CT- 82-100-101LR

FAA WJH Technical Center
00092667

FAA TECHNICAL CENTER LETTER REPORT

[redacted]

MAR 22 Rec'd

TECHNICAL CENTER LIBRARY
ATLANTIC CITY, N.J. 08405

ATADS SOFTWARE SIMULATION

by
Charles Dudas

U. S. DEPARTMENT OF TRANSPORTATION
FEDERAL AVIATION ADMINISTRATION
TECHNICAL CENTER
Atlantic City Airport, N.J. 08405

ABSTRACT

This letter report documents a simulation designed to show the effects of DME beacon position and vehicle dynamics on ATADS computed position error. The two-month co-op period proved insufficient to complete this task. A program flow chart, program listing, and details required to complete this task are contained in this report.

TABLE OF CONTENTS

Abstract	i
Introduction	1
Purpose	1
Background	1
Discussion	1
Requirements for Completion	2
Appendix A - Program Flow Charts	A-1 to A-2
Appendix B - Data Base Program Listing	B-1 to B-2
Appendix C - Beacon Simulation Program Listing	C-1 to C-14

INTRODUCTION

PURPOSE

This simulation was developed to quantify the effects of beacon location and vehicle dynamics on ATADS computed position error.

BACKGROUND

A multi-positioning system, ATADS, was developed as a position reference system for navigation receiver evaluation and testing. ATADS uses a program called RAPPRO to analyze range data. A modified version of RAPPRO forms the core of this simulation. ATADS, along with the RAPPRO program is further described in reference 1.

DISCUSSION

This simulation is a combination of two programs. The first program, called FILFIL, forms the data base for the main program, call BCNSIM. A program flowchart for each of the programs can be found in appendix A.

Program FILFIL creates a data base based on the replies to a series of user prompts. The data base program consists of two subroutines which can be accessed together or separately.

The first subroutine prompts the user for a file name and then loops through code which requests the user to input the simulated user position values in latitude, longitude, and altitude. After the position data acquisition is completed, the program then prompts the user for program termination. If not terminated, a prompt is issued requesting another file name for the DME beacon coordinate library file. Here, another loop is encountered wherein the latitude, longitude, and altitude of a series of beacon placements are entered by the user. Also, a sequential identifier is issued to each beacon for later reference purposes. The program then asks if more range data files or beacon library files (BLF) are to be entered. If so, the process repeats. If not, the program terminates. FILFIL does not require any other parameters or additional files, therefore, the user has maximum control of the simulated data at the run-time of program BCNSIM.

Program BCNSIM begins by a call to subroutine QUERY. QUERY prompts the user for the input and output file names along with other parameters. The BLF acts as a general guide for the user. The user may select any number of beacons from the BLF and/or may choose to use any number of non-standard beacons. If an error occurs due to a read format error, subroutine ERRR will echo the problem and prompt the user with instructions.

Once the file information and other parameters are set in QUERY, subroutines ECEF and ENU will convert the beacons into the ENU coordinates. ENU (East-North-Up) is a local coordinate system wherein a predetermined position is designated the origin. In this case, the software sequentially cycles through the beacons and collects range values from different beacons each of which, at the time of collection, is considered an ENU origin.

Next, subroutine RAPPS reads the RAPPS data file. Subroutine TMETAG will associate a four-digit integer, calibrated in seconds, to each entry in the range file. The TMETAG begins at 0000 seconds for each simulation run. The value of TMETAG is dependent on the speed of the aircraft and the distance it travels from one range entry to the next. Contained in this routine, is a subroutine call to compute distances from differences in latitude, longitude, and altitude. Subroutine STIME allows the user to process any portion of the total simulated flight. The user inputs a start time and stop time. It also readies the first window based on the previously acquired (from QUERY) width of window value.

Subroutine RAPPS computes the user position at the center of the current window. Each window allows the user positions from various beacons, occuring at different times to be extrapolated/interpolated to a common specified time. This common time is chosen to be at the center of the current data window. This is necessary to employ the least squares solution, which assumes simultaneous measurement data.

Next, subroutine HDOP computes the HDOP of the extrapolated range value with the series of beacon positions defined as a matrix.

Subroutine INCRMT slides the window forward, removes old data, and goes back and processes the last data which was read, but not used.

After the range data has been exhausted and processing completed, a prompt is issued. This prompt asks if the user would like to continue and, if so, starts the process over with prompts for input and output file names. If the user so specifies, the program terminates.

