AD-A032 850 NAVAL SURFACE WEAPONS CENTER WHITE OAK LAB SILVER SP--ETC F/G 17/1
SENSOR POSITION LOCATOR.(U)
SEP 75 J VALVANO

UNCLASSIFIED NSWC/WOL/TR 76-8 NL

| OF 2
AD
A032 850



AD A032850

NSWC/WOL/TR 76-8

NSWC/WOL/TR 76-8

# NSWC TECHNICAL REPORT

WHITE OAK LABORATORY

## SENSOR POSITION LOCATOR

BY
Jonathan Valvano

September 1976

NAVAL SURFACE WEAPONS CENTER
WHITE OAK LABORATORY
SILVER SPRING, MARYLAND 20910

Approved for public release; distribution unlimited

COPY AVAILABLE TO DDC DOES NOT PERMIT FULLY LEGIBLE PRODUCTION

DDC
RECEIVED
DEC 2 1976
A

NAVAL SURFACE WEAPONS CENTER
WHITE OAK, SILVER SPRING, MARYLAND 20910

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

## REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: NSWC/WOL/TR 76-8
2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): SENSOR POSITION LOCATOR.
5. TYPE OF REPORT & PERIOD COVERED: Technical rept. Mar[redacted] — Jun[redacted]75
6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): Jonathan Valvano
8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Naval Surface Weapons Center, White Oak Laboratory, White Oak, Silver Spring, Maryland 20910
10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: AIF; 0 0; WA4002
11. CONTROLLING OFFICE NAME AND ADDRESS
12. REPORT DATE: September 1975
13. NUMBER OF PAGES: 113
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)
15. SECURITY CLASS. (of this report): UNCLASSIFIED
15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report): Approved for public release; distribution unlimited
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)
18. SUPPLEMENTARY NOTES
19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Sensor Location
Air Acoustic Sensor
Bearing Angle Sensor
Passive Sensor Location

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A remote acoustic sensor, capable of reporting bearing information on targets it hears, may be deployed in the field without the knowledge of its exact location. This report develops an algorithm which calculates the sensor position through use of the bearing reports from a moving target whose location is known as a function of time. A math model along with FORTRAN listings and simulation runs are included.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

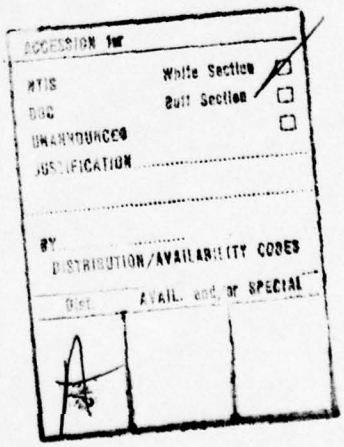
391 596

# PREFACE

This report documents a portion of the work performed at the Naval Surface Weapons Center, White Oak, Silver Spring, Maryland on the Multiple Target Classification and Location task for the REMBASS Advanced Development under Agreement Number 75-14.

C. A. Fisher

C. A. FISHER
By direction



CONTENTS


| | Page |
|---|---|
| Chapter 1 | |
| INTRODUCTION | 4 |
| Chapter 2 | |
| PROBLEMS AND SOLUTIONS | 6 |
| Target Location Errors | 6 |
| Sensor Report Errors | 6 |
| Sensor Rotation Error | 7 |
| Chapter 3 | |
| MATH MODEL FOR CALCULATION OF ROTATION BIAS | 9 |
| Least Squares Method for Bias Calculation | 11 |
| Effects of the Speed of Sound | 12 |
| Correction Algorithm | 13 |
| Adjustments for 3-D | 15 |
| Correction Due to Speed of Sound in 3-D | 15 |
| Calculating $RXYT_1$ | 19 |
| Chapter 4 | |
| CALCULATION OF SENSOR POSITION | 22 |
| Math Model for SPL | 22 |
| Calculation of $A_i$, $B_i$, $C_i$ | 27 |
| Chapter 5 | |
| APPLICATION OF SPL ALGORITHM | 29 |
| Runs #1 | 29 |
| Runs #2 | 32 |
| Conclusions | 33 |
| APPENDIX A ERROR ANALYSIS | A-1 |
| APPENDIX B FORTRAN PROGRAMS AND FLOW CHARTS | B-1 |

## ILLUSTRATIONS


| Figure | Title | Page |
|---|---|---|
| 1 | Sensor Field | 5 |
| 2 | Bias Calculation - Triangles | 10 |
| 3 | Speed of Sound Correction in 2-D | 14 |
| 4 | Sample Bearing Error Due to the Speed of Sound | 16 |
| 5 | Assumptions in the 3-D Model | 17 |
| 6 | Speed of Sound Correction in 3-D | 18 |
| 7 | Calculation of $RXYT_1$ | 20 |
| 8 | Sensor Position Locator | 23 |
| 9 | Definition of theta, $\theta$ | 24 |
| 10 | Calculation of $A_i$, $B_i$, $C_i$ | 28 |
| 11 | Simulated Run #1 - Target Track | 30 |
| 12 | Simulated SPL Lines (Run #1); Bias = 0 degrees | 33 |
| 13 | Simulated SPL Lines (Run #1); Bias = -25 degrees | 34 |
| 14 | Simulated Run #2 - Target Track | 35 |
| 15 | Simulated SPL Lines (Run #2); Bias = 0 degrees | 38 |
| 16 | Affects of Bias on SPL Errors | A-4 |
| 17 | Affects of K (Gaussian noise) on SPL Errors | A-6 |
| 18 | Affects of $\Delta$ Course on SPL Errors | A-7 |
| 19 | Affects of $\Delta Z$ on SPL Errors | A-9 |
| 20 | Affects of $\Delta$ Velocity on SPL Errors | A-10 |
| 21 | Affects of $\Delta$ Temperature on SPL Errors | A-11 |
| 22 | Affects of Number of Bearing Bins on SPL Errors | A-12 |
| 23 | Flowchart of Simulate | B-2 |
| 24 | Flowchart of Input | B-3 |
| 25 | Flowchart of SENCHK | B-4 |
| 26 | Flowchart of OUTPUT | B-5 |
| 27 | Flowchart of SPL | B-6 |
| 28 | Flowchart of ENTER | B-8 |
| 29 | Flowchart of SHIFT | B-9 |
| 30 | Flowchart of TXY | B-10 |
| 31 | Computer Program | B-11-B-63 |


### Tables


| | | |
|---|---|---|
| 1 | Error Analyses of Sensor Position Locator | A-3 |

# Chapter 1

## INTRODUCTION

Remote unattended ground sensors are being developed which have the capability of passively determining the bearing angle of the targets they detect and reporting this information in real time to a central location via an RF link. For one reason or another, it is not always known just where the reporting sensors are located in an exact geographical sense. Although each sensor possesses a unique identifier in the message code transmitted, the original emplacement of a sensor field may not be done by surveying them in, but rather by dropping them out of aircraft, or using rocket or gun delivery.

The purpose of this task has been to determine the location of such sensors after they have been emplanted. The proposed method is to move a target, whose location as a function of time is known, over the sensor field. The sensors will each report the bearing angle of the target several times as the target passes by. Since the target is moving, each report by a sensor will (usually) be somewhat different. That is, the sensor will report a series of angular measurements, which will differ as the target changes location. By knowing the target location as a function of time back at the sensor reporting unit the present algorithm will use the series of reports to calculate the sensor position. The method used constructs radials from the target's position towards the sensor. Each of the many radials should intersect at the sensor position, as shown in Figure 1.

A weighted least-squares algorithm is used to fit the lines to the point which represents the sensor position. This theory is relatively simple, but there are problems which complicate the calculations.

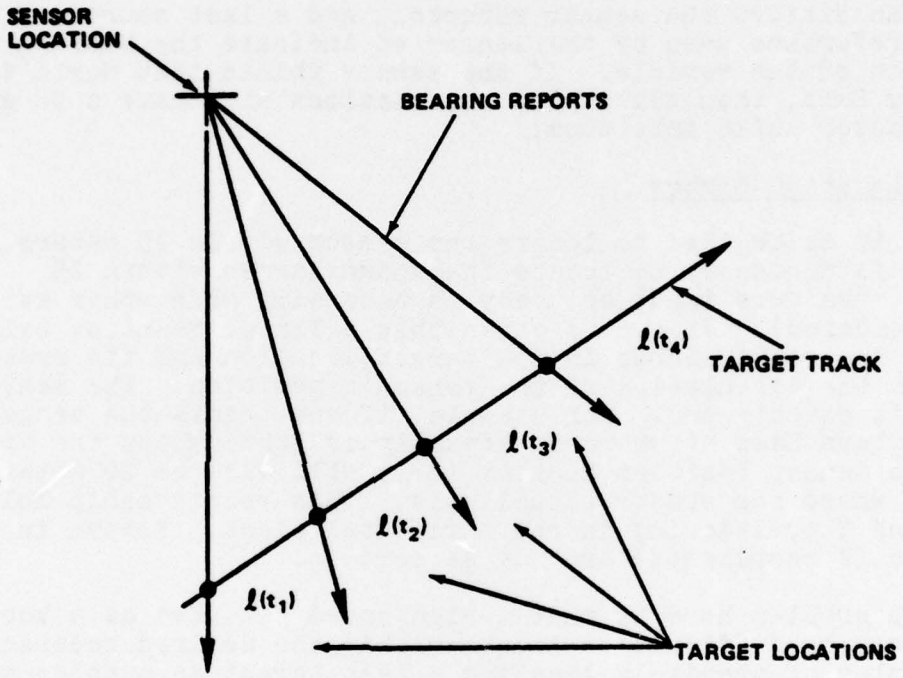



FIG. 1 SENSOR FIELD

# Chapter 2

# PROBLEMS AND SOLUTIONS

As stated there are several factors which complicate the calculation of a sensor's position. One of these is the uncertainty of the location and speed of the "known" target used to locate the sensor. A second problem occurs due to extraneous noise sources which can distort the sensor reports. And a last source of error is the reference used by the sensor to indicate the bearing or direction of the vehicle. If the sensor thinks that North is actually East, then all of its calculations will have a 90 degree error factor built into them.

