A Graphical Analysis of the Kalman Filter

by

Conrad Jay Remley

Electrical Engineering

Submitted in Partial Fulfillment of the Requirements of the University Undergraduate Fellows Program

1981-1982

Approved by:

Dr. J.H. Painter

April 1982

ABSTRACT

Research has been generated by the desire to make the Global Positioning System (GPS), a highly accurate navigation system using satellites, accessible to the civilian user at reasonable cost. Research up to this point has been done on developing the algorithms for a low cost GPS users set which employs the Kalman Filter. This paper is an overview of past research in the development of these algorithms and documentation of the software to be used to give future researchers a three dimensional graphics tool to analyze developments in the low cost user set algorithms.

ACKNOWLEDGEMENTS

Dr. Stan Swanson

Steven B. Hyde

TABLE OF CONTENTS

LIST OF FIGURES

I. Introduction

The basic principles of navigation as we know them today were used many years ago with very crude means of estimating one's position on the earth. With the use of telecommunications in recent years, navigation techniques have become increasingly accurate. In order to achieve this accuracy, a great amount of complexity was introduced, both in the hardware and software of the systems. Currently, a navigation system is being developed that employs satellites as the fixed points in the navigation process. This navigation system is known as the Global Positioning System (GPS).

The GPS is made up of three types of hardware components, two of which will be considered, the satellites and the receiver. Currently six (of the proposed 24) satellites are orbiting the earth at 10898 meters. They are constantly transmitting their ephemeris (position) and the time in a complex digital code down to earth [1]. They are also equipped with a clock accurate to at least 976 microseconds.

The receiver is capable of estimating the position of the user by calculating the range it is from each of a constellation of three satellites. The output is a coordinate reading in a convenient coordinate system. The system used in this project is the Earth Centered-Earth Fixed (ECEF) having variables x, y and z. Hyde [2] sufficiently describes this system. The optimal receiver has a very accurate clock, the capability of receiving three streams of satellite data simultaneously (multi-channeled) and a static position (stationary). In practical usage, the user is moving and must use a low cost set which has a semi-accurate clock and the capability of receiving only one data stream at a time.

Errors are categorized in two major sections. Errors inherent to the system such as slight deviations in the ephemeris, clock bias, atmospheric propagation delay and multipath error are estimated by the system itself. However, error is introduced by the use of the low cost system. Because of the semi-accurate clock, the correct distance will not be calculated to the respective satellite. Since the user is mobile, inaccurate measurements result because the user has changed positions by the time a coordinate set is calculated. Introducing the sampled data requirement also reduces accuracy because the system does not operate with the latest range measurement from each satellite.

Two major methods are used to minimize the errors induced by the use of the low cost user set. In order to estimate the clock error, a fourth range measurement is taken. This is called a bias measurement. The other method of minimizing user errors is the employment of a Kalman Filter in the user's set. This filter algorithm has the ability to estimate a user's position based on the previous position.

It is desirable to test variations of the Kalman filter to reduce the user error even more. A simulation of the entire GPS system is the method used to analyze these variations. Using a digital computer, a known trajectory can be generated and subjected to the GPS scheme as simulated by a computer program. Therefore, the input true trajectory can then be compared to the trajectory generated by the simulation.

In the past, graphical analysis of the comparison of the two trajectories was limited to a two dimensional media. Thus only the magnitudes of the differences could be displayed. It is now possible to use the three dimensional media of computer graphics to aid researchers in their attempt to analyze the performance of a Kalman Filter variation. The purpose of this project was to develop the software that enabled the comparison of two trajectories in a three dimension media using computer graphics.

II. GPS Simulation

As previously mentioned, it is desirable to study the performance of revised Kalman filter algorithms using a digital computer. Boehm [3] presented an interactive simulation to test an adaptive Kalman Filter. Hyde [4] used the same simulation concept to develop an interactive simulation compatible with the VAX 11/45 digital computer in order to test an innovation of the Kalman filter. The following is a discussion of this simulation.