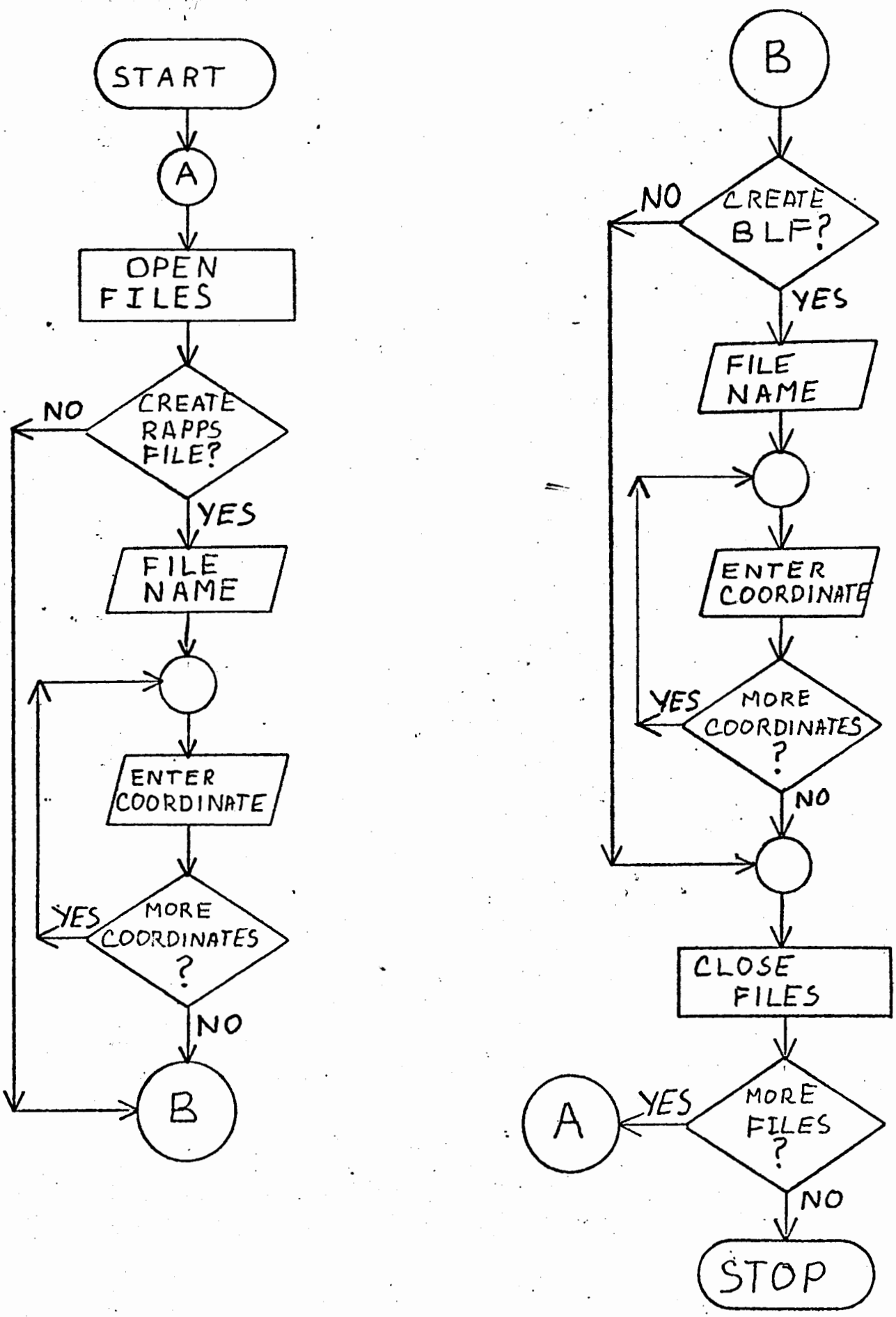
REQUIREMENTS FOR COMPLETION

As the program stands now, subroutine TMETAG needs to be completed. Since only key values of range position are given (i.e., direction changes), intermediate range positions are needed to complete each window's data. Based on the speed, the aircraft will travel a certain distance during the one second interval when the software switches from one beacon to the next. The new range position is the sum of the previous position and the distance travelled in one second. Each range position needs to be converted to the ENU coordinate system with the operating beacon as the local origin. Besides subroutine TMETAG, all logic has been completed. Only typical subroutine interfaces (i.e., common statements, passing parameters) need to be handled because of TMETAG.