## Target Location Errors

If it is desired to locate the sensor within 25 meters, then it is necessary to locate the known target within 25 meters (even more input accuracy is necessary when other errors are considered). It can be shown that a linear relation exists between the planar error in the target location and the resulting error in the calculation of the sensor's position. The scale factor is exactly one. For example, if one thinks the target is 20 meters East of where it actually is (throughout the track) then the Sensor Position Locator (SPL) will also be 20 meters East of where the sensor actually is. This relationship holds for X and Y positioning in the horizontal plane. Errors in the altitude (Z coordinate) are not as serious.

The problem here is that a high speed jet used as a known target may be difficult to locate within the desired accuracy. The problem of precisely locating a test target is outside the scope of this investigation, but the errors it causes are not. It is evident that the available accuracy for sensor position location is determined by the current capabilities of target position location for cooperating targets.

## Sensor Report Errors

At the time of investigation, the sensors themselves are also under development. It is not known exactly what kind of accuracy these sensors will have, and the bearing values reported

by the sensors have to be believed, to some extent, as the truth. Therefore, errors originating in the sensor will propagate to the SPL algorithm. It is not the purpose of this task to reduce these inherent sensor errors, but rather to reduce the propagation of these errors to the greatest possible extent. Associated with these "sensor errors" are errors caused by other noise sources in the environment.

It is assumed that the magnitude of the error does fluctuate and the simulation in this work has used gaussian noise as an error model. It is reasonable to try to catch and use the data when it is close to error-free and ignore the data when it is believed to contain large errors. The problem is therefore to find some indicator to tell us of the magnitude of the error, and such indicators do exist. Possibilities include signal to noise ratio, signal strength, and distance from the sensor to the target. The first and last of these indicators were used in the simulation and favorable results were obtained.

The least-squares algorithm will accept a weighting factor (W) based on data quality and it appears that full use of it will improve the quality of the SPL.

Because of the multi-pass SPL algorithm, it is possible to estimate the radial distance from the sensor to the target, and then use it to construct a weighting factor within the algorithm itself. This too has been done in the simulation and improvements were noted.

Sensor Rotation Error

In addition to assuming that the sensor will fall in an arbitrary (X, Y) position, it is also assumed that the sensor will have an arbitrary angular orientation. When the sensor reports, "I see a target at 90 degrees," does that mean North, East, South, or West?

It is probable that each sensor will ultimately contain its own compass, in which case, this source of error would be minimized. It seems reasonable that errors in angular orientation calculated inside the sensor might be on the order of five degrees, when one considers that the compass will be in close proximity to metal parts of the sensor, etc. This amount of angler error, called angle bias, can produce very unsatisfactory errors in the SPL

simulations (on the order of hundreds of meters). Therefore, it seems necessary to include a bias calculation within the SPL algorithm. Fortunately, it is possible to calculate the angle bias without knowing the sensor position, and the latter can then be determined using this bias for compensation.

Errors in the SPL simulation due to all of these sources are discussed in more detail in Appendix A.

# Chapter 3

## MATH MODEL FOR CALCULATION OF ROTATION BIAS

This is a straightforward algorithm that uses three bearing reports ($\alpha 1$, $\alpha 2$, $\alpha 3$) that all have the same bias. It is not necessary that the reports be sequential, and in fact if there are many reports which differ only slightly in bearing, then it is advisable not to use sequential reports because they will probably be in the same or adjacent bearing bin. The algorithm uses angle differences in which the bias subtracts out and, using the geometry of adjacent triangles, calculates an internal angle which in turn leads to the bias. The algorithm will not work if the angle differences are zero which occurs if the target has not moved.

Figure 2 shows the geometry necessary for the calculation of the rotational bias of the sensor. If the sensor reports the three angular bearings ($\alpha 1$, $\alpha 2$, $\alpha 3$) shown in Figure 2, then $d_{12}$ and $d_{23}$ are the linear distances along the target track between those reports. $\alpha_A$ is the direction of the target track with respect to North. By applying the law of sines to Figure 2 one obtains equations 1 and 2 and by also applying Figure 2, the law that the sum of the interior angles of a triangle is 180 degrees, one obtains equations 3 and 4.

$$\frac{d_{12}}{\text{Sin } \theta_1} = \frac{d_{S2}}{\text{Sin } A} \tag{1}$$

$$\frac{d_{23}}{\text{Sin } \theta_2} = \frac{d_{S2}}{\text{Sin } D} \tag{2}$$

$$\theta_1 + \theta_2 + A + D = \tag{3}$$

$$\alpha_A + (\pi - A) + (\pi - \text{Bias} - \alpha_1) = \pi \tag{4}$$

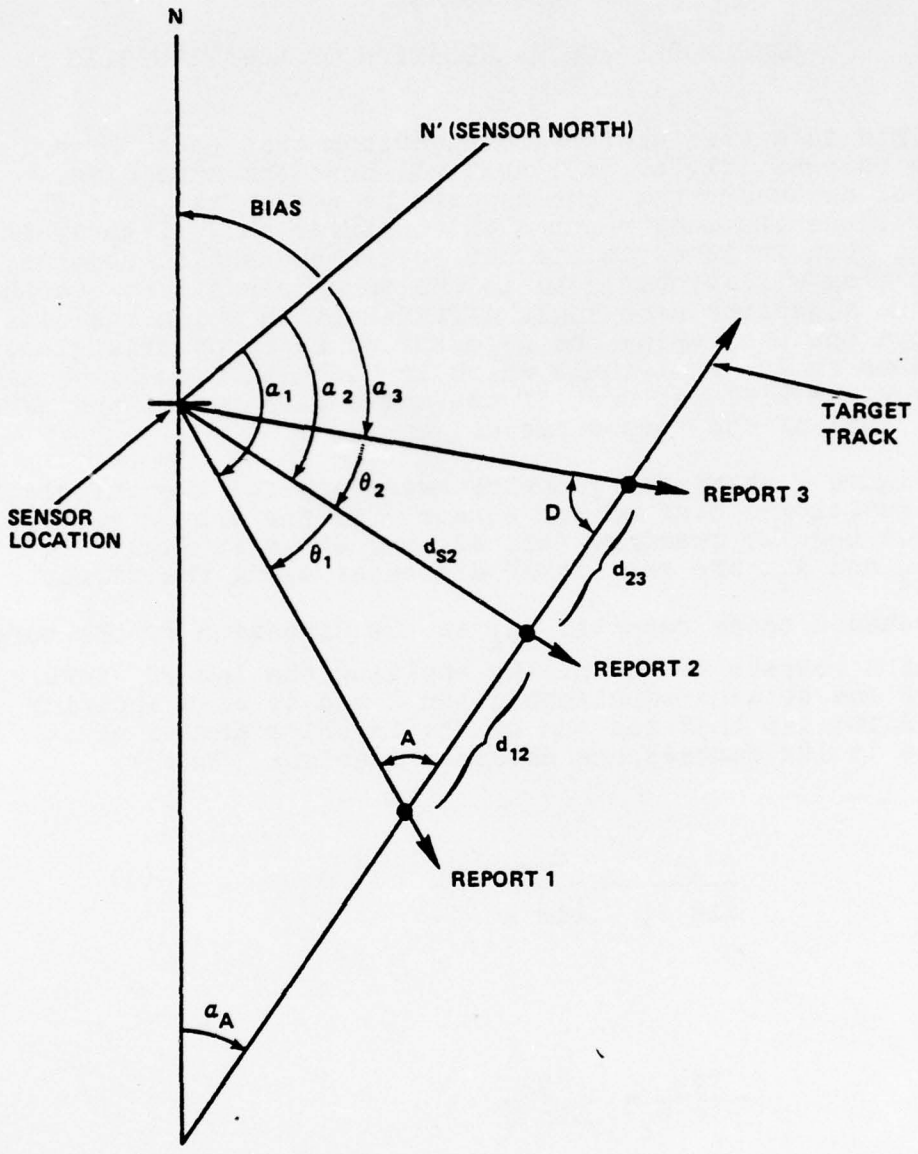



FIG. 2 BIAS CALCULATION - TRIANGLES

Eliminating $d_{S2}$ from equations 1 and 2 yields:

$$d_{12} \frac{\text{Sin } A}{\text{Sin } \theta_1} = d_{23} \frac{\text{Sin } D}{\text{Sin } \theta_2} \tag{5}$$

Using equations 3 and 5 to eliminate D and then solving for A one finds:

$$A = \arctan\left[\frac{\text{Sin } (\theta_1 + \theta_2)}{\frac{d_{12}}{d_{23}} \bullet \left(\frac{\text{Sin}\theta_2}{\text{Sin } \theta_1}\right) - \cos (\theta_1 + \theta_2)}\right] \tag{6}$$

$$\text{where } \theta_1 = (\alpha_1 - \alpha_2)$$
$$\theta_2 = (\alpha_2 - \alpha_3)$$

Equation 4 can be rewritten to solve for the rotational bias:

$$\text{BIAS} = \pi + \alpha_A - \alpha_1 - A \tag{7}$$

Referring to Figure 2, it is possible for the rotational bias to be in the other direction. That is, for the line N' to be on the other side of N. If this is the case, the rotational bias from equation 7 will turn out to be negative.