Figure 1 (p. 5) is a diagram of the overall GPS simulation. The input to this system is a set of coordinates labeled $y_{u1}(k)$, $y_{u2}(k)$, $y_{u3}(k)$ and $y_{u4}(k)$ which represent the known trajectory coordinates. Therefore the first procedure the GPS simulation algorithm must perform is to generate a path with specifications selected by the researcher. The specifications are 1) velocity of vehicle (determined by the type of vehicle), 2) angle through which the vehicle must turn, 3) time between samples, 4) estimation of clock drift rate and 5) estimation of clock offset. For the purpose of initially setting up a graphical analysis, specifications 4) and 5) will be set to zero so that a simple three dimensional coordinate set is generated.

As shown in Figure 1, the predicted coordinates (the output of the system) are subtracted from the known coordinates. This operation results in a coordinate difference term that drives the rest of the system. In order to

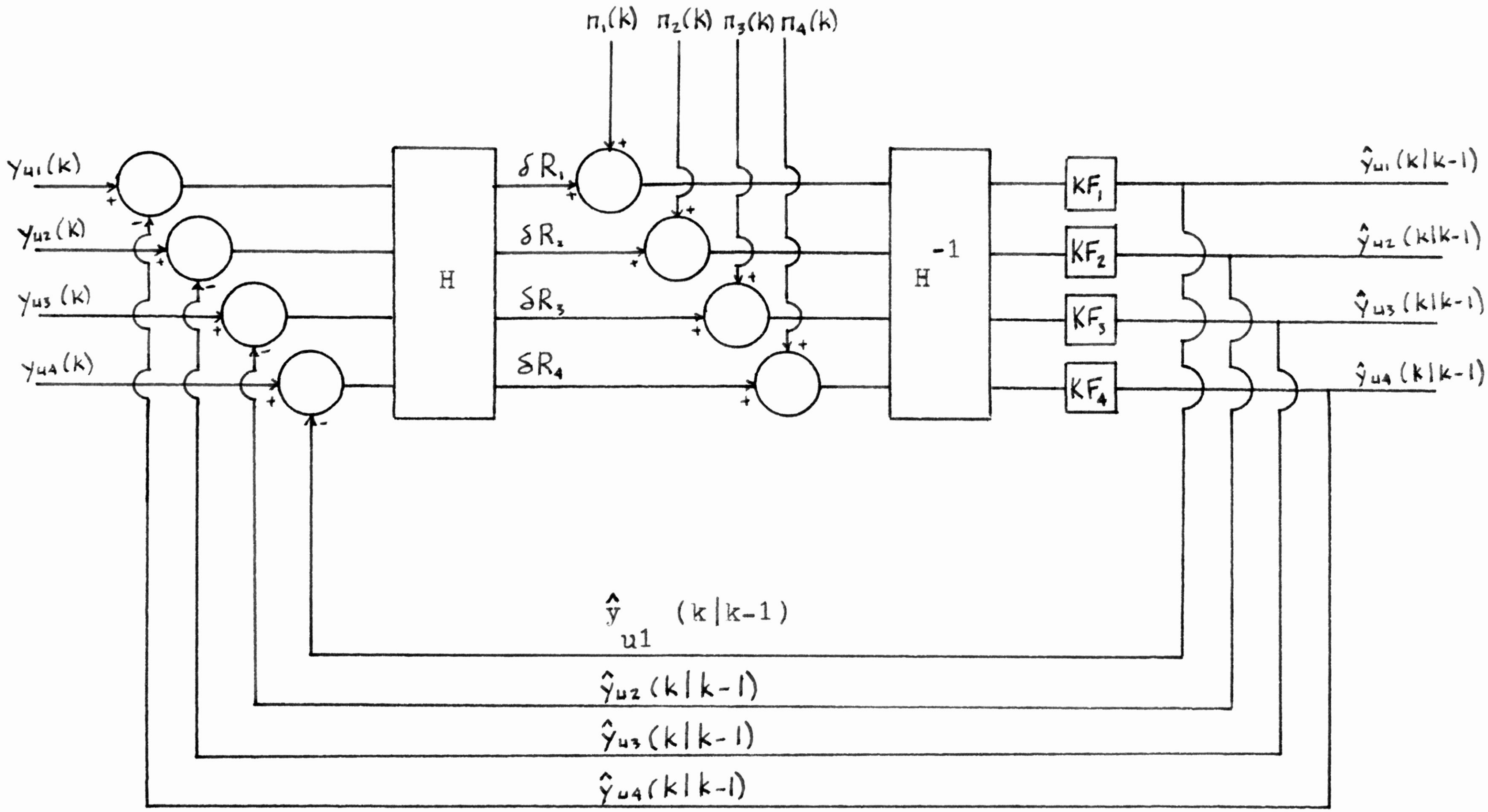

Figure 1. GPS Simulation Diagram

simulate the receiver, the coordinate error terms are converted into a set of range difference measurements, δR_1, δR_2, δR_3 and δR_4. The block, labeled Y in the diagram, a Taylor series linearization about the position estimate, performs this conversion. This linearization results in the direction of cosines matrix which is a function of the angles between the user and the satellites. This matrix changes slowly due to the movement of the user and the satellites but will be considered constant to simplify the algorithm. Optimally each angle should be 120°; but due to physical limitations of satellite communication, the best angle obtained is 85° between all satellites and is used in this simulation.