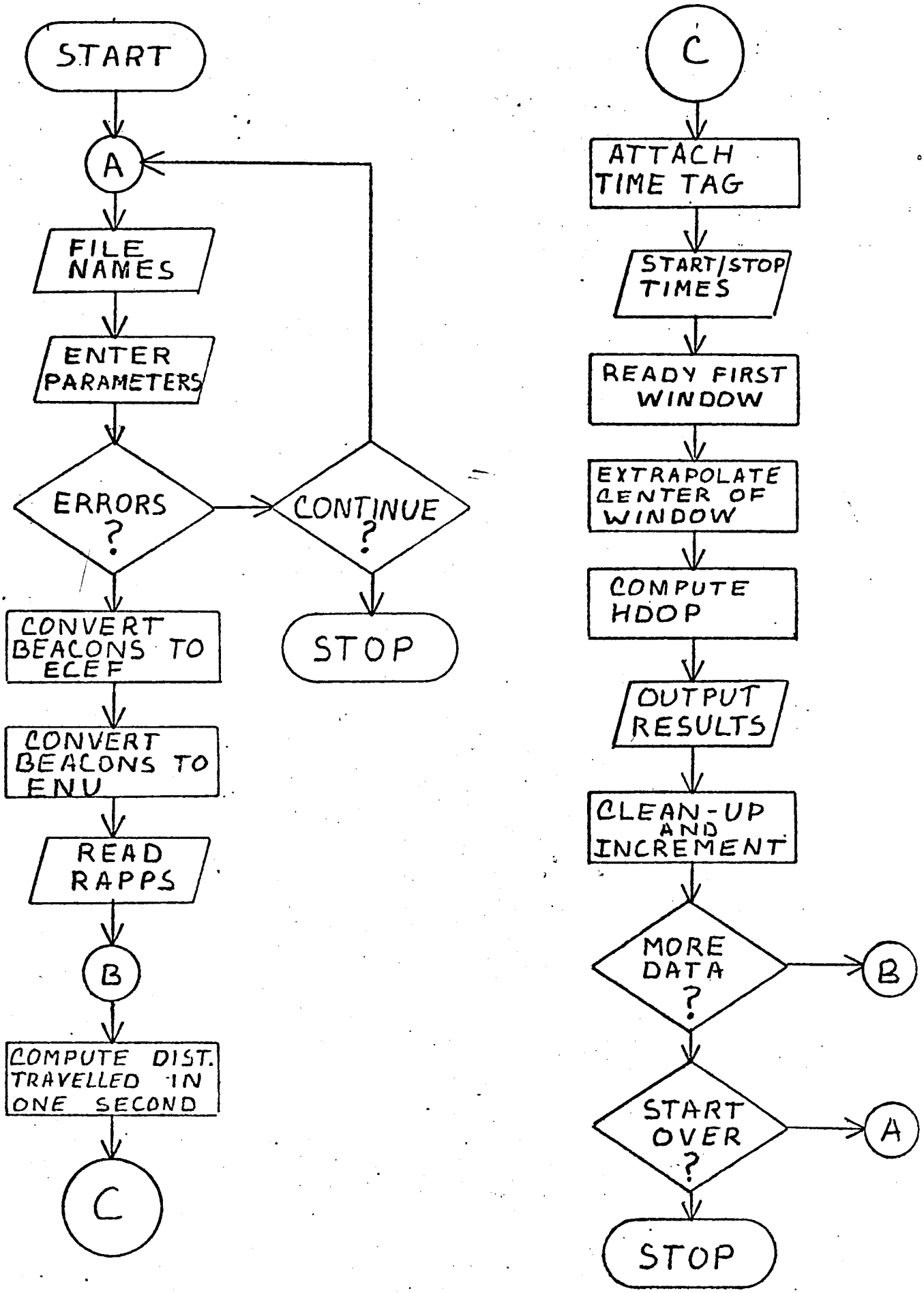
APPENDIX A

PROGRAM FLOW CHARTS

PROGRAM FILFIL



PROGRAM BCNSIM



APPENDIX B

FILFIL PROGRAM LISTING

```
C
C
C
C   ***********************************************************
C    PROGRAM FILFIL
C    THIS PROGRAM IS DESIGNED TO CREATE TWO INPUT DATA FILES
C    FOR PROGRAM BCNSIM. THE USER HAS AN OPTION OF CREATING
C    ONE OR MORE OF HIS CHOICE OF EITHER A RAPPRO FILE OR A
C    BEACON LIBRARY FILE.
C
C    ***************************************************************
      WRITE(5,10)
   10  FORMAT(1H ,'DO YOU WANT TO CREATE A RAPPS FILE?(NO=0,YES=1)'/)
      READ (5,20) ICREAT
   20 FORMAT(I1)
      IF(ICREAT.EQ.0)GO TO 145
   25 CONTINUE
      WRITE (5,30)
   30 FORMAT(1H ,'ENTER RAPPS FILE NAME.'/)
      CALL ASSIGN(1,,-1,'NEW')
   40 CONTINUE
      CALL QUEST(DLAT,MLAT,SLAT,DLON,MLON,SLON,ALT)
      WRITE(1,120)DLAT,MLAT,SLAT,DLON,MLON,SLON,ALT
  120 FORMAT(3I2,1X,3I2,1X,I5)
      WRITE(5,130)
  130 FORMAT(H ,'ANY MORE COORDINATES?(Y=1,N=0)'/)
      READ(5,140)IMORE
  140 FORMAT(I1)
      IF(IMORE.EQ.1)GO TO 40
  145 CONTINUE
      WRITE(5,150)
  150 FORMAT(1H ,' DO YOU WANT TO CREATE A BLF?(Y=1,N=0)'/)
      READ(5,160) ICREAT
      IF(ICREAT.EQ.0)GO TO 215
  160 FORMAT(I1)
  165 CONTINUE
      WRITE(5,170)
  170 FORMAT(1H ,'ENTER BLF NAME.'/)
      CALL ASSIGN(4,,-1,'NEW')
  180 CONTINUE
      CALL QUEST(DLAT,MLAT,SLAT,DLON,MLON,SLON,ALT)
      WRITE(4,190)DLAT,MLAT,SLAT,DLON,MLON,SLON,ALT
  190 FORMAT(3I2,1X,3I2,1X,I5)
      WRITE(5,200)
  200 FORMAT(1H ,'ANY MORE COORDINATES?(Y=1,N=0)'/)
```

```
      READ(5,210)IMORE
  210 FORMAT(I1)
      IF(IMORE.EQ.1) GO TO 180
      CALL CLOSE(1)
      CALL CLOSE(4)
      WRITE(5,220)
  220 FORMAT(1H ,'DO YOU WANT TO CREATE MORE FILES?(N=0,RAPPS=1
     1,BLF=2)'/)
      READ(5,230) IMORE
  230 FORMAT(I1)
      IF(IMORE.EQ.1)GO TO 25
      IF(IMORE.EQ.2)GO TO 165
      STOP
      END
C
C   ***********************************************************
C
C   SUBROUTINE QUEST ASKS THE USER TO TYPE IN THE LATITUDE,
C   LONGITUDE, AND ALTITUDE OF EACH RAPPS OR BEACON
C   POSITION.
C
C   ***********************************************************
      SUBROUTINE QUEST(DLAT,MLAT,SLAT,DLON,MLON,SLON,ALT)
      WRITE(5,50)
   50 FORMAT(1H ,'ENTER LATITUDE.(DD,MM,SS)'/)
      READ(5,60)DLAT,MLAT,SLAT
   60 FORMAT(3(I2,1X)
      WRITE(5,80)
   80 FORMAT(1H ,'ENTER LONGITUDE.(DD,MM,SS)'/)
      READ(5,90)DLON,MLON,SLON
   90 FORMAT(3(I2,1X))
      WRITTE(5,100)
  100 FORMAT(IH ,'ENTER ALTITUDE IN FEET(FFFFF)'/)
      READ(5,110)ALT
  110 FORMAT(I5)
      RETURN
      END
```


