse of this matrix is also found. If the inverse matrix cannot be computed, i.e., the original matrix was singular, the following error message is printed:

TRANSFORMATION MATRIX INVERSE DOES NOT EXIST

If this condition exists a trigger parameter, KEY, is set to 1 and a return to the MAIN program is made. The subroutine AINVR, called by TMATRX to compute the inverse matrix, is a JPL library subroutine. It can be made available upon request or a user supplied subroutine to compute the matrix inverse may be substituted.

Transformation of a vector from the ($\bar{A}$, $\bar{B}$, $\bar{C}$) to the ($\bar{X}$, $\bar{Y}$, $\bar{Z}$) coordinate system is made through the entry point TRANS.

5. Use.

(1) Calling sequences:
```
CALL TMATRX
CALL TRANS(X, Y)
```

(2) Input

Through COMMON blocks:

/COM1/

AXX	REAL pitch axis clock angle (deg)
BXX	REAL pitch axis cone angle (deg)
AZZ	REAL roll axis clock angle (deg)
BZZ	REAL roll axis cone angle (deg)

/COM3/

DEG	REAL conversion factor from radians to degrees

Through calling sequence:

X(3)	REAL vector in ($\bar{A}$, $\bar{B}$, $\bar{C}$) coordinate system

(3) Output

Through files:

UNIT 6 System output file (PRINT$)
Error message

Through COMMON blocks:

/COM4/

TI(3, 3)	REAL transformation matrix from ($\bar{X}$, $\bar{Y}$, $\bar{Z}$) to ($\bar{A}$, $\bar{B}$, $\bar{C}$) coordinate systems

/COM5/

KEY	INTEGER trigger. When set to 1 matrix does not exist

Through calling sequence:

Y(3)	REAL vector in the ($\bar{X}$, $\bar{Y}$, $\bar{Z}$) coordinate system

6. Fortran V Listing of Subroutine TMATRX.

```
      SUBROUTINE TMATRX
C     THIS PROGRAM COMPUTES THE TRANSFORMATION MATRIX
      COMMON/COM1/AXX,BXX,AZZ,BZZ/COM3/DEG/COM4/TI/COM5/KEY
      DIMENSION T(3,3),X(3),Y(3),TI(3,3),WORK(12)
      AX=AXX/DEG
      BX=BXX/DEG
      AZ=AZZ/DEG
      BZ=BZZ/DEG
      T(1,1)=SIN(BX)*COS(AX)
      T(1,2)=SIN(BX)*SIN(AX)
      T(1,3)=COS(BX)
      T(2,1)=SIN(BZ)*SIN(AZ)*COS(BX)-SIN(BX)*SIN(AX)*COS(BZ)
      T(2,2)=COS(BZ)*SIN(BX)*COS(AX)-COS(BX)*SIN(BZ)*COS(AZ)
      T(2,3)=SIN(BZ)*COS(AZ)*SIN(BX)*SIN(AX)-SIN(BX)*COS(AX)*SIN(BZ)*
     1SIN(AZ)
      T(3,1)=SIN(BZ)*COS(AZ)
      T(3,2)=SIN(BZ)*SIN(AZ)
      T(3,3)=COS(BZ)
      DO 5 I=1,3
      DO 5 J=1,3
5     TI(I,J)=T(I,J)
C     COMPUTES INVERSE OF TRANSFORMATION MATRIX
      CALL AINVR(TI,3,3,$6,WORK)
      GO TO 7
6     WRITE(6,100)
      KEY=1
7     RETURN
C     TRANSFORMS FROM ABC TO XYZ COORDINATE SYSTEM
      ENTRY TRANS(X,Y)
      DO 10 I=1,3
10    Y(I)=0.
      DO 20 I=1,3
      DO 20 J=1,3
20    Y(I)=Y(I)+T(I,J)*X(J)
      RETURN
100   FORMAT(1H0,-TRANSFORMATION MATRIX INVERSE DOES NOT EXIST-)
      END
```

G. Description of Subroutine YTURN

1. Identification.

Subroutine YTURN
FORTRAN V

2. Purpose. This subroutine maps a vector through a pitch, yaw, or roll turn.

3. Restrictions.

(1) COMMON used: COM3

(2) Subroutines called: SIN, COS

(3) Entry points: PTURN, RTURN

4. Method. The appropriate coordinate rotation matrix and its transpose are computed and they are multiplied by the input vector to generate the vector in the rotated coordinate system.