Least Squares Method for Bias Calculation

Due to the inherent errors in any system, the results achieved by this calculation of the rotational bias will be in error by some amount. If one was to perform this calculation several times for different target runs one would get several values for the rotational bias, called $B_i$. Assuming the errors are gaussian, a best value for the rotational bias would be the mean and an indication of the accuracy of that value would be the standard deviation. If some values are known to be more accurate than others, they can be multiplied by some weighting factor, $W_i$, to yield better results. Then the final value for the bias would be:

$$\text{BIAS} = \frac{\sum_{i=1}^{n} (W_i B_i)}{\sum_{i=1}^{n} (W_i)} \qquad (8)$$

The standard deviation for this BIAS calculation is:

$$\sigma = \frac{\sum_{i=1}^{n} W_i (B_i^2 - \text{BIAS}^2)^{1/2}}{\sum_{i=1}^{n} W_i} \qquad (9)$$

Effects of the Speed of Sound

It is easily observed, when listening to an approaching high-speed target, that the target is ahead of the incoming sound waves. This is because the sound takes a finite $\Delta t$ seconds to travel from the target to the observer. In this $\Delta t$ seconds the target moves, and is therefore ahead of where it was when the sound originated.

The point is to determine the effect of this error on the calculation of the rotational bias and to correct for it. In two dimensions this error can be deterministically calculated independent of radial distance and therefore can be calculated without knowing the sensor's location.

Correction Algorithm. Let $\Delta t$ be the time elapsed between the time when the sound originated at the target ($t_1$) and the time when the sound is received at the sensor ($t_2$). Figure 3 depicts this case where C is the speed of sound, V is the velocity of the target, C ($\Delta t$) is the distance from the sensor to the target at time $t_1$, V($\Delta t$) is the distance the target will travel in the time $\Delta t$, $\alpha_A$ is the target course, $\theta$ is the angular error due to the speed of sound, and $\alpha t_1$ is the report vector referenced to true North (not sensor North). Applying the law of signs (equation 12) and the sum of the interior angles of a triangle equals 180 degrees (equations 10 and 11) to Figure 3 yields:

$$(180 - \alpha_{t1}) + (180 - \beta) + \alpha_A = 180^\circ \qquad (10)$$

$$\gamma = 180^\circ - \theta - \beta \qquad (11)$$

$$\frac{C(\Delta t)}{\text{Sin } \gamma} = \frac{V(\Delta t)}{\text{Sin } \theta} \qquad (12)$$

Substituting from equation 11 into equation 12 and simplifying gives the error due to the speed of sound, $\theta$, to be:

$$\theta = \arctan \frac{\text{Sin } \beta}{\frac{C}{V} - \cos \beta} \qquad (13)$$

And from equation (10) we find B is:

$$\beta = 180 - \alpha_{t1} = \alpha_A$$

= 180 degrees - report bearing referenced to true North + target course.

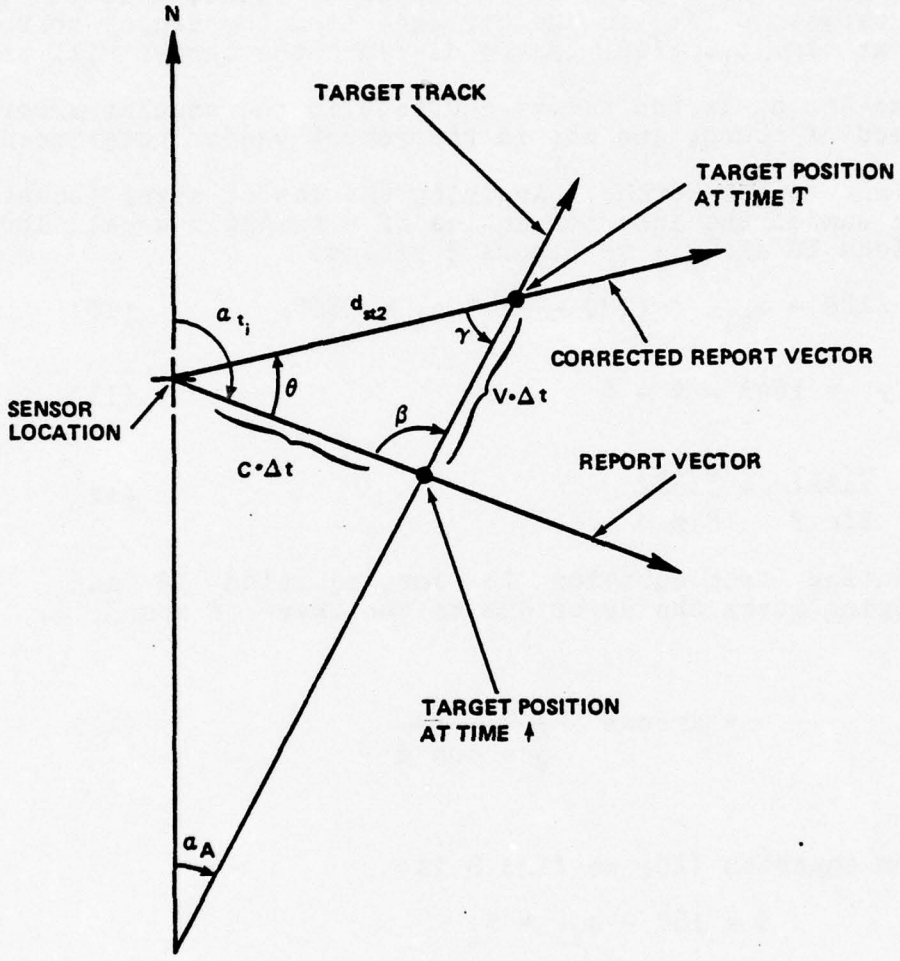



FIG. 3 SPEED OF SOUND CORRECTION IN 2-D

Figure 4 gives the correction angles for some values of B and V/C. As an example, if the target were reported such that B = 60 degrees and the velocity was 300 mph, or V/C = 0.4; then the target would really be 23 degrees ahead of the report bearing.

Adjustments for 3-D. The previous discussions were all done in a two dimensional space, that is for the target track in the same plane as the sensor. To extend the algorithm to three dimensions will require the following simplifying assumptions; (1) that the target will travel in a horizontal plane parallel to the ground plane, (2) that the sensor will report in the ground plane as if the target were also in the ground plane, and (3) that the altitude of the sensor, Z0, will be assumed to be zero. Thus the altitude of the target, ZA, is just the perpendicular distance between the two parallel planes. Figure 5 depicts these assumptions.

Correction Due to Speed of Sound in 3-D. In 2-D the correction error, $\theta$, due to the speed of sound is independent of the horizontal distance from the sensor to the target track. Unfortunately, this is not true if there is a sizable altitude difference between the target track and the sensor.

Since $\theta$ is needed to calculate the sensor location and the sensor location is needed to calculate the altitude correction to $\theta$, an iterative algorithm must be used whereby the sensor location is first calculated without the altitude correction for $\theta$. This sensor location is then used to determine a more accurate value for $\theta$, this new value of $\theta$ is used to generate an improved sensor location, and so forth until the desired accuracy is achieved.

Figure 6 shows the speed of sound correction for three dimensions where ZA is the perpendicular distance between the target plane and the sensor plane, $RXYT_1$ is the distance from the sensor to the projection of the target onto the sensor plane at time $t_1$, and $\theta$ is the correction error due to the speed of sound. Note that both $\gamma$ and $\theta$ are in the ground plane.