FAA WJH Technical Center
00092667

APPENDIX C

BCNSIM PROGRAM LISTING

```
      BLOCK DATA
      IMPLICIT REAL*8(A-H,O-Z)
C   I,J,K,L,M,N: ARE INTEGERS
      COMMON/GMATX/ RTSEG,INDX,GX(20),GY(20),GZ(20)
      COMMON/HDOPP/ X(20),Y(20),Z(20)
      COMMON/DATS/KBEAC,DTR,A,B,KCOOR
      COMMON/PRE/IDENT(10),SLAT(10),SLON(10),IALT(10)
      COMMON/REF/REFALT,REFLON,REFLAT
      COMMON/ARRYS/CLALON(10,3),CECEF(10,3)
      COMMON/TIMES/T0,T1,T2,START,STOP
      COMMON/SLDE/IS1,IM1,IH1,WIDTH,INCTIM
      DATA GX,GY,GZ,ENTRY,CECEF,X,Y,Z,RTSEG/361*0.D0/
      DATA INDX,KWYPNT,KBEAC/3*0/
      END
C   ***************************************************************
C
C
C
C
C      PROGRAM NAME:        RAPPRO-THIS IS A MODIFIED VERSION CONSISTING
C                                  OF SIMULATED DATA WITH NO ERRORS.
C                                  BOTH THE BEACON MATRIX COMPONENTS AND
C                                  THE RANGE VALUE MATRIX ARE USER
C                                  GENERATED OR USER MODIFIED.
C
C
C      SUBROUTINES:         GMTRA,GMPRD FROM SSPLIB
C
C
C      DATA FILES:
C                           1-INPUT RAPPS DATA CREATED BY RAPFIL
C                           2-OUTPUT POSITION DATA
C                           3-INPUT FLIGHT DATA CONTROL FILE
C                           4-INPUT DME COORDINATE LIBRARY
C
C
C  PROGRAM BCNSIM-
C  THIS PROGRAM IS A DYNAMIC SIMULATION OF AN AIRCRAFT IN FLIGHT
C  BEING TRACKED BY THE ATADS SYSTEM. THE AIRCRAFT FOLLOWS A
C  PREDETERMINED SERIES OF WAYPOINTS IN ITS APPROACH TO A RUN-
C  WAY. PLACED NEAR THE RUNWAY ARE SEVERAL BEACONS WHICH POSITION
C  THE AIRCRAFT. BOTH THE WAYPOINT MATRIX AND THE BEACON MATRIX
C  ARE USER SET. THE NUMBER OF WAYPOINTS, THE NUMBER OF BEACONS, AND
C  THE SIZE OF A ROUTE SEGMENT ARE ALSO USER SET. THE CREATION OF A
C  DATA FILE IS NECCESSARY AND IS EXPLAINED IN THE PROGAM
C  DOCUMENTATION.
```

```
C
C
C
C
C
C   ****************************************************************
C
C
      IMPLICIT REAL*8 (A-H,O-T,V-Z)
      COMMON/GMATX/ RTSEG,INDX,GX(20),GY(20),GZ(20)
      COMMON/HDOPP/ X(20),Y(20),Z(20)
      COMMON/DATS/KBEAC,DTR,A,B,KCOOR
      COMMON/PRE/IDENT(10),SLAT(10),SLON(10),IALT(10)
      COMMON/REF/REFALT,REFLON,REFLAT
      COMMON/ARRYS/CLALON(10,3),CECEF(10,3)
      COMMON/TIMES/T0,T1,T2,START,STOP,WIDTH
      COMMON/SLDE/IS1,IM1,IH1,WIDTH,INCTIM
      REAL*8 U0
      DIMENSION T9(10),S(10),D6(10),XM2(10)
      DIMENSION Q5(3,3)
      DIMENSION RANGE1(10,50),RTIME(10,50),ISTAG(10)
      DIMENSION IRCOUN(10),R(10),DELRT(10,50),CECEF(11,3),CLOCAL(11,3)
      DIMENSION CLALON(11,3)
      INTEGER Q(10)
      DIMENSION D9(10)
      DIMENSION Q1(3,3),P03(3),X03(3)
      DATA X1,Y1,Z1/3*0.D0/
      DATA T9,S,D6/30*0.D0/
      DATA RANGE1,RTIME,DELRT/1500*0.D0/
      DATA Q,ISTAG/20*0/
      DATA D9/10*0.D0/
      DATA IRCOUN/10*0/
      DATA INCTIM,WIDTH/5,12.D0/
      DFLOAT(I)=DBLE(FLOAT(I))
      TIME(IH,IM,IS,IMS)=DFLOAT(IH)*3600.D0+DFLOAT(IM)*60.D0+
     1DFLOAT(IS)+DFLOAT(IMS)/1000.D0
      DIST(X,Y,Z,X0,Y0,Z0)=DSQRT((X-X0)**2+(Y-Y0)**2+(Z-Z0)**2)
      DMSDEG(ID,IM,US)=DFLOAT(ID)+DFLOAT(IM)/60.D0+DBLE(US)/3600.D0
      DTR=3.141592654D0/180.D0
      A=3443.949D0
      B=3432.274D0
      G3=A
      G4=B
      CALL QUERY
  110 CONTINUE
```

```
      Z0=REFALT
      KCOUNT=0
      READ(4,120,ERR=156,END=150)IDENT,SLAT,SLON,IALT
  120 FORMAT(1X,I1,2D18.9,I7)
      WRITE(6,141)IDENT,SLAT,SLON,IALT
  141 FORMAT(1H ,I5,2D18.9,I7)
      INDEX=J
      CLALON(INDEX,1)=SLAT*DTR
      CLALON(INDEX,2)=SLON*DTR
      CLALON(INDEX,3)=DFLOAT(IALT)/6076.1155D0
      INDEX=INDEX+1
      KCOUNT=KCOUNT+1
      GO TO 110
  150 CONTINUE
      CALL ECEF
      CCALL ENU
      CALL RAFPS
  130 CONTINUE
      CALL TMETAG
      CALL STIMES
      CALL  HDOP