When receiving data from these satellites, the user is also receiving "white" noise from outer space. This is simulated by a noise generator in the algorithm. Mathematically, the noise is modeled as a Gaussian function with the mean of 1. Once the random number generator injects noise terms into the range measurements, they are converted into coordinate meausrements using the inverse of the direction of cosine matrix.

The importance of the consequence of using a sequential system enters in at this point. The simulation diagram suggests δR_1, δR_2, δR_3 and δR_4 with noise added are known at all times. If a multichannel receiver were used, this would be correct. However, a single channel receiver is being used and only one of the range plus noise measurements is known at a time. Intuition tells us that the projection of these sequential range measurements into position coordinates will cause the predicted position coordinates to be corrected in the line of sight of the satellite the range data is obtained from. However, inspection of the inverse of the direction of cosine matrix shows this is not true. Appendix A contains a proof of this fact based on a two dimensional case.

The next step in the simulation is to process the position coordinates obtained indirectly from the satellites. The Kalman Filters (labeled KF_1, KF_2, KF_3 and KF_4 in Figure 1) use the measured position coordinates to predict the user's position. In general, the Kalman filter is a discrete state variable controller.

The Kalman Filter is a discrete variable compensator that estimates the true position based on the user's previous position estimate. As shown in Figure 2 (p. 8), the filter is driven by one component of the coordinate error, ε_i, to produce the estimated coordinate component - $\hat{y}\,(k/k-1)$. Since this is a compensator, the state variables, a velocity estimation and acceleration estimation, are accessible. However, only the position estimate is fed back to complete the GPS simulation loop. This filter is physically realized as a computer algorithm which processes the actual data obtained in the user's set.