The calculations for the three dimensional case are very similar to those for the two dimensional case. In fact, equations 10 and 11 still hold. However, equation 12 now reads:

$$\frac{RXYT_1}{\sin \gamma} = \frac{(V)\ (\Delta t)}{\sin \theta} \qquad (15)$$

| V/C OR RATIO | .9 | .8 | .7 | .6 | .5 | .4 | .3 | .2 | .1 | .05 | .025 | .0125 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ANGLE β MPH | 675 | 600 | 525 | 450 | 375 | 300 | 225 | 150 | 75 | 37.5 | 18.8 | 9.4 | 0 |
| ANGLE β M/S | 302 | 268 | 235 | 201 | 168 | 134 | 101 | 67 | 34 | 17 | 8.4 | 4.2 | 0 |
| 180° | 0° | 0° | 0° | 0° | 0° | 0° | 0° | 0° | 0° | 0° | 0° | 0° | 0° |
| 150° | 14° | 13° | 12° | 11° | 10° | 8° | 7° | 5° | 3° | 1° | 1° | 0° | 0° |
| 135° | 21° | 20° | 18° | 17° | 15° | 12° | 10° | 7° | 4° | 2° | 1° | 1° | 0° |
| 120° | 32° | 26° | 24° | 22° | 19° | 16° | 13° | 9° | 5° | 2° | 1° | 1° | 0° |
| 90° | 42° | 39° | 35° | 31° | 27° | 22° | 17° | 11° | 6° | 3° | 1° | 1° | 0° |
| 60° | 55° | 49° | 43° | 37° | 30° | 23° | 17° | 11° | 5° | 3° | 1° | 1° | 0° |
| 45° | 60° | 52° | 44° | 36° | 27° | 22° | 15° | 9° | 4° | 2° | 1° | 1° | 0° |
| 30° | 64° | 52° | 42° | 32° | 24° | 17° | 11° | 7° | 3° | 1° | 1° | 0° | 0° |
| 10° | 54° | 33° | 21° | 14° | 10° | 7° | 4° | 2° | 1° | 1° | 0° | 0° | 0° |
| 0° | 0° | 0° | 0° | 0° | 0° | 0° | 0° | 0° | 0° | 0° | 0° | 0° | 0° |

FIG. 4 SAMPLE BEARING ERROR DUE TO THE SPEED OF SOUND

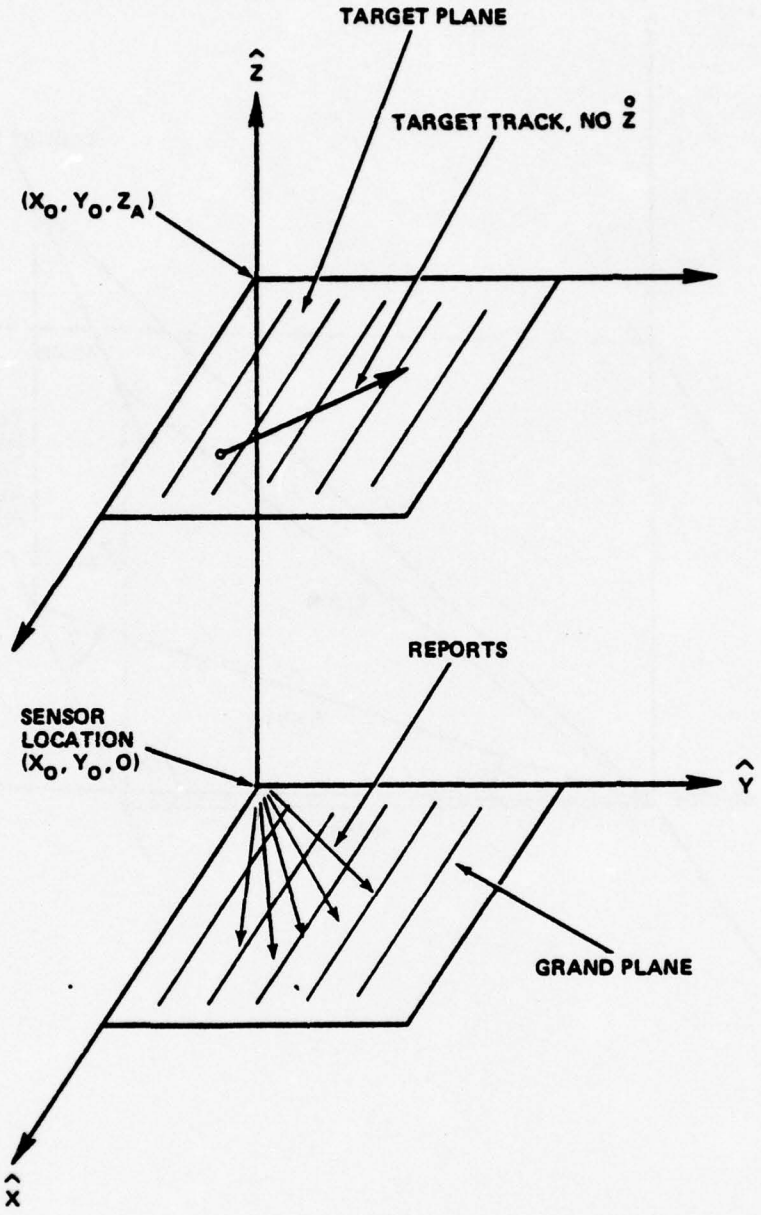



FIG. 5 ASSUMPTIONS IN THE 3-D MODEL

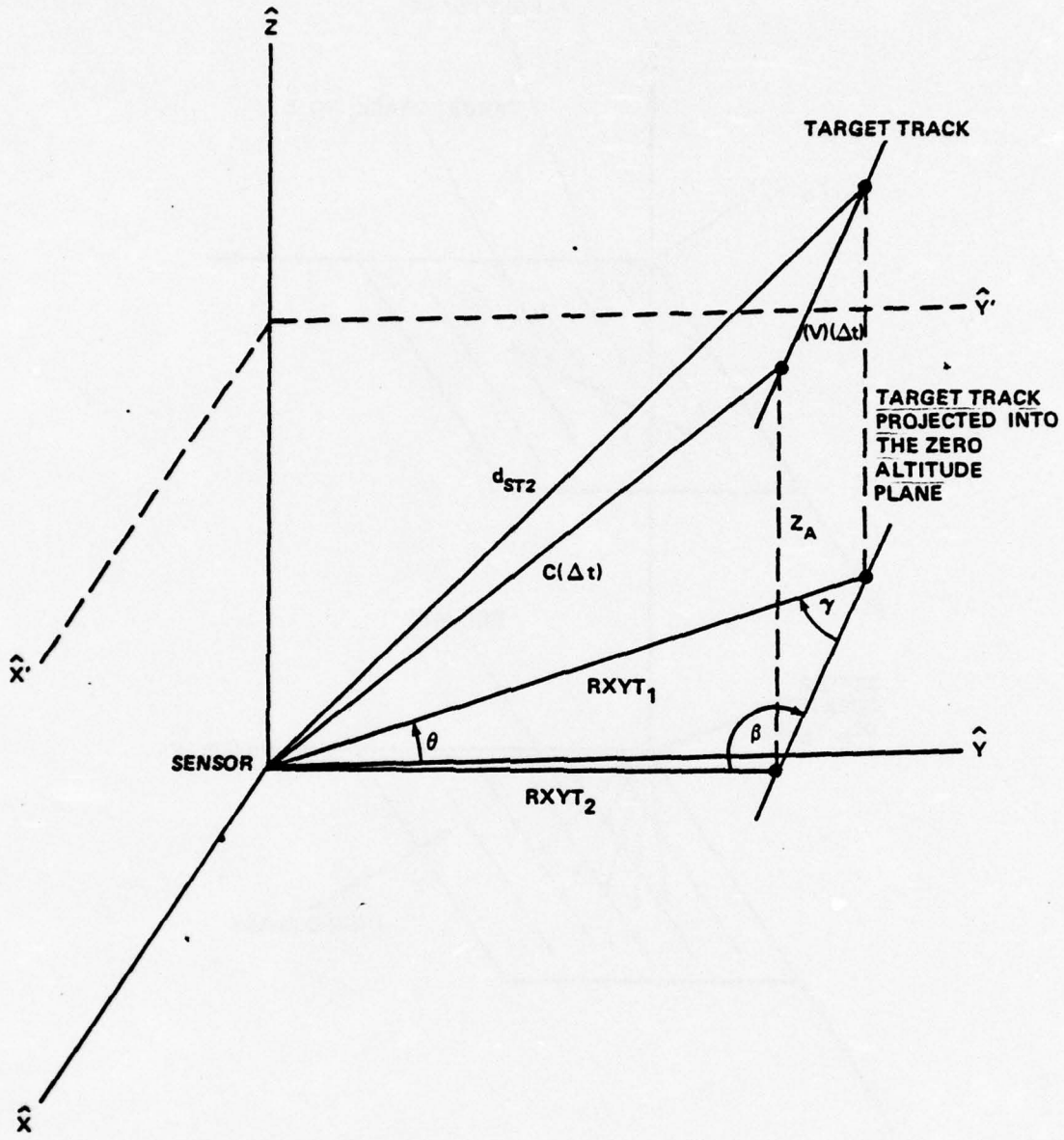



FIG. 6 SPEED OF SOUND CORRECTION IN 3-D

Observe that the projection of the line (C) (Δt) onto the sensor plane forms a right triangle. This gives one a relationship between equations 12 and 15, namely:

$$(C)\ (\Delta t) = \sqrt{(ZA)^2 + (RXTT_1)^2} \qquad (16)$$

Let $$\text{Ratio} = \left(\frac{C}{V}\right) \left(\frac{1}{\sqrt{1 - (ZA/RXYT_1)^2}}\right) \qquad (17)$$

Then equation 13 for the three dimensional case becomes:

$$\theta = \arctan \frac{\sin \beta}{\text{Ratio} - \cos \beta} \qquad (18)$$

Unfortunately, to calculate θ, one needs to know $RXYT_1$ (equation 17), and $RXYT_1$ is dependent upon the sensor location.

Calculating $RXYT_1$. Chapter 4 discusses the sensor position. Once this is done one can come back and calculate $RXYT_1$.

Figure 7 is a two dimensional view of the ground plane of the three dimensional speed of sound correction shown in Figure 6. β is defined by equation 14, θ is the correction error due to the speed of sound, and $RXYT_1$ and $RXYT_2$ are the projections of $(C)(\Delta t)$ and $(d_{st2})$ onto the sensor plane.