      CALL INCRMNT
      GO TO 130
      CALL CLOSE(1)
      CALL CLOSE(2)
      CALL CLOSE(3)
      CALL CLOSE(4)
      STOP
      END
C     ******************************************************
C
C     SUBROUTINE ERRR PROCESSES THE TYPE OF ERROR ENCOUNTERED.
C
C     ******************************************************
      SUBROUTINE ERRR
      COMMON/DATS/KBEAC,DTR,A,B,KCOOR
      IF(KCOUNT.EQ.KBEAC)GO TO 160
      WRITE(6,155)
  155 FORMAT(1H ,'PROBLEM IN NUMBER OF STATIONS IDENTIFIED !!'/)
      CALL CLOSE(1)
      CALL CLOSE(2)
      CALL CLOSE(3)
      CALL CLOSE(4)
      STOP
  156 CONTINUE
```

```
      WRITE(6,157)
  157 FORMAT(1H ,'READ ERROR IN STATION DATA FILE !!')
      CALL CLOSE(1)
      CALL CLOSE(2)
      CALL CLOSE(3)
      CALL CLOSE(4)
      STOP
  160 CONTINUE
      RETURN
      END
C    ***************************************************************
C
C     SUBROUTINE QUERY FILLS THE DATA FILES AND SETS OTHER
C     PARAMETERS TO USER SPECIFICATIONS.
C
C     ***********************************************
      SUBROUTINE QUERY
      COMMON/PRE/ IDENT(10),SLAT(10),SLON(10),IALT(10)
      WRITE(5,10)
   10 FORMAT(1H ,'ENTER RAPPS INPUT FILE NAME .'/)
      CALL ASSIGN(1,,-1,'RDO')
      WRITE(5,20)
   20 FORMAT(1H ,'ENTER POSITION OUTPUT FILE .'/)
      CALL ASSIGN(2,,-1,'NEW')
      WRITE(5,30)
   30 FORMAT(1H ,' ENTER BEACON LIBRARY FILE NAME .'/)
      CALL ASSIGN(4,,-1,'RDO')
      WRITE(5,40)
   40 FORMAT(1H ,20X,' BEACON LIBRARY FILE .')
   50 CONTINUE
      I=I+1
      READ(4,60,ERR=156,END=100) IDENT(I),SLAT(I),SLON(I),IALT(I)
   60 FORMAT(1X,I5,1X,2(D18.9,1X),I7)
      I2=I
      WRITE(5,70) IDENT(I),SLAT(I),SLON(I),IALT(I)
   70 FORMAT(1H,I5,1X,2(D18.9,1X),I7)
      GO TO 50
      DO 150 J=1,10
         WRITE(5,80) J
   80    FORMAT(1H ,' INPUT # ',I2', BEACON IDENTIFIER IS?
     1   (INPUT 3-DIGIT INTEGER IDENTIFIER FROM BLF.)'/)
         READ(5,90) IBEAC
   90    FORMAT(I3)
           DO 100 K=1,10
             IND=K
```

```
          IF(IBEAC.EQ.IDENT(K)) GO TO 110
  100     CONTINUE
  110     CONTINUE
          WRITE(6,120) IDEENT(IND),SLAT(IND),SLON(IND),IALT(IND)
  120     FORMT(1H ,I3,2(1X,D18.9),1X,I7)
          WRITE (5,130)
  130     FORMAT(1H ,'ANY MORE BEACONS FROM BLF?(Y=1,N=0)'/)
          READ(5,140)IMOR1
  140     FORMAT(I1)
          IND2=J
          IF(IMOR1.EQ.0) GO TO 160
  150 CONTINUE
  160 CONTINUE
     IND2=IND2+1
      WRITE(5,170)
  170  FORMAT(1H ,' ANY NON-STANDARD BEACONS ?(Y=1,N=0)'/)
     READ(5,180)IMOR2
  180 FORMAT(I1)
     IF(IMOR2.EQ.0) GO TO 260
  185  WRITE(5,190)
  190  FORMAT(1H ,' ENTER A UNIQUE IDENTIFIER.(3-DIGIT INTEGER).'/)
     READ(5,200) IDENT(IND2)
  200 FORMAT(I3)
     WRITE(5,210)
  210 FORMAT(1H ,'ENTER LATITUDE AND LONGITUDE.(D18.9,D18.9)'/)
     READ(5,220) SLAT(IND2),SLON(IND2),IALT(IND2)
  220 FORMAT(2(D18.9,1X),I5)
C    ****************************************************************
C
C    SUBROUTINE ECEF TAKES DMSDEG COORDINATES AND CHANGES THEM TO ECEF.
C
C    ****************************************************************
       SUBROUTINE ECEF
       COMMON/REF/REFLAT,REFLON,REFALT
       COMMON/DATS/DTR,KBEAC,A,B,KCOOR
       COMMON/ARRYS/CLALON(10,3),CECEF(10,3)
       IMPLICIT REAL*8(A-H,O-Z)
C    I,J,K,L,M,N: ARE INTEGERS
C    TRANSFORM LAT LON  TO ECEF COORDINATES
       KCOOR=KBEAC+1
       CLALON(KCOOR,1)=REFLAT*DTR
       CLALON(KCOOR,2)=REFLON*DTR
       CLALON(KCOOR,3)=REFALT
       DO 170 J=1,KCOOR
       TT1=DCOS(CLALON(J,1))
```

```
      TT2=DSIN(CLALON(J,1))
      TT3=DCOS(CLALON(J,2))
      TT4=DSIN(CLALON(J,2))
      TT5=CLALON(J,3)
      TT6=DSQRT((A*TT1)**2+(B*TT2)**2)
      TT7=A*A/TT6+TT5
      CECEF(J,1)=TT1*TT4*TT7
      CECEF(J,2)=(B*B/TT6+TT5)*TT2
      CECEF(J,3)=TT1*TT3*TT7