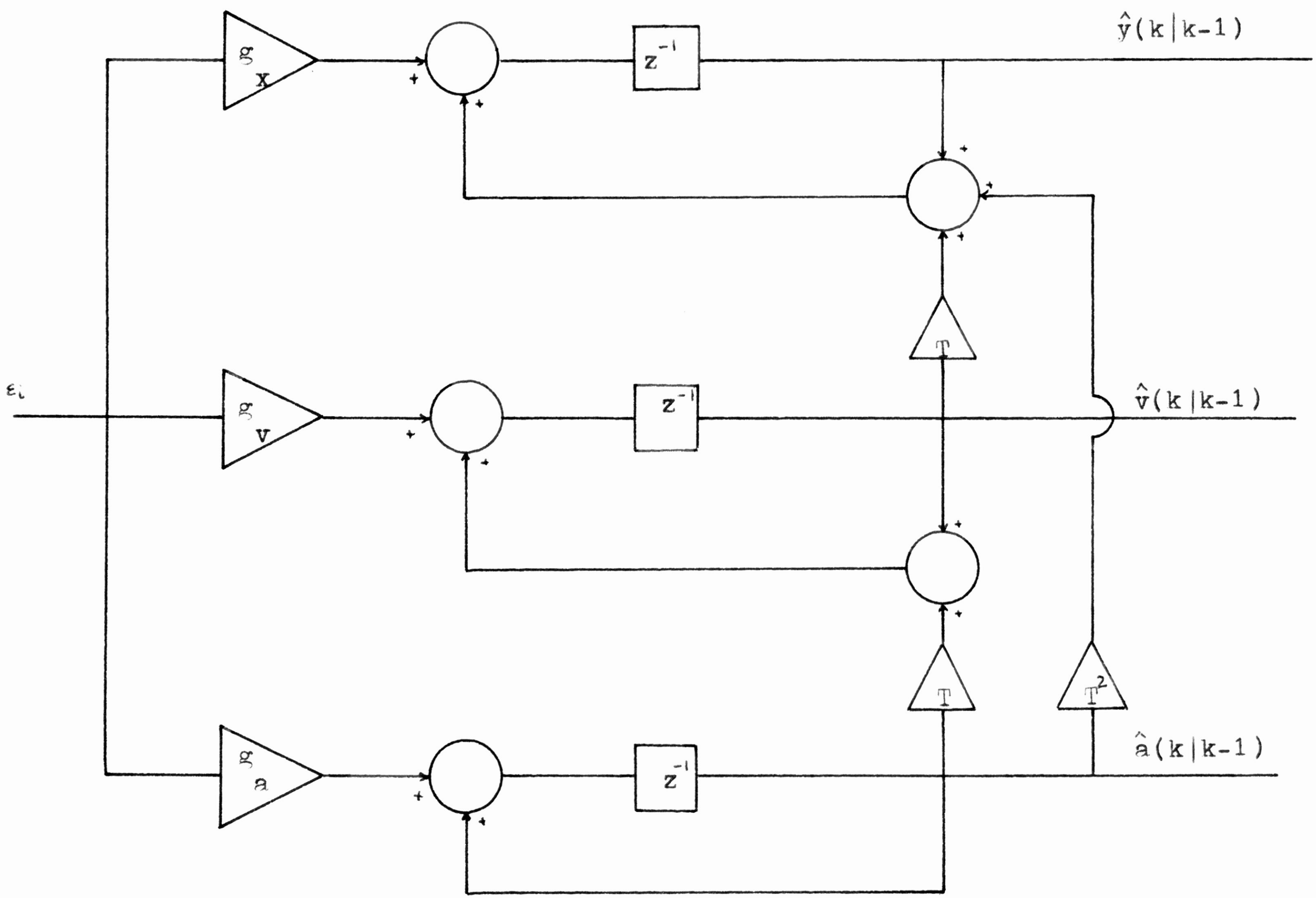

Figure 2. The Kalman Filter

The general result of the use of this filter on the estimated data is a smoothing effect. Past research has been directed toward improving the ability of the Kalman filter to track the known trajectory. Boehm [3] proposed a method to change the gains g_x, g_v and g_a according to the output error. Hyde [4] proposed a nonlinear function to further process the estimation based on the bias error.

III. GRAPHICS SOFTWARE

In order to aid the GPS researcher, it is desirable to display the input and output of the GPS simulation simultaneously in a useful format. A versatile tool that will perform this task is interactive computer graphics. Interactive computer graphics is a means by which a set of data can be displayed in three dimensions on a CRT (cathode ray tube) and manipulated by hardware devices accessible to the user.

Interactive computer graphics systems differ from model to model but most consist of three basic parts--a host computer, the graphics control unit and external hardware devices [5]. The host computer functions as a data generator that performs most data processing tasks. The graphics control unit is a semi-independent processor that stores data from the host and carries out the graphical transformations of the data based on user commands. The external hardware devices such as dials, light pens or digitizers provide means by which transformation commands can be made by the user.

Initially this project was to be done on a Vector General (VG) graphics system managed by the Biochemistry Department at Texas A&M University. The software that linked the hardware devices to the display has been developed and used successfully for several years. However, the computing power of the host computer was limited because of its lack of available memory. Therefore a program was developed to convert data generated by the VAX into a format compatible with the VG. This program is found in Appendix B. The two systems were not linked for direct data transfer, therefore the process was slowed down considerably.

In January 1982, an Evans and Sutherland PS 300 was obtained by the College of Engineering at Texas A&M University and linked to the VAX of the Department of Electrical Engineering. It was then considered feasible to develop the software required to display GPS data on the PS 300 instead of the VG and thus eliminate the host computer limitations imposed and the direct data transfer obstacles. The PS 300 is considered a superior graphics tool to the VG because of the versatile interaction system and the larger storage capacity, which reduces its dependence on the host computer.