From the definition of the sine of a right triangle it is seen from Figure 7 that:

$$RXYT_1 = \frac{\sqrt{(X_0 - X_p)^2 + (Y_0 - Y_p)^2}}{\sin \beta} \qquad (19)$$

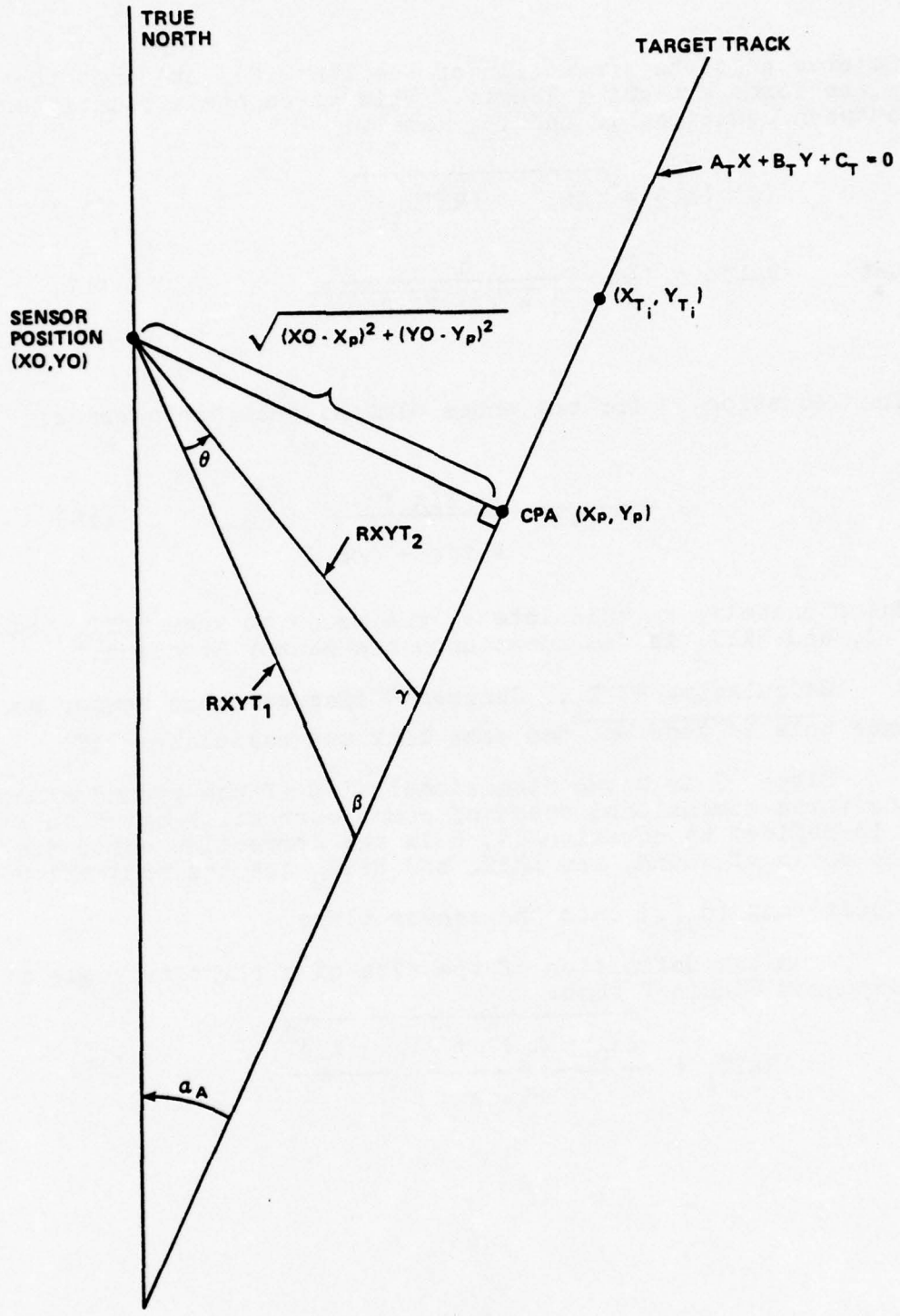



FIG. 7 CALCULATION OF $RXYT_1$

The value $\sqrt{(X_0 - Y_p)^2 + (Y_0 - Y_p)^2}$ is simply the distance between the two points $(X_0, Y_0)$ and $(X_p, Y_p)$. The point $(X_0, Y_0)$ is the sensor location which is calculated in Chapter 4. The other point $(X_p, Y_p)$ is calculated as the point on the target track closest to the sensor.

The equation of the line ($A_T X + B_T Y + C_T = 0$) can be found from any point $(X_{T_i}, Y_{T_i})$ on that line and from the slope ($\alpha_A$) of that line. From standard algebraic formulas:

$$A_T = \cos(\alpha_A) \tag{20}$$

$$B_T = \sin(\alpha_A) \tag{21}$$

$$C_T = A_T X_{T_i} - B_T Y_{T_i} \tag{22}$$

The co-ordinates of the point $(X_p, Y_p)$ on the line $A_T X + B_T Y + C_T = 0$ that is closest to the point $(X_0, Y_0)$ is given by:

$$X_p = \frac{-(A_T C_T + B_T X_0)}{(A_T^2 + B_T^2)} \tag{23}$$

$$Y_p = \frac{A_T(A_T Y_0 - B_T X_0) - B_T C_T}{(A_T^2 + B_T^2)} \tag{24}$$

Now $RXYT_1$ (equation 19) can be calculated and substituted into equations 17 and 18.

# Chapter 4

## CALCULATION OF SENSOR POSITION

At this point it is assumed that the reported bearing has been corrected and that it points from the sensor to the target at time $t_2$. The method of approach is to calculate the parameters of a line, $L_i = a_i x + b_i y + c_i$, passing through the target and along the reported bearing ray. This is done for every report. Therefore, many lines will be calculated, all of which should intersect at the sensor's position. Because of the inaccuracies in the bearing report and in the target's position it is not expected that every line will intersect exactly at the sensor's position. Figure 8 shows a typical case.

The idea is to find a point $(X_0, Y_0)$ which best fits the intersection of all the lines. A weighted least-squares algorithm is used, and the distance function is the ratio of the perpendicular distance, d, from the point $(X_0, Y_0)$ to each line over the radial distance, rxy, (where rxy is equivalent to $RXYT_2$ in Figure 6) from the target to the sensor. This ratio is the arc sine of theta, $\theta_i$, as shown in Figure 9 for one bearing report. For small $\theta$,

$$\theta_i = \frac{d_i}{rxy_i} \qquad (25)$$

Math Model for SPL

The weighted least-squares algorithm to be used requires minimizing the equation below where $W_i$ is a weighting function:

$$S = \sum_{i=1}^{n} W_{i(\theta_i)^2} \qquad (26)$$

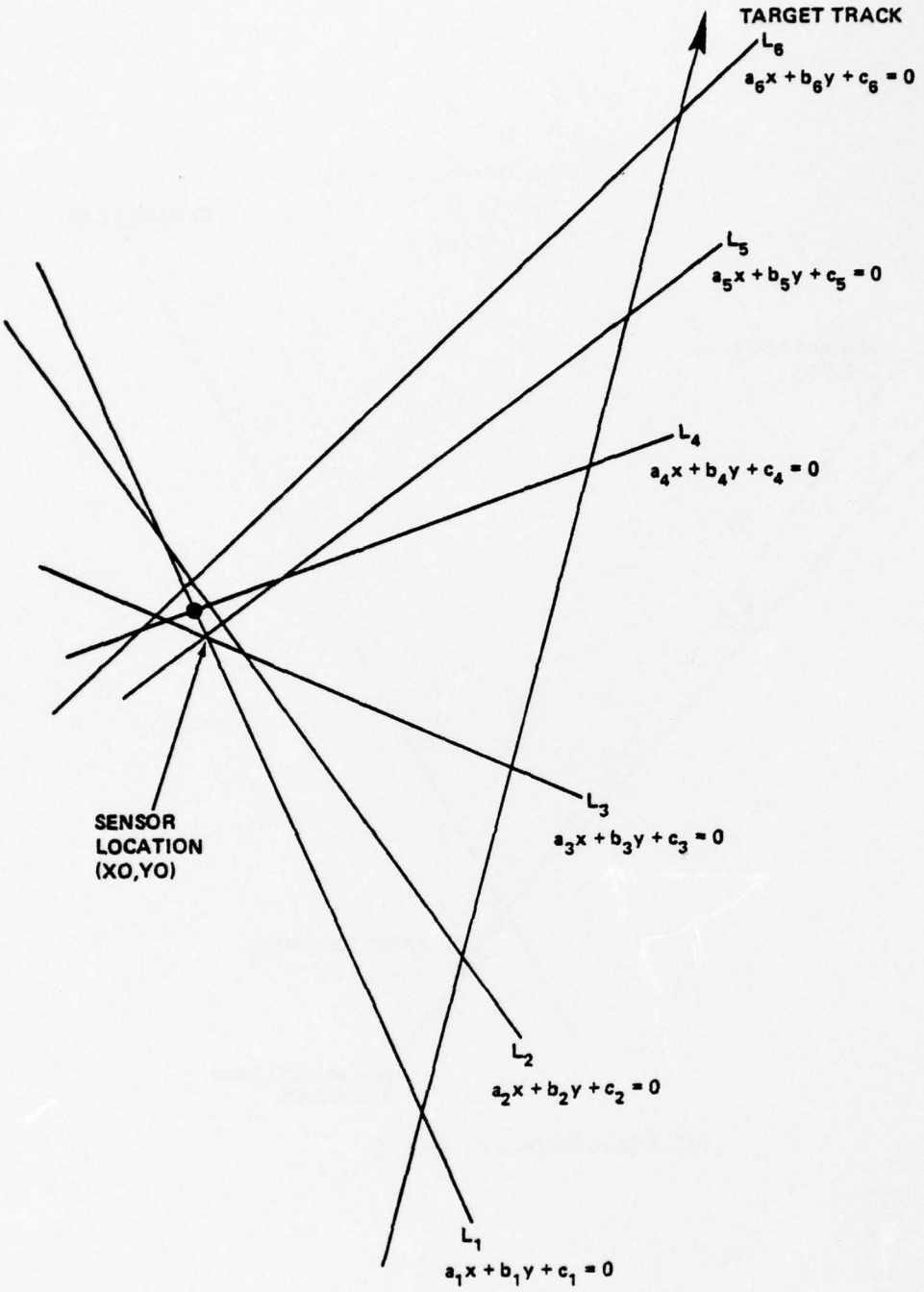



FIG. 8 SENSOR POSITION LOCATOR

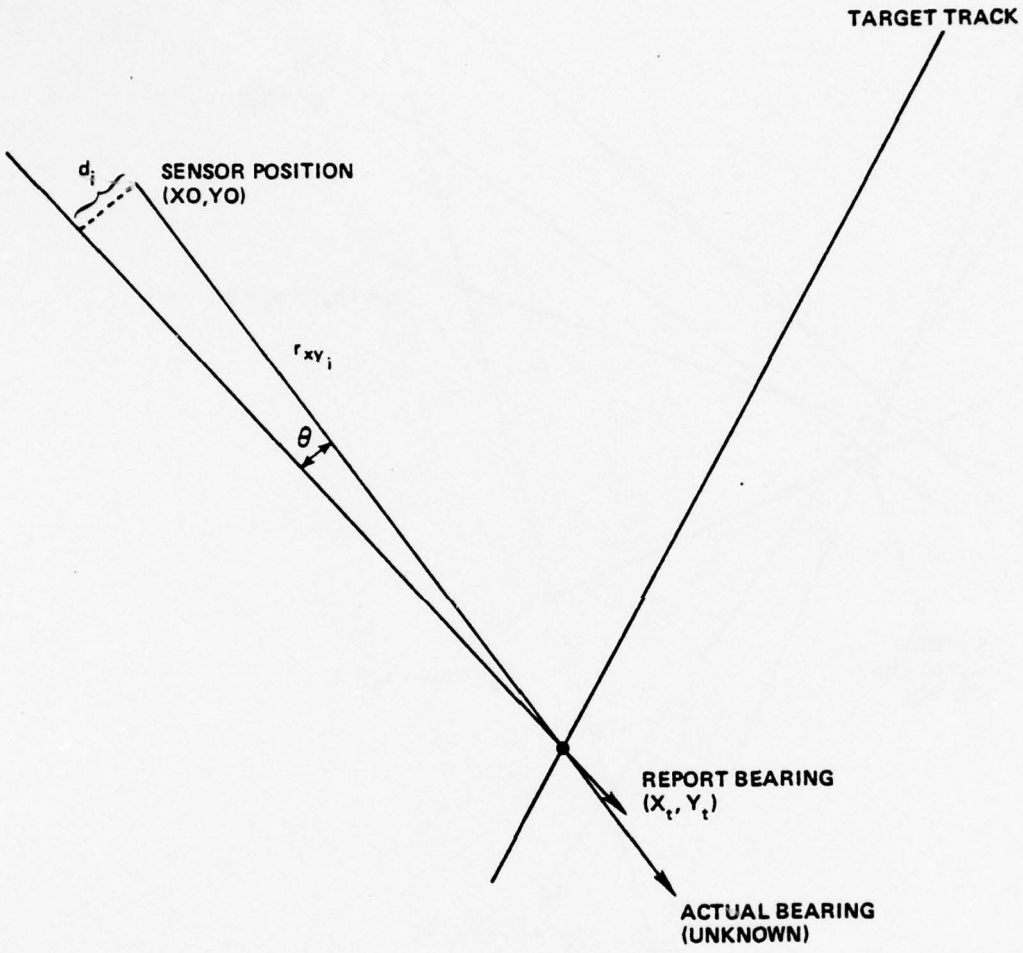



FIG. 9 DEFINITION OF THETA, $\theta$

The distance from the sensor location, point $(X_0, Y_0)$, to the line $L_i$ is given by:

$$d_i = \left| \frac{a_i X_0 = b_i Y_0 + C_i}{a_i^2 + b_i^2} \right| \qquad (27)$$

The length of the line $rxy_i$ is given below where $(X_0, Y_0)$ is the sensor location and $(X_{Ti}, Y_{Ti})$ is the $i^{th}$ target location.

$$rxy_i = \sqrt{(X_0 - X_{Ti})^2 + (Y_0 - Y_{Ti})^2} \qquad (28)$$

Substituting equations 25 and 27 into equation 26, one gets:

$$S = \sum_{i=1}^{n} \frac{W_i (a_i X_0 + b_i Y_0 + C_i)^2}{(a_i^2 + b_i^2)(rxy_i)^2} \qquad (29)$$

To minimize equation 29 one needs to take the partials with respect to $X_0$ and $Y_0$. To simplify these calculations let $X_0$ and $Y_0$ in equation 28 be the sensor position from a previous calculation $(\overline{X}_0, \overline{Y}_0)$. Then $rxy_i$ is not a function of $X_0$ and $Y_0$ and can be treated as a constant. The above simplifying assumption results in acceptable errors. Without this assumption the model cannot be brought into closed form. A more complex derivation has been worked out but not implemented here. It was abandoned because of the fast growth of complexity with little promise of any accuracy to be gained.

Taking the partials of equation 29 with respect to $X_0$ and $Y_0$ and setting them equal to zero gives the following formulas for calculation of $X_0$ and $Y_0$:

$$X_0 = \frac{\left(\sum \frac{w_i b_i c_i}{(a_i^2+b_i^2)RXY_i^2}\right)\left(\sum \frac{w_i a_i b_i}{(a_i^2+b_i^2)RXY_i^2}\right)-\left(\sum \frac{w_i a_i c_i}{(a_i^2+b_i^2)RXY_i^2}\right)\left(\sum \frac{w_i b_i^2}{(a_i^2+b_i^2)RXY_i^2}\right)}{\sum \frac{w_i a_i^2}{(a_i^2+b_i^2)RXY_i^2}\left(\sum \frac{w_i b_i^2}{(a_i^2+b_i^2)RXY_i^2}\right)-\left(\sum \frac{w_i a_i b_i}{(a_i^2+b_i^2)RXY_i^2}\right)^2} \quad (30)$$

$$Y_0 = \frac{\left(\sum \frac{w_i a_i c_i}{(a_i^2+b_i^2)RXY_i^2}\right)\left(\sum \frac{w_i a_i b_i}{(a_i^2+b_i^2)RXY_i^2}\right)-\left(\sum \frac{w_i b_i c_i}{(a_i^2+b_i^2)RXY_i^2}\right)\left(\sum \frac{w_i a_i^2}{(a_i^2+b_i^2)RXY_i^2}\right)}{\left(\sum \frac{w_i a_i^2}{(a_i^2+b_i^2)RXY_i^2}\right)\left(\sum \frac{w_i b_i^2}{(a_i^2+b_i^2)RXY_i^2}\right)-\left(\sum \frac{w_i a_i b_i}{(a_i^2+b_i^2)RXY_i^2}\right)^2} \quad (31)$$

Calculation of $A_i$, $B_i$, $C_i$

In order to evaluate equations 30 and 31 one needs to calculate the parameters $A_i$, $B_i$, and $C_i$ that form the line $A_iX + B_iY + C_i = 0$. This line passes through the target position $(XT_i, YT_i)$ and has a corrected report bearing of $d \uparrow c$ as shown in Figure 10. $d \uparrow c$ is just $\alpha t_1 - \theta$. Since the slope and one point on a line determine its equation, we find that:

$$A_i = \cos (d \uparrow c) \tag{32}$$

$$B_i = \text{Sin} (d \uparrow c) \tag{33}$$

$$C_i = A_i(XT_i) - B_i(YT_i) \tag{34}$$

Substituting these values and $rxy_i$ (equation 28) into equations 30 and 31 one can solve iteratively for the sensor position $(X_0, Y_0)$. An iterative solution is necessary since $rxy_i$ is calculated using the values of $(X_0, Y_0)$ obtained from the previous iteration.