5. Use.

(1) Calling sequence:

```
CALL YTURN (X, Y, YT, YA)
CALL PTURN (X, Y, YT, PA)
CALL RTURN (X, Y, YT, RA)
```

(2) Input

Through COMMON blocks:

/COM3/	REAL conversion factor from radians to degrees

Through calling sequence:

PA	REAL pitch rotation angle (deg)
RA	REAL roll rotation angle (deg)
X(3)	REAL vector before the turn
YA	REAL yaw rotation angle (deg)

(3) Output

Through calling sequence:

Y(3)	REAL vector after the turn formed from product of input vector and rotation matrix

YT(3) REAL vector after the turn formed from product of input vector and transpose of rotation matrix

6. Fortran V Listing of Subroutine YTURN.

```
      SUBROUTINE YTURN(X,Y,YT,YA)
C     THIS PROGRAM COMPUTES YAW,ROLL AND PITCH TURNS
C     COMPUTES YAW TURN
      DIMENSION X(3),Y(3),YT(3),YM(3,3),RM(3,3),PM(3,3),YMT(3,3),RMT(3,3
     1),PMT(3,3)
      COMMON/COM3/DEG
      DATA YM,RM,PM,YMT,RMT,PMT/54*0./
      D=YA/DEG
      YM(1,1)=COS(D)
      YM(1,3)=-SIN(D)
      YM(2,2)=1.
      YM(3,1)=SIN(D)
      YM(3,3)=COS(D)
      YMT(1,1)=YM(1,1)
      YMT(1,3)=-YM(1,3)
      YMT(2,2)=YM(2,2)
      YMT(3,1)=-YM(3,1)
      YMT(3,3)=YM(3,3)
      DO 10 I=1,3
      YT(I)=0.
10    Y(I)=0.
      DO 20 I=1,3
      DO 20 J=1,3
      YT(I)=YT(I)+YMT(I,J)*X(J)
20    Y(I)=Y(I)+YM(I,J)*X(J)
      RETURN
C     COMPUTES PITCH TURN
      ENTRY PTURN(X,Y,YT,PA)
      D=PA/DEG
      PM(1,1)=1.
      PM(2,2)=COS(D)
      PM(2,3)=SIN(D)
      PM(3,2)=-SIN(D)
      PM(3,3)=COS(D)
      PMT(1,1)=PM(1,1)
      PMT(2,2)=PM(2,2)
      PMT(2,3)=-PM(2,3)
      PMT(3,2)=-PM(3,2)
      PMT(3,3)=PM(3,3)
      DO 30 I=1,3
      YT(I)=0.
30    Y(I)=0.
      DO 40 I=1,3
      DO 40 J=1,3
      YT(I)=YT(I)+PMT(I,J)*X(J)
40    Y(I)=Y(I)+PM(I,J)*X(J)
      RETURN
C     COMPUTES ROLL TURN
      ENTRY RTURN(X,Y,YT,RA)
      D=RA/DEG
      RM(1,1)=COS(D)
      RM(1,2)=SIN(D)
      RM(2,1)=-SIN(D)
      RM(2,2)=COS(D)
      RM(3,3)=1.
      RMT(1,1)=RM(1,1)
      RMT(1,2)=-RM(1,2)
      RMT(2,1)=-RM(2,1)
      RMT(2,2)=RM(2,2)
      RMT(3,3)=RM(3,3)
      DO 50 I=1,3
      YT(I)=0.
50    Y(I)=0.
      DO 60 I=1,3
      DO 60 J=1,3
      YT(I)=YT(I)+RMT(I,J)*X(J)
60    Y(I)=Y(I)+RM(I,J)*X(J)
      RETURN
      END
```

H. Description of Subroutine ATAN36

1. Identification.

Function ATAN36
FORTRAN V

2. Purpose. This function routine computes the four quadrant arc tangent.

3. Restriction.

Subroutines called: ATAN

4. Method. Angle = $\tan^{-1}$ (X/Y), $0° \leq$ angle $< 360°$; if either |X| or |Y| are within 0.000001 of 0, then these values are set equal to zero respectively.