Some of the features of the PS 300 system itself made it attractive for use in displaying GPS trajectories. The modeling of the trajectories could be approximated by the PS 300 by setting up a series of small vectors with GPS data points as vector endpoints. The PS 300 also provides library control functions that allow the displayed data to be rotated around any of the three axes and varied in scale using dial inputs. LED readouts are provided in order to inform the user of the degree of rotation and scaling factor being used.

Due to the structured format of the PS 300 control language, the host computer must be programmed to convert the output files of the GPS simulation to a structure compatible with the graphics computer. It is desirable to retain a set of conversion programs to perform the tasks of selecting data points to be graphed, scaling them properly and forming a vector list to be transferred to the PS 300 system without requiring redundant programming steps by the user. Algorithms that initiate the hardware assignments can also be stored in the host computer and transferred to the PS 300 at the time the trajectories are to be plotted.

In order to preserve the generality of the simulation programs, they are not to be revised for the exclusive use of the PS 300. Therefore a program to read the two output files into one file and assign variable names to each data point for future manipulation must be formulated. It is also desirable to plot a selected number of data points. The GPS simulation allows the user to specify the number of samples needed. As stated in the path generator section of the simulation, eighty samples are required to initiate a turn and a variable number of points required to complete the turn, depending on the angle through which the vehicle travels. In order to increase the resolution between the two trajectories, the fewest number of points need to be plotted. It is unnecessary to plot all eighty initial points, therefore this program can delete some of the initialization points. The largest angles will require the most data points, therefore a 180° turn is considered the worst case. The total number of points is then determined by the number of points generated in the complete 180° turn.

The most difficult conversion step is transforming the data generated by the GPS simulation into the coordinate system specified by the PS 300 [6]. The GPS simulation assumes an initial values of x, y and z to be zero and computes coordinates relative to this origin in real space (with dimensional units of meters). The center of the graphics display is assigned to be the origin with the three Cartesian coordinates axes intersecting at this point. The axes only extend one unit in the positive and negative direction, thus data points with magnitude greater than 1 are not displayed on the screen. The maximum value of any coordinate must be found and transformed to be located near the edge of the screen. Therefore all data must be scaled by a factor determined by the maximum datapoint value, so the display limits of the PS 300 are not exceeded. The entire data set must also be translated so the origin of the generated data corresponds with the origin of the display.

Two files must be formulated by the host computer to be used by the PS 300 to display the data and provide external means to alter the display. The first program sequence defines how the hardware devices are to be connected to provide the user with the most beneficial transformations of his application. It also specifies the vector list (the second file mentioned above) for which these connections are valid. In the GPS application, the ability to rotate the graph around each axis and scale it using dials would be needed to give the researcher the ability to focus on any fragment of the trajectory for closer examination. The vector list is created by loading the data that has been scaled and translated into a formatted file which can be referenced by the first command sequence.

Software must be written to transfer this data from the VAX to the PS 300. For debugging purposes, a line by line transfer would be desirable due to the lack of error diagnostics produced by the PS 300. Otherwise a bulk loading program would be faster and easier to use if debugging is unnecessary.

IV. RESULTS AND CONCLUSIONS

At this time the display software package is not complete, therefore no final conclusions can be drawn. Some sections of the software package have been completed and tested.

The program that was used initially to transfer data from the VAX to the VG is included in Appendix B. It was not used directly in the project but included for future reference.

Appendix C contains the main command sequence used by the PS 300 to link the hardware devices to the graphics software. Rotation around the x, y and z axes and a scaling function are enabled by four continuous dials with LED readouts indicating the degree of rotation and scaling factor respectively. Appendix C also contains the programs used to transfer the PS 300 command sequences from the VAX incrementally and in bulk.

The programs yet to be completed are the conversion programs that transform the GPS formatted data to PS 300 compatible data.

The two characteristics that will determine the power of this tool in plotting GPS data is its minimization of user interaction and its ability to resolve the two similar trajectories enough to adequately analyze them. The approach presented here has emphasized the ease in which data can be transformed for use on the PS 300 from a user standpoint. The resolution quality of this approach has not been verified.