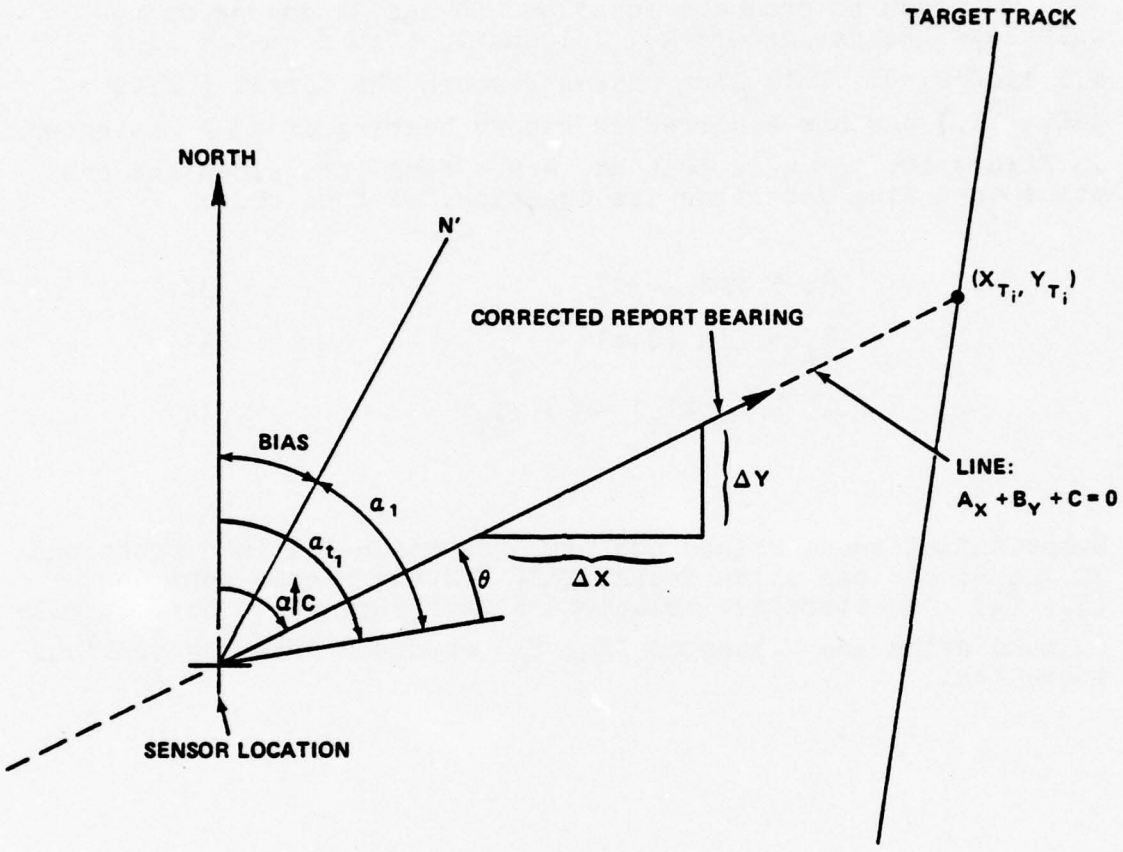



FIG. 10 CALCULATION OF $A_i$, $B_i$, $C_i$

# Chapter 5

## APPLICATION OF SPL ALGORITHM

To facilitate the calculations involved with the implementation of this algorithm, a computer program was written. This program and the flow charts associated with it are shown in Appendix B.

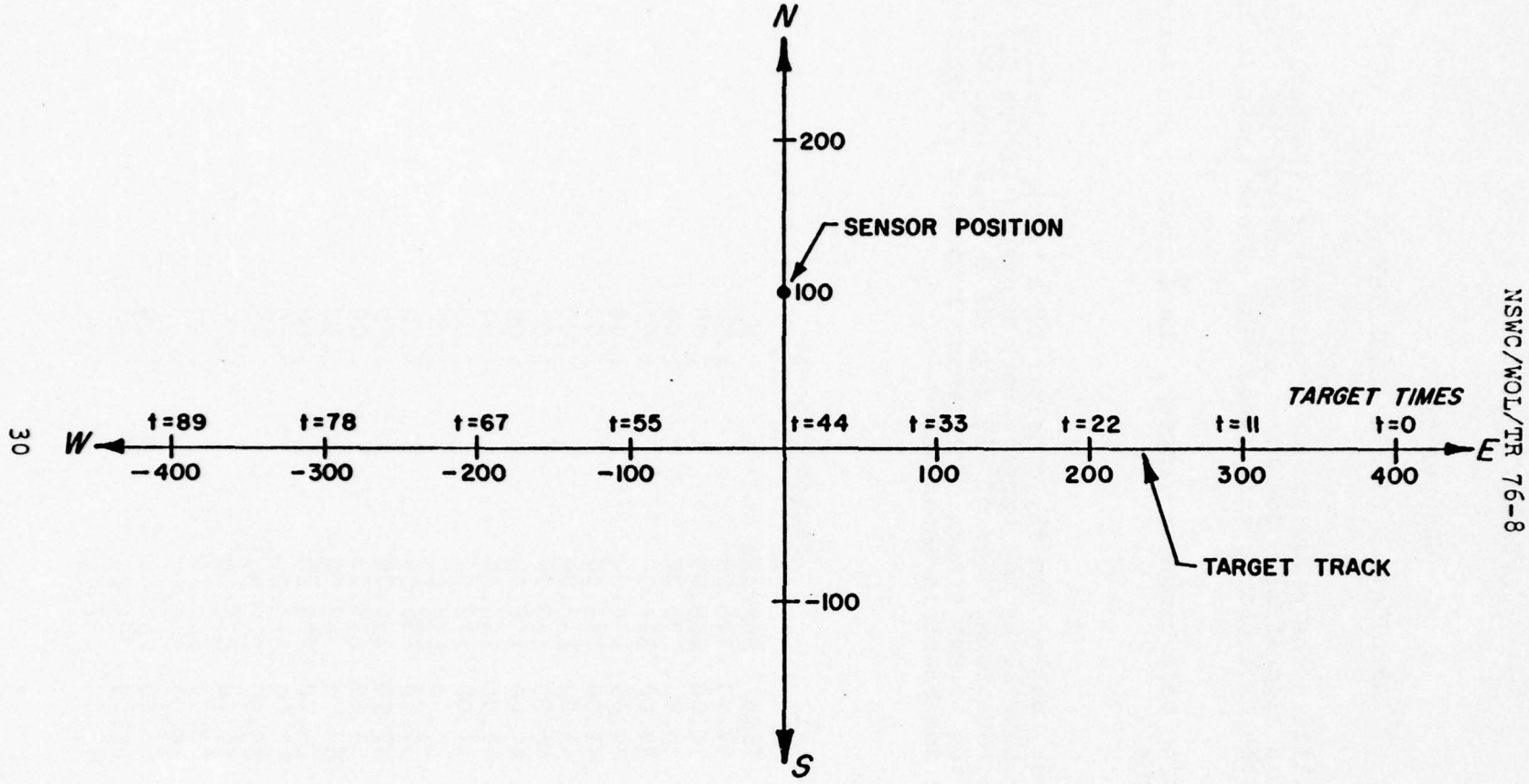
Then using this computer program, the algorithm was tested with two trial runs.

Run #1

The first, called Run #1, consists of a truck traveling at 20 miles per hour along the target track shown in Figure 11. The sensor had a rotational bias of -25 degrees and was located 100 meters from the target track. The sensor indicated 25 detections at the times and bearings shown below.

Run #1 Sensor Reports

| Time | Bearing | W |
|---|---|---|
| 34.00 | 116.00 | 1.00 |
| 35.00 | 119.00 | 1.00 |
| 38.00 | 122.00 | 1.00 |
| 39.00 | 127.00 | 1.00 |
| 40.00 | 136.00 | 1.00 |
| 41.00 | 144.00 | 1.00 |
| 45.00 | 147.00 | 1.00 |
| 46.00 | 153.00 | 1.00 |
| 47.00 | 158.00 | 1.00 |
| 48.00 | 164.00 | 1.00 |
| 49.00 | 172.00 | 1.00 |
| 50.00 | 178.00 | 1.00 |
| 51.00 | 186.00 | 1.00 |
| 52.00 | 192.00 | 1.00 |
| 53.00 | 198.00 | 1.00 |
| 54.00 | 203.00 | 1.00 |
| 55.00 | 209.00 | 1.00 |
| 57.00 | 212.00 | 1.00 |
| 58.00 | 217.00 | 1.00 |
| 59.00 | 223.00 | 1.00 |
| 63.00 | 226.00 | 1.00 |
| 65.00 | 229.00 | 1.00 |
| 66.00 | 231.00 | 1.00 |
| 67.00 | 234.00 | 1.00 |
| 69.00 | 237.00 | 1.00 |




NSWC/WOL/TR 76-8

# SIMULATED RUN #1; TARGET TRACK

FIGURE 11

The weighting factor used in the calculations was 1.00 for all points. When the test vehicle passed certain locations along the target track, called stakes, the time was noted. These times and the stake locations are also included in Figure 11.

The sensor position was calculated under three different conditions. For the first case, the rotational bias of the sensor was calculated and this value was used in calculating the location of the sensor. In the other two cases, a value for the rotational bias of the sensor was put into the program and this value was used to calculate the sensor location. The results of these three calculations are shown below where (X0, Y0) is the sensor location, Radial error $= \sqrt{(X0)^2 + (Y0)^2}$, and ALROT is the sensor angular rotation bias (either calculated or externally set equal to some value.