      WRITE(6,169)(CECEF(J,JJ),JJ=1,3)
  169 FORMAT(1H ,3F20.3)
  170 CONTINUE
      RETURN
      END
C    ***************************************************************
C
C   SUBROUTINE ENU CHANGES ECEF COORDINATES INTO ENU COORDINATES.
C
C    ***************************************************************
      SUBROUTINE ENU(J)
      COMMON/DATS/DTR,KBEAC,A,B,KCOOR
      COMMON/ARRYS/CLALON(10,3),CECEF(10,3)
      DIMENSION Q1(3,3),Q5(3,3),P03(3),X03(3),CLOCAL(10,3)
      IMPLICIT REAL*8(A-H,O-Z)
C    I,J,K,L,M,N: ARE INTEGERS
C    TRANSFORM ECEF TO LOCAL ENU COORDINATES.
      U0=CECEF(KCOOR,1)
      V0=CECEF(KCOOR,2)
      W0=CECEF(KCOOR,3)
      TT1=DCOS(CLALON(KCOOR,1))
      TT2=DSIN(CLALON(KCOOR,1))
      TT3=DCOS(CLALON(KCOOR,2))
      TT4=DSIN(CLALON(KCOOR,2))
      Q1(1,1)=-TT3
      Q1(1,2)=0.0D0
      Q1(1,3)=TT4
      Q1(2,1)=-TT2*TT4
      Q1(2,2)=TT1
      Q1(2,3)=-TT2*TT3
      Q1(3,1)=TT1*TT4
      Q1(3,2)=TT2
      Q1(3,3)=TT1*TT3
      CALL GMTRA(Q1,Q5,3,3)
      DO 180 J=1,KCOOR
```

```
      PO3(1)=CECEF(J,1)-UO
      PO3(2)=CECEF(J,2)-VO
      PO3(3)=CECEF(J,3)-WO
      CALL GMPRD(Q1,PO3,XO3,3,3,1)
      CLOCAL(J,1)=XO3(1)
      CLOCAL(J,2)=XO3(2)
      CLOCAL(J,3)=XO3(3)
      WRITE(6,179)(CLOCAL(J,JJ),JJ=1,3)
  179 FORMAT(1H ,3D20.9)
  180 CONTINUE
      EO=DSQRT(UO**2+VO**2+WO**2)
      RETURN
      END
C   ***************************************************************
C
C SUBROUTINE GDOP FINDS THE HDOP VALUE OF EACH POSITION ALONG THE ROUTE
C IN RELATION TO THE BEACON POSITION MATRIX BY A SERIES OF MATRIX
C OPERATIONS.
C
C   ***************************************************************
      SUBROUTINE GDOP(K,VAL,D)
C
      IMPLICIT REAL*8(A-H,O-Z)
C   I,J,K,L,M,N: ARE INTEGERS
      COMMON/GMATX/ RTSEG,INDX,GX(20),GY(20),GZ(20)
      COMMON/HDOPP/ X(20),Y(20),Z(20)
      REAL*8 XO,YO,ZO,XI,YI,ZI
      REAL*8 R(20),H(20,3),HT(3,20),HTH(3,3)
      REAL*8 VAL,D
      DIMENSION KF(20),LF(20)
      DIST(XI,YI,ZI,XO,YO,ZO)=DSQRT((XI-XO)**2+(YI-YO)**2+(ZI-ZO)**2)
      DO 50 L=1,3
         DO 50 M=1,20
            H(M,L)=0.D0
            HT(L,M)=0.D0
   50 CONTINUE
      DO 100 J=1,K
         R(J)=DIST(X(J),Y(J),Z(J),GX(J),GY(J),GZ(J))
         H(J,1)=(GX(J)-X(J))/R(J)
         H(J,2)=(GY(J)-Y(J))/R(J)
         H(J,3)=(GZ(J)-Z(J))/R(J)
         WRITE(6,55)J,(H(J,JJ),JJ=1,3)
   55    FORMAT(1H ,'H MATRIX ROW # ',I4,' =',3D12.5)
  100 CONTINUE
      CALL GMTRA(H,HT,20,3)
      CALL GMPRD(HT,H,HTH,3,20,3)
```

```
      DO 120 J=1,3
         WRITE(6,115) J,(HTH(J,JJ),JJ=1,3)
  115    FORMAT(1H ,' HTH MATRIX ROW # ',I4,' =',3D12.5)
  120 CONTINUE
      CALL MINV(HTH,3,D,KF,LF)
      VAL=0.D0
      IF(D.NE.0.D0) VAL=DSQRT(HTH(1,1)+HTH(2,2))
      DO 150 J=1,3
        WRITE(6,140)J,(HTH(J,JJ),JJ=1,3)
  140   FORMAT(1H ,'HTH INVERSE MATRIX ROW #',I4,' =' 3D12.5)
  150 CONTINUE
      RETURN
      END
C    ***************************************************************
C
C
C     SUBROUTINE STIMES PROMPTS THE USER FOR A STARTING AND STOPPING
C     TIME TAG. EACH SIMULATED FILE IS TIME TAGGED FROM 0000.00
C     AND IS INCREMENTED IN INTERVALS DEPENDANT ON THE USER
C     SPECIFIED SPEED OF AIRCRAFT.
C
C      ***************************************************************
C
      SUBROUTINE STIMES
      COMMON/TIMES/T0,T1,T2,START,STOP
C    GET START AND STOP TIMES
      WRITE(5,200)
  200 FORMAT(1H ,'TYPE START TIME (HH,MM,SS).'/)
      READ(5,210)IH1,IM1,IS1
  210 FORMAT(3I3)
      WRITE(5,220)
  220 FORMAT(1H ,'TYPE STOP TIME (HH,MM,SS).'/)
      READ(5,210)IH2,IM2,IS2
      START=TIME(IH1,IM1,IS1,0)
      STOP=TIME(IH2,IM2,IS2,0)
C    SET  FIRST WINDOW TIMES.
      T0=START
      T1=START+(WIDTH/2.D0)
      T2=START+WIDTH
      RETURN
      END
C     ***************************************************