5. Use.

(1) Calling sequence: Angle = ATAN36(X, Y)

(2) Input

Through calling sequence:

X	REAL sine of the angle
Y	REAL cosine of the angle

(3) Output

Through the function name ATAN36

6. Fortran V Listing of Function ATAN36.

```
      FUNCTION ATAN36(X,Y)
      J=1
      K=1
      IF((ABS(X)-1.E-6).LE.0.) J=0
      IF((ABS(Y)-1.E-6).LE.0.) K=0
      IF(J.EQ.0.AND.K.EQ.0) GO TO 10
      IF(J.EQ.0) GO TO 20
      IF(K.EQ.0) GO TO 30
      Z=ABS(X)/X
      ATAN36=90.*(2.-Z)-57.295779*ATAN(Y/X)
      RETURN
10    X=0.
      Y=0.
      ATAN36=0.
      RETURN
20    X=0.
      Z=ABS(Y)/Y
      ATAN36=90.-90.*Z
      RETURN
30    Y=0.
      Z=ABS(X)/X
      ATAN36=180.-90.*Z
      RETURN
      END
```

I. Description of Subroutine YAWTRN

1. Identification.

Subroutine YAWTRN
FORTRAN V

2. Purpose. This subroutine maps a vector through a pitch, yaw, or roll turn by incremental angular steps.

3. Restrictions.

(1) COMMON used: COM11

(2) Subroutines called: YTURN, XYZCCL, RTURN, PTURN

(3) Files used: Unit 6 System output file (PRINT$)

(4) Entry points: ROLTRN, PITTRN

4. Method. The turn magnitude is divided into small increments of size DELANG and the vector is thus progressively stepped through the turn. The intermediate position of the vector after each step is printed.

5. Use.

(1) Calling sequence:

```
CALL YAWTRN (RV1, RV2, YAWANG)
CALL ROLTRN (RV1, RV2, ROLANG)
CALL PITTRN (RV1, RV2, PITANG)
```

(2) Input

Through COMMON blocks:

/COM11/

DELANG	REAL increment by which the vector is stepped through a turn

Through calling sequence:

PITANG	REAL pitch turn magnitude (deg)
ROLANG	REAL roll turn magnitude (deg)
RV1(3)	REAL vector at the start of the turn (deg)
RV2(3)	REAL vector at the end of the turn (deg)
YAWANG	REAL yaw turn magnitude (deg)

(3) Output

Through files:

Unit 6 System output file (PRINT$)

A	REAL clock angle of vector (deg)
ANG	REAL turn magnitude (deg)
B	REAL cone angle of vector (deg)