REFERENCES

[1] A.J. Van Dierendonck, S.S. Russel, E.R. Kopitzke and M. Birnbaum, "The GPS Navigation Message," Global Positioning System, pp. 55-73, 1980.

[2] S.B. Hyde, "A Non-linear Technique to Compensate for Range Bias Error in a Low-Cost GPS Set," TCSL Memo #80-09, August 1980, p. 2.

[3] R.J. Boehm, "Simulation of a Kalman Filter for GPS," TCSL Research Memo 79-13, June 1979.

[4] S.B. Hyde, "A Non-linear Technique to Compensate for Range Bias Error in a Low-Cost GPS Set," TCSL Memo #80-09, August 1980.

[5] J.F. Callan, "Architectural Overview of the Evans and Sutherland PS 300," Evans and Sutherland Computer Corporation, July 1981.

[6] Evans and Sutherland Computer Corporation, "PS 300 User's Manual--Preliminary Edition," Document E&S #901172-007 PI, September 1981.

APPENDIX A

Consider the following graph:

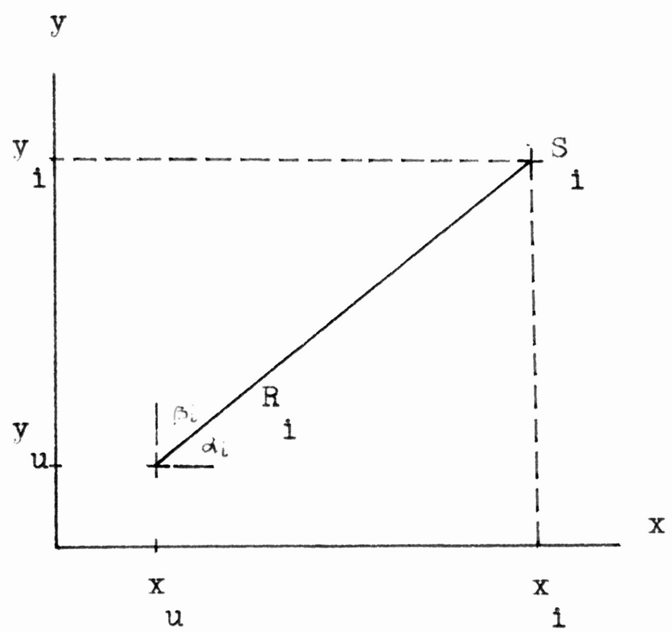

Figure A1

Assume the user's coordinates are (x_u, y_u) and the predicted coordinates are $(\hat{x}_u, \hat{y}_u)$. Assume two satellites, labeled S_i in the figure, have ranges R_1 and R_2 from the user at (x_u, y_u) and ranges $\hat{R}_1$ and $\hat{R}_2$ from the predicted user coordinates $(\hat{x}_u, \hat{y}_u)$. The relationship between the two range differentials, $R_1 - \hat{R}_1$ and $R_2 - \hat{R}_2$, can be expressed as follows:

$$R_1 - \hat{R}_1 = (x_u - \hat{x}_u)\cos\alpha_1 + (y_u - \hat{y}_u)\cos\beta_1$$
$$R_2 - \hat{R}_2 = (x_u - \hat{x}_u)\cos\alpha_2 + (y_u - \hat{y}_u)\cos\beta_2 .$$

Expressed as a vector equation:

$$\begin{bmatrix} R_1 - \hat{R}_1 \\ R_2 - \hat{R}_2 \end{bmatrix} = \begin{bmatrix} \cos\alpha_1 & \cos\beta_1 \\ \cos\alpha_2 & \cos\beta_2 \end{bmatrix} \begin{bmatrix} x_u - \hat{x}_u \\ y_u - \hat{y}_u \end{bmatrix} \qquad (A1)$$

where $\begin{bmatrix} \cos\alpha_1 & \cos\beta_1 \\ \cos\alpha_2 & \cos\beta_2 \end{bmatrix}$ is the direction of cosines matrix, H.