C
C     SUBROUTINE RAPPS READS THE RANGE DATA OFF OF THE USER
```

```
C     CREATED RAPPS DATA FILE AND ALSO CHECKS THE TIME TAGS FOR CONSIS-
C     TENCY WITH START,STOP PARAMETERS.
C
C    ***********************************************************

      SUBROUTINE RAPPS
C    READ RANGE DATA FROM RAPPS DATA FILE.
  500 READ(1,ERR=500,END=1000)UTOTIM,UTIM1,UALT,ISEC,IMS1,IDENT1
     1,IDENT2,IMS2,URANGE,IR1,IR2
      TOTIM=DBLE(UTOTIM)
      TOTIM1=DBLE(UTIM1-FLOAT(IMS1)/1000.)+DFLOAT(IMS1)*1.D-03
C      TOTIM1=TOTIM+DFLOAT(IMS1)*1.D-03
      ALT=DBLE(UALT)
      RANGE=DFLOAT(IR1)+DFLOAT(IR2)*1.D-03
CC   ***************************************************************
C
C  SUBROUTINE TMETAG IS DESIGNED TO COMPUTE THE ROUTE SEGMENTS AND
C  THEIR CORRESPONDING HDOP VALUES IN RELATION TO THE BEACON
C  MATRIX.
C
C    ***************************************************************
      SUBROUTINE TMETAG
      IMPLICIT REAL*8(A-H,O-Z)
C    I,J,K,L,M,N: ARE INTEGERS
      INTEGER TOTAL
      COMMON/GMATX/ RTSEG,INDX,GX(20),GY(20),GZ(20)
      COMMON/HDOPP/ X(20),Y(20),Z(20)
      TOTAL=0
      KSEG=0
      DIST=DABS(GY(INDX)-GY(INDX+1))
      IF(GY(INDX+1).EQ.0.D0) GO TO 300
  100 CONTINUE
      CALL GDOP(HDOP,VAL,D)
      WRITE(6,200) INDX,KSEG,VAL
  200 FORMAT(1H ,' THE HDOP VALUE FOR WAYPOINT #',I3,' SEGMENT #',I3,
     1' IS',D20.7)
      XMTX=GX(INDX)
      YMTX=GY(INDX)
      ZMTX=GZ(INDX)
      X(INDX)=XMTX
      Y(INDX)=YMTX-RTSEG
      Z(INDX)=ZMTX
      XMTX=X(INDX)
      YMTX=Y(INDX)
      ZMTX=Z(INDX)
```

```
      TOTAL=TOTAL+RTSEG
      KSEG=KSEG+1
      IF(TOTAL.LT.DIST) GO TO 100
  300 CONTINUE
      INDX=INDX+1
      RETURN
      END
  505C CONTINUE
C   TEST TIME TAG.
      IF(TOTIM1.GT.STOP)GO TO 2000
      IF(TOTIM1.GT.T2)GO TO 600
      Z1=(ALT+ALTCOR)/6076.1155D0
      JISTAG=10*IDENT1+IDENT2
C      WRITE(6,503)TOTIM1,JISTAG,IMS1,IMS2,RANGE
  503 FORMAT(1H ,D20.11,3I5,D20.11)
C   PICK UP ARRAY INDEX VIA STATION IDENT CODE.
      DO 510 J=1,KBEAC
      IF(JISTAG.EQ.ISTAG(J))GO TO 515
  510 CONTINUE
C   STATION IDENT NOT IN DATA BASE, GET NEXT RETURN.
      GO TO 500
C   STATION IDENTIFIED; PROCESS DATA.
  515 INDEX=J
C      PREFILTER ALGORITHM
      IF(R(INDEX).EQ.0.D0)GO TO 550
      FX3=1.5D0
      IF((T1-T9(INDEX)).GT.DFLOAT(INCTIM))FX3=40.D0
      EE1=DABS(RANGE-D9(INDEX)-R(INDEX))
      IF(EE1.GT.FX3)GO TO 500
      IF((IRCOUN(INDEX).LT.2).OR.(FX3.EQ.40.D0))GO TO 550
      IF(DABS(RANGE-D9(INDEX)-RANGE1(INDEX,IRCOUN(INDEX)-1)).GT..7D0)
     1GO TO 500
  550 CONTINUE
C      END OF PREFILTER LOGIC ALGORITHM.
C   STORE NEW RANGE RETURN FOR CURRENT WINDOW.
      IRCOUN(INDEX)=IRCOUN(INDEX)+1
      RANGE1(INDEX,IRCOUN(INDEX))=RANGE-D9(INDEX)
      RTIME(INDEX,IRCOUN(INDEX))=TOTIM1
      DELRT(INDEX,IRCOUN(INDEX))=TOTIM1-TO
      KR=IRCOUN(INDEX)
C      WRITE(6,557)INDEX,RANGE1(INDEX,KR),RTIME(INDEX,KR),DELRT(INDEX,KR)
  557 FORMAT(1H ,I4,3D20.9)
      GO TO 500
  600 CONTINUE
```

```
C   STRAIGHT LINE FIT OF RANGE VS. TIME.
      K1=0
      DO 650 J=1,KBEAC
  605 CONTINUE
C      WRITE(6,606)J,IRCOUN(J)
  606 FORMAT(1H ,2I10)
      IF(IRCOUN(J).LT.2)GO TO 650
      SUMR=0.D0
      SUMRT=0.D0
      SUMT=0.D0
      SUM2T=0.D0
C   ACCUMULATE SUMS FOR NORMAL EQUATIONS.
      DO 610 K=1,IRCOUN(J)
      SUMR=SUMR+RANGE1(J,K)
      SUMT=SUMT+DELRT(J,K)
      SUMRT=SUMRT+DELRT(J,K)*RANGE1(J,K)
      SUM2T=SUM2T+DELRT(J,K)**2
  610 CONTINUE
C   COMMPUTE SLOPE AND INTERCEPT OF LEAST SQUARES STRAIGHT LINE.
C   A0=SLOPE OF LINE, B0=INTERCEPT OF LINE.
      D0=DFLOAT(IRCOUN(J))*SUM2T-SUMT**2
      IF(D0.EQ.0.0D0)WRITE(6,615)IH1,IM1,IS1,IRCOUN(J)
  615 FORMAT(1H ,3I4,1X,I5)
      IF(D0.EQ.0.0D0)GO TO 650
      A0=(DFLOAT(IRCOUN(J))*SUMRT-SUMT*SUMR)/D0
      B0=(SUM2T*SUMR-SUMT*SUMRT)/D0
      GO TO 605