ROTATIONAL BIAS CALCULATED

| | X0 | Y0 | ALROT |
|---|---|---|---|
| REAL | 0 | 100.00000 | 0 |
| CALCULATED | -38.21974 | 86.88408 | -5.29993 |
| DIFFERENCE | 38.21974 | 13.11592 | 5.29993 |
| RADIAL ERROR | 40.40762 | | |

ROTATIONAL BIAS SET = 0 DEGREES

| | X0 | Y0 | ALROT |
|---|---|---|---|
| REAL | 0 | 100.00000 | 0 |
| CALCULATED | -50.52054 | 88.81540 | 0 |
| DIFFERENCE | 50.52054 | 11.18460 | 0 |
| RADIAL ERROR | 51.74380 | | |

ROTATIONAL BIAS SET -25 DEGREES

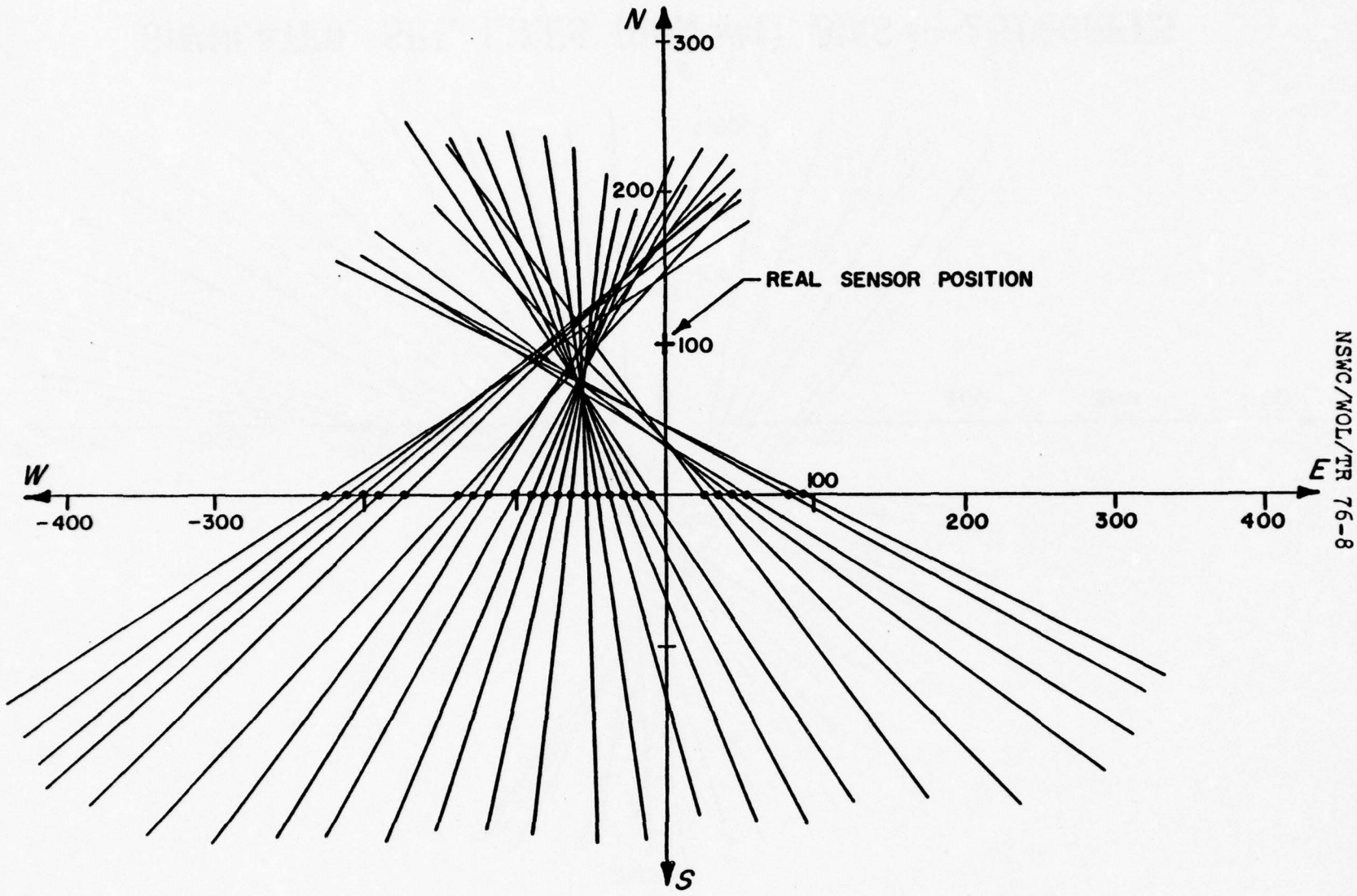
| | X0 | Y0 | ALROT |
|---|---|---|---|
| REAL | 0 | 100.00000 | -25.00000 |
| CALCULATED | -3.74224 | 103.15131 | -25.00000 |
| DIFFERENCE | 3.74224 | -3.15131 | 0 |
| RADIAL ERROR | 4.89235 | | |

The Radial Error is lowest for the case where the rotational bias was set at -25 degrees, which indicates that the best results were achieved in this case. This is true because the actual rotational bias of the sensor was -25 degrees.

Figures 12 and 13 graphically depict the two cases where the rotational bias was externally set to either 0 or -25 degrees. In these figures lines were drawn from the vehicle location (whenever the sensor made a report) towards the sensor along a corrected bearing. This corrected bearing was just the bearing reported by the sensor plus $180^{\circ}$ minus the rotational bias of the sensor.

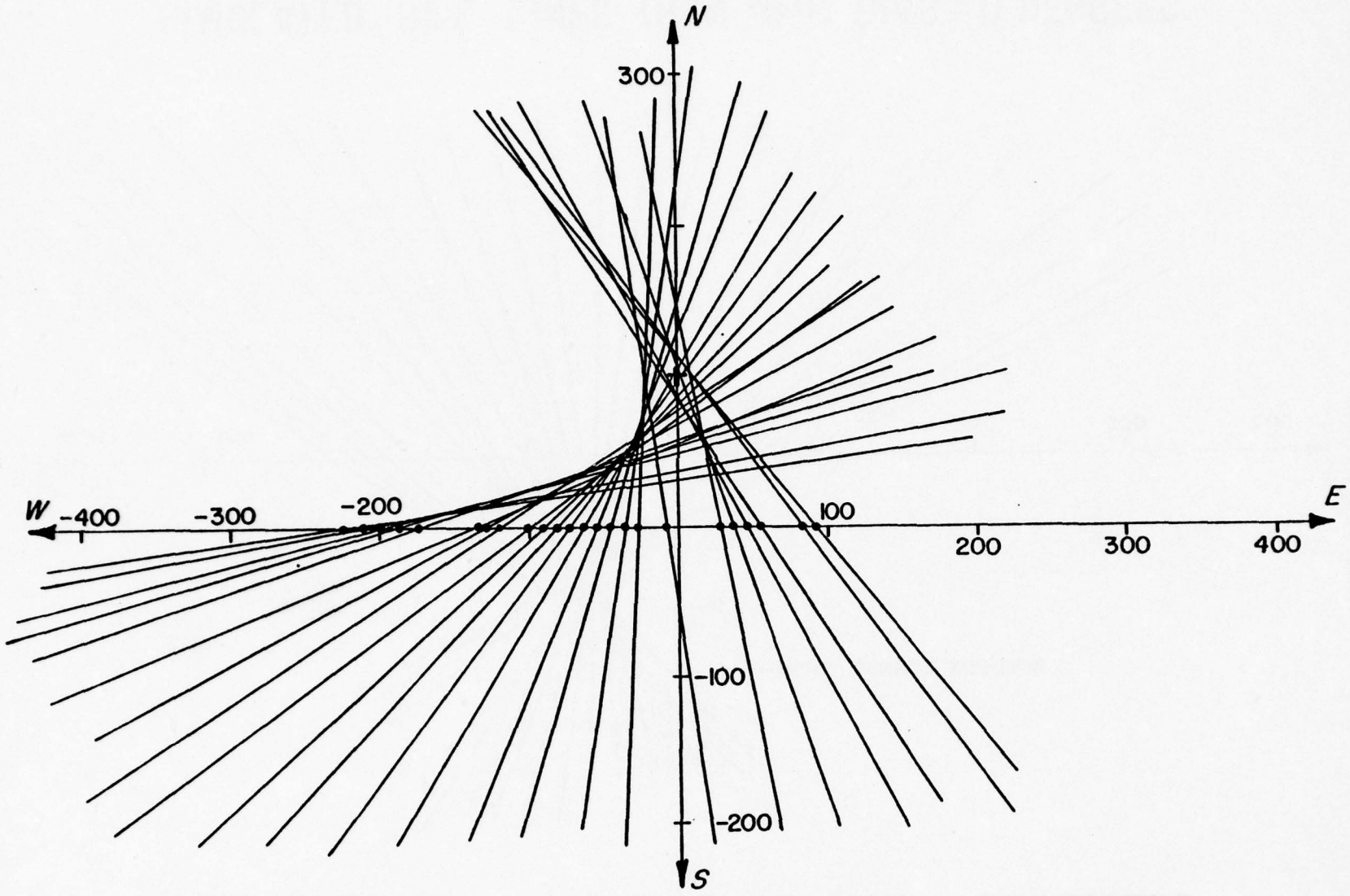
Run #2

The second run was very similar to the first. This time a truck was again driven by the sensor at a CPA of 100 meters, and with a velocity of 9 meters/second. However, this time the sensor had a rotation bias of 9 degrees, and the sensor was placed on the other side of the target track. Figure 14 shows the target track and the times that the vehicle passed the different stake locations.