To find the coordinate differentials from the range diferentials, the inverse of the direction of cosines matrix is multiplied to both sides of equation A1. Thus:

$$\begin{bmatrix} x_u - \hat{x}_u \\ y_u - \hat{y}_u \end{bmatrix} = \frac{1}{\cos\alpha_1 \cos\beta_2 - \cos\alpha_2 \cos\beta_1} \begin{bmatrix} \cos\beta_2 & -\cos\beta_1 \\ -\cos\alpha_2 & \cos\alpha_1 \end{bmatrix} \begin{bmatrix} R_1 - \hat{R}_1 \\ R_2 - \hat{R}_2 \end{bmatrix}$$

The coordinates are coupled, that is, each coordinate component depends on both range measurements and their corresponding angles. This illustrates that the range to coordinate conversion does not preserve the direction of the differential range.

APPENDIX B

```
C  THIS IS THE MAIN PROGRAM WHICH READS DATA AND CALLS
C  SUBROUTINES TO TRANSFORM THE DATA TO AN IMAGE THAT THE
C  VECTOR GENERAL CAN USE.
      INTEGER*2 JBUF(60),ITEM
      COMMON /VGDATA/ S,ST(3),LU,LASTEL,JPO,JBUF
      CALL ASSIGN(2,'PICTURE.DAT')
      CALL ASSIGN(3,'TRAJ.DAT')
      LUN=2
      CALL INITVG(LUN)
      READ(3,300) X,Y,Z
      CALL MOVE(X,Y,Z)
C THIS IS SET FOR 122 DATA POINTS
      DO 10 INC=1,122
      READ(3,300)X,Y,Z
      CALL DRAW(X,Y,Z)
   10 CONTINUE
  300 FORMAT(3F10.5)
      END
      SUBROUTINE VGLIST(ITEM)
C  MANAGE BUFFER OF 60 I*2 INTEGERS
      INTEGER*2 JBUF,ITEM
      COMMON /VGDATA/ S,ST(3),LU,LASTEL,JPO,JBUF(60)
      IF (JPO.LT.60) GO TO 100
      WRITE (LU) JBUF
      DO 50 J=1,60
   50 JBUF(J)=0
      JPO=0
  100 JPO=JPO+1
      JBUF(JPO)=ITEM
      LASTEL=ITEM
      RETURN
      END
      FUNCTION LISTEL(VALUE,MODE)
C  SCALES VALUE BETWEEN 0.0 AND 1.0 TO INTEGER RANGE
C  -2048 TO 2047 ; SHIFTS LEFT 4 BITS AND ADDS VG CONTROL
C  FIELD MODE
      L=AMAX1(-2048.,AMIN1(4096.*(VALUE-0.5),2047.))
      LISTEL=16+MODE
      RETURN
      END
      SUBROUTINE MOVE(X,Y,Z)
      INTEGER MCF(3)
      DATA MCF /1,2,11/
      CALL VGLIST(LISTEL(X,MCF(1)))
      CALL VGLIST(LISTEL(Y,MCF(2)))
      CALL VGLIST(LISTEL(Z,MCF(3)))
      RETURN
      END
      SUBROUTINE DRAW(X,Y,Z)
      INTEGER LCF(3)
      DATA LCF/1,2,7/
      CALL VGLIST(LISTEL(X,LCF(1)))
      CALL VGLIST(LISTEL(Y,LCF(2)))
      CALL VGLIST(LISTEL(Z,LCF(3)))
```

```
      RETURN
      END
C   THESE PRODUCE VG DISPLAY DATA FOR 3D VECTOR ABSOLUTE
C   MOVE/DRAW
      SUBROUTINE INITVG(LUN)
C   INITIALIZE PIX FORMAT DISPLAY LIST FOR BIOCHEMISTRY VG
      INTEGER*2 JBUF
      COMMON /VGDATA/ S,ST(3),LU,LASTEL,JPO,JBUF(60)
      REAL SST(4)
      DATA SST/1.0,3*0.0/
      DATA NOOP,IVABS /0,'10004'O/
      LU=LUN
      WRITE(LU)SST
      DO 50 J=1,60
   50 JBUF(J)=0
      JPO=0
      CALL VGLIST(NOOP)
      CALL VGLIST(NOOP)
      CALL VGLIST(IVABS)
      RETURN
      END
      SUBROUTINE FINVG
C   FINISH UP AND CLOSE DISPLAY LIST
      INTEGER*2 JBUF
      COMMON /VGDATA/ S,ST(3),LU,LASTEL,JPO,JBUF(60)
      DATA NOOP,IPRET,IDT /0,'130000'O,'14'O/
      CALL VGLIST(IDT)
      CALL VGLIST(IPRET)
      CALL VGLIST(NOOP)
      IF(JPO.NE.1) WRITE(LU) JBUF
      CALL CLOSE(LU)
      RETURN
      END
```