  640 CONTINUE
C   NO OUTLIERS, STORE EXTRAPOLATED RANGE VALUE FOR CURRENT WINDOW.
C   RANGE IS VALID AT CENTER OF CURRENT WINDOW, BASED ON CURRENT
C   LEAST SQUARES STRAIGHT LINE CURVE FIT TO EXISTING RANGE VALUES.
C   K1 COUNTS NUMBER OF DIFFERENT STATIONS WITH VALID RANGES.
      K1=K1+1
      Q(K1)=J
      R(J)=A0*(WIDTH/2.D0)+B0
      T9(J)=T1
C      WRITE(6,645)J,T1,R(J),A0,B0
  645 FORMAT(1H ,I4,4D20.9)
  650 CONTINUE
  660 ITER=0
C   BEGIN POSITION SOLUTION PROCESSING; LEAST SQUARES.
C   COMPUTE POSITION IN LOCAL ENU COORDINATES BY SOLVING SIMULTANEOUS
C   RANGE EQUATIONS IN LEAST SQUARES MANNER.
  690 XL1=0.D0
      XL2=0.D0
```

```
      XL3=0.D0
      XL4=0.D0
      XL5=0.D0
      DD6=0.D0
      K2=0
C   ACCUMULATE RANGE RESIDUALS FOR CURRENT ITERATION.
      DO 700 I=1,K1
      N=Q(I)
      IF(N.LT.0)GO TO 700
      K2=K2+1
      D1=X1-CLOCAL(N,1)
      D2=Y1-CLOCAL(N,2)
      D3=Z1-CLOCAL(N,3)
      S(N)=DSQRT(D1**2+D2**2+D3*(D3))
      D4=D1/S(N)
      D5=D2/S(N)
      XM2(N)=R(N)-S(N)
      D6(N)=XM2(N)
C      WRITE(6,6001)N,D1,D2,D3,S,D4,D5,D6(N)
 6001 FORMAT(1H ,I5,7D15.7)
      XL1=XL1+D6(N)*D4
      XL2=XL2+D6(N)*D5
      XL3=XL3+D4*D4
      XL4=XL4+D4*D5
      XL5=XL5+D5*D5
      IF(DABS(D6(N)).LE.DD6)GO TO 692
      IR9=I
      DD6=DABS(D6(N))
  692 CONTINUE
  700 CONTINUE
C   SOLVE FOR LEAST SQUARES DELX, DELY AT CURRENT ITERATION.
      IF(K2.LT.3)GO TO 745
      D0=XL3*XL5-XL4*XL4
      IF(D0.EQ.0.D0)X1=0.D0
      IF(D0.EQ.0.D0)Y1=0.D0
      IF(D0.EQ.0.D0)GO TO 745
      E1=(XL5*XL1-XL4*XL2)/D0
      E2=(XL3*XL2-XL4*XL1)/D0
C   UPDATE CURRENT ESTIMATE OF POSITION IN X AND Y.
      X1=X1+E1
      Y1=Y1+E2
C      WRITE(6,7001)XL1,XL2,XL3,XL4,XL5
C      WRITE(6,7001)D0,E1,E2,X1,Y1
 7001 FORMAT(1H ,'CTPR',5D20.9)
```

```
C    TEST FOR CONVERGENCE. IF NOT, ITERATE AGAIN.
       ITER=ITER+1
       IF(ITER.GT.ITERAT)X1=0.D0
       IF(ITER.GT.ITERAT)Y1=0.D0
       IF(ITER.GT.ITERAT)GO TO 745
       IF((DABS(E1)+DABS(E2)).GT..001D0)GO TO 690
C       BEGIN POST FILTER PROCESSING.
       IF(R(Q(IR9)).EQ.0.D0)GO TO 701
       EE2=DABS(D6(Q(IR9)))
       IF(EE2.LT..04D0)GO TO 701
       Q(IR9)=-Q(IR9)
       GO TO 660
  701  CONTINUE
       EE1=0.D0
       DO 709 J=1,K1
       N=Q(J)
       IF(N.LT.0)GO TO 702
       GO TO 708
  702  CONTINUE
       Q(J)=-N
       N=-N
       S(N)=DIST(X1,Y1,Z1,CLOCAL(N,1),CLOCAL(N,2),CLOCAL(N,3))
       XM2(N)=R(N)-S(N)
       R(N)=S(N)
       GO TO 709
  708  EE1=EE1+DABS(D6(N))
  709  CONTINUE
C       END OF POST FILTER PROCESSING.
C
C    ***********************************************************
C
C
C    SUBROUTINE INCRMT SLIDES THE WINDOW FORWARD AND
C    CLEARS OUT OLD DATA SO THAT THE LAST DATA WHICH WAS READ BUT
C    PROCESSED CAN BE HANDLED.
C
C    ***********************************************************
       COMMON/TIMES/T0,T1,T2,START,STOP
       COMMON/DATS/DTR,KBEAC,A,B,KCOOR
       COMMON/SLIDE/IS1,IM1,IH1,WIDTH,INCTIM
C     SLIDE WINDOW FORWARD INCTIM SECONDS.
       IS1=IS1+INCTIM
       IF(IS1.LT.60)GO TO 749
       IS1=IS1-60
       IM1=IM1+1
       IF(IM1.LT.60)GO TO 749
```

```
      IM1=IM1-60
      IH1=IH1+1
      IF(IH1.LT.24)GO TO 749
      IH1=IH1-24
  749 CONTINUE
      TO=TIME(IH1,IM1,IS1,0)
      T1=TO+(WIDTH/2.D0)
      T2=TO+WIDTH
C   REMOVE OLD DATA FROM CURRENT WINDOW.
      DO 770 I=1,KBEAC
      N9=IRCOUN(I)
      IF(N9.EQ.0)GO TO 770
      DO 750 J=1,N9
      IF(RTIME(I,J).GT.TO)GO TO 755
  750 CONTINUE
      IRCOUN(I)=0
      GO TO 770
  755 CONTINUE
      J8=J-1
      DO 760 J9=1,N9-J8
      RANGE1(I,J9)=RANGE1(I,J9+J8)
      RTIME(I,J9)=RTIME(I,J9+J8)
      DELRT(I,J9)=RTIME(I,J9)-TO
  760 CONTINUE
      IRCOUN(I)=N9-J8
  770 CONTINUE
C   GO BACK AND PROCESS LAST DATA WHICH WAS READ, BUT NOT USED.
      RETURN
      END
```