6. Fortran V Listing of Subroutine YAWTRN.

```
      SUBROUTINE YAWTRN(RV1,RV2,YAWANG)
C     THIS PROGRAM MAPS A VECTOR THROUGH A SPECIFIED TURN
      DIMENSION RV1(3),RV2(3),RVX(3),Q(3)
      COMMON/COM11/DELANG
      CALL XYZCCL(RV1,A,B)
      KSTOP=0
      ANG=0.
      WRITE(6,1000) ANG,A,B
5     ANG=ANG+DELANG*YAWANG/ABS(YAWANG)
      IF(ABS(ANG)-ABS(YAWANG)) 20,10,30
10    KSTOP=1
20    CALL YTURN(RV1,RVX,Q,ANG)
      CALL XYZCCL(RVX,A,B)
      WRITE(6,1000) ANG,A,B
      IF(KSTOP.EQ.1) RETURN
      GO TO 5
30    CALL XYZCCL(RV2,A,B)
      WRITE(6,1000) YAWANG,A,B
      RETURN
      ENTRY ROLTRN(RV1,RV2,ROLANG)
      CALL XYZCCL(RV1,A,B)
      KSTOP=0
      ANG=0.
      WRITE(6,1000) ANG,A,B
35    ANG=ANG+DELANG*ROLANG/ABS(ROLANG)
      IF(ABS(ANG)-ABS(ROLANG)) 50,40,60
40    KSTOP=1
50    CALL RTURN(RV1,RVX,Q,ANG)
      CALL XYZCCL(RVX,A,B)
      WRITE(6,1000) ANG,A,B
      IF(KSTOP.EQ.1) RETURN
      GO TO 35
60    CALL XYZCCL(RV2,A,B)
      WRITE(6,1000) ROLANG,A,B
      RETURN
      ENTRY PITTRN(RV1,RV2,PITANG)
      CALL XYZCCL(RV1,A,B)
      KSTOP=0
      ANG=0.
      WRITE(6,1000) ANG,A,B
65    ANG=ANG+DELANG*PITANG/ABS(PITANG)
      IF(ABS(ANG)-ABS(PITANG)) 80,70,90
70    KSTOP=1
80    CALL PTURN(RV1,RVX,Q,ANG)
      CALL XYZCCL(RVX,A,B)
      WRITE(6,1000) ANG,A,B
      IF(KSTOP.EQ.1) RETURN
      GO TO 65
90    CALL XYZCCL(RV2,A,B)
      WRITE(6,1000) PITANG,A,B
      RETURN
1000  FORMAT(1H ,3F15.2)
      END
```

J. Description of Subroutine DETAIL

1. Identification.

Subroutine DETAIL
FORTRAN V

2. Purpose. This subroutine generates a detailed mapping of the reference vector through each maneuver turn.

3. Restrictions.

(1) COMMON used: COM2, COM6, COM7, COM9, COM12

(2) Subroutines called: YAWTURN, ROLTRN, PITTRN

(3) Files used: Unit 6 System output file (PRINT$)

4. Method. Calling sequences to the appropriate subroutines are made to generate a mapping of the reference vector through each turn of the maneuver. Any two- or three-turn maneuver can be handled.

5. Use.

(1) Calling sequence: CALL DETAIL

(2) Input:

Through COMMON blocks:

/COM2/

TP(4)	REAL pitch turn magnitudes (deg)
TR(4)	REAL roll turn magnitudes (deg)
TY(4)	REAL yaw turn magnitudes (deg)

/COM6/

RV11(3), RV12(3)	REAL reference vector after the first turn
RV21(3), RV22(3)	REAL reference vector after the second turn

/COM7/

TURN1	INTEGER trigger designating first turn of three turn sequence
TURN2	INTEGER trigger designating second and third turns

/COM9/

NUMTRN	INTEGER trigger representing turn size. NUMTRN = 2 means two turn sequence; NUMTRN = 3 means three turn sequence.
RV(3)	REAL reference vector at start of two turn maneuver
TURN	INTEGER trigger designating the turn maneuver

/COM12/

RSAVE(3)	REAL reference vector before the maneuver

(3) Output

Through files:

Unit 6 System output file (PRINT$)
Column headings

6. Fortran V Listing of Subroutine DETAIL.

```
      SUBROUTINE DETAIL
C     THIS PROGRAM GENERATES DETAILED MAPPING OUTPUT
      DIMENSION RV(3),RV11(3),RV12(3),RV21(3),RV22(3),TR(4),TY(4),TP(4),
     1V(3),VT(3),RSAVE(3),RV1(3)
      COMMON/COM6/RV11,RV12,RV21,RV22,A11,A12,A21,A22,B11,B12,B21,B22
      COMMON/COM9/NUMTRN,TURN,RV/COM2/V,VT,TY,TR,TP
      COMMON/COM7/ITRIG,ANG,TURN1,TURN2,T1,T2,T3/COM12/RSAVE,RV1
      INTEGER TURN,TURN1,TURN2
20    WRITE(6,800)
      WRITE(6,900)
      WRITE(6,800)
      IF(NUMTRN.EQ.3) GO TO 100
25    GO TO(30,40,50,60,70,80),TURN
30    DO 35 I=1,3,2
      WRITE(6,1000) T3,T3
      CALL YAWTRN(RV,RV11,TY(I))
      WRITE(6,1000) T1,T1
      CALL ROLTRN(RV11,RV21,TR(I))
      WRITE(6,1000) T3,T3
      CALL YAWTRN(RV,RV12,TY(I+1))
      WRITE(6,1000) T1,T1
35    CALL ROLTRN(RV12,RV22,TR(I+1))
      RETURN
40    DO 45 I=1,3,2
      WRITE(6,1000) T1,T1
      CALL ROLTRN(RV,RV11,TR(I))
      WRITE(6,1000) T3,T3
      CALL YAWTRN(RV11,RV21,TY(I))
      WRITE(6,1000) T1,T1
      CALL ROLTRN(RV,RV12,TR(I+1))
      WRITE(6,1000) T3,T3
45    CALL YAWTRN(RV12,RV22,TY(I+1))
      RETURN
50    DO 55 I=1,3,2
      WRITE(6,1000) T2,T2
      CALL PITTRN(RV,RV11,TP(I))
      WRITE(6,1000) T1,T1
      CALL ROLTRN(RV11,RV21,TR(I))
      WRITE(6,1000) T2,T2
      CALL PITTRN(RV,RV12,TP(I+1))
      WRITE(6,1000) T1,T1
55    CALL ROLTRN(RV12,RV22,TR(I+1))
      RETURN
60    DO 65 I=1,3,2
      WRITE(6,1000) T1,T1
      CALL ROLTRN(RV,RV11,TR(I))
      WRITE(6,1000) T2,T2
      CALL PITTRN(RV11,RV21,TP(I))
      WRITE(6,1000) T1,T1
      CALL ROLTRN(RV,RV12,TR(I+1))
      WRITE(6,1000) T2,T2
65    CALL PITTRN(RV12,RV22,TP(I+1))
      RETURN
70    DO 75 I=1,3,2
      WRITE(6,1000) T3,T3
      CALL YAWTRN(RV,RV11,TY(I))
      WRITE(6,1000) T2,T2
      CALL PITTRN(RV11,RV21,TP(I))
      WRITE(6,1000) T3,T3
      CALL YAWTRN(RV,RV12,TY(I+1))
      WRITE(6,1000) T2,T2
```

```
75    CALL PITTRN(RV12,RV22,TP(I+1))
      RETURN
80    DO 85 I=1,3,2
      WRITE(6,1000) T2,T2
      CALL PITTRN(RV,RV11,TP(I))
      WRITE(6,1000) T3,T3
      CALL YAWTRN(RV11,RV21,TY(I))
      WRITE(6,1000) T2,T2
      CALL PITTRN(RV,RV12,TP(I+1))
      WRITE(6,1000) T3,T3
85    CALL YAWTRN(RV12,RV22,TY(I+1))
      RETURN
100   GO TO(110,120,130),TURN1
110   WRITE(6,1000) T1,T1
      CALL ROLTRN(RSAVE,RV1,ANG)
112   DO 115 I=1,3
115   RV(I)=RV1(I)
      GO TO 25
120   WRITE(6,1000) T2,T2
      CALL PITTRN(RSAVE,RV1,ANG)
      GO TO 112
130   WRITE(6,1000) T3,T3
      CALL YAWTRN(RSAVE,RV1,ANG)
      GO TO 112
800   FORMAT(1H0)
900   FORMAT(1H0,20X,-DETAIL MAPPING OF REFERENCE VECTOR THROUGH THE MAN
     1EUVER TURNS-)
1000  FORMAT(1H0,23X,A5,1X,-TURN-/1H ,10X,A5,10X,-CLOCK-,10X,-CONE-/1H ,
     13(10X,-ANGLE-)/)
      END
```


NASA — JPL — Coml, L.A., Calif.