APPENDIX C

```
WORLD := INSTANCE OF MAIN_1;
MAIN_1 := BEGIN_STRUCTURE
            XROT :=ROTATE IN X 0;
            YROT := ROTATE IN Y 0;
            ZROT := ROTATE IN Z 0;
            SCALE := SCALE BY 1;
DOROTX := F:DXROTATE;
DOROTY := F:DYROTATE;
DOROTZ := F:DZROTATE;
DOSCALE := F:SCALE;
CONNECT DIALS(1):(1)DOROTX;
CONNECT DIALS(2):(1)DOROTY;
CONNECT DIALS(3):(1)DOROTZ;
CONNECT DIALS(4):(1)DOSCALE;
CONNECT DOROTX(1):(1)MAIN_1.XROT;
CONNECT DOROTY(1):(1)MAIN_1.YROT;
CONNECT DOROTZ(1):(1)MAIN_1.ZROT;
CONNECT DOSCALE(1):(1)MAIN_1.SCALE;
PD1:= F:PRINT;
PD2:= F:PRINT;
PD3:= F:PRINT;
PD4:= F:PRINT;
CONNECT DOROTX(2):(1)PD1;
CONNECT PD1(1):(1)DLABEL1;
CONNECT DOROTY(2):(1)PD2;
CONNECT PD2(1):(1)DLABEL2;
CONNECT DOROTZ(2):(1)PD3;
CONNECT PD3(1):(1)DLABEL3;
CONNECT DOSCALE(2):(1)PD4;
CONNECT PD4(1):(1)DLABEL4;
SEND 0 TO (2)DOROTX;
SEND 0 TO (2)DOROTY;
SEND 0 TO (2)DOROTZ;
SEND 0 TO (2)DOSCALE;
SEND 1 TO (3)DOSCALE;
```

```
C THIS PROGRAM INCREMENTALLY TRANSFERS DATA TO THE E&S PS300
      PROGRAM ENSTRANS
      CHARACTER FILE NAME*32, RECORD*80, GO*1
      OPEN(UNIT=2,FILE='TT: ',STATUS='UNKNOWN')
      CALL ASSIGN (10, 'TTD4: ')
      WRITE (2,2002)
 2002 FORMAT (' ', 'TYPE FILE NAME  ', $)
      READ (2,2000) FILE_NAME
 2000 FORMAT (A32)
      OPEN (UNIT=1, FILE=FILE_NAME,STATUS='OLD')
 10   READ (1,1000,END=9999) RECORD
 1000 FORMAT (A80)
      WRITE (10,1005) RECORD
      WRITE (2,1005) RECORD
 1005 FORMAT (' ',A80)
      WRITE (2,1006)
 1006 FORMAT (' ','*',$)
      READ (2,1007) GO
 1007 FORMAT (A1)
      GOTO 10
 9999 CONTINUE
      STOP
      END
```

```
C  THIS PROGRAM TRANSFERS DATA TO THE E&S PS 300 IN BULK
      PROGRAM ENSTRANS
      CHARACTER FILE_NAME*32,RECORD*80,GO*1
      OPEN(UNIT=2,FILE='TT: ',STATUS='UNKNOWN')
      CALL ASSIGN (10,'TTD4:')
      WRITE (2,2002)
 2002 FORMAT (' ', 'TYPE FILE NAME    ' $)
      READ (2,2000) FILE NAME
 2000 FORMAT (A32)
      OPEN (UNIT=1,FILE=FILE_NAME,STATUS='OLD')
 10   READ (1,1000,END=9999) RECORD
 1000 FORMAT (A80)
      WRITE (10,1005) RECORD
 1005 FORMAT (' ',A80)
      GO TO 10
 9999 CONTINUE
      STOP
      END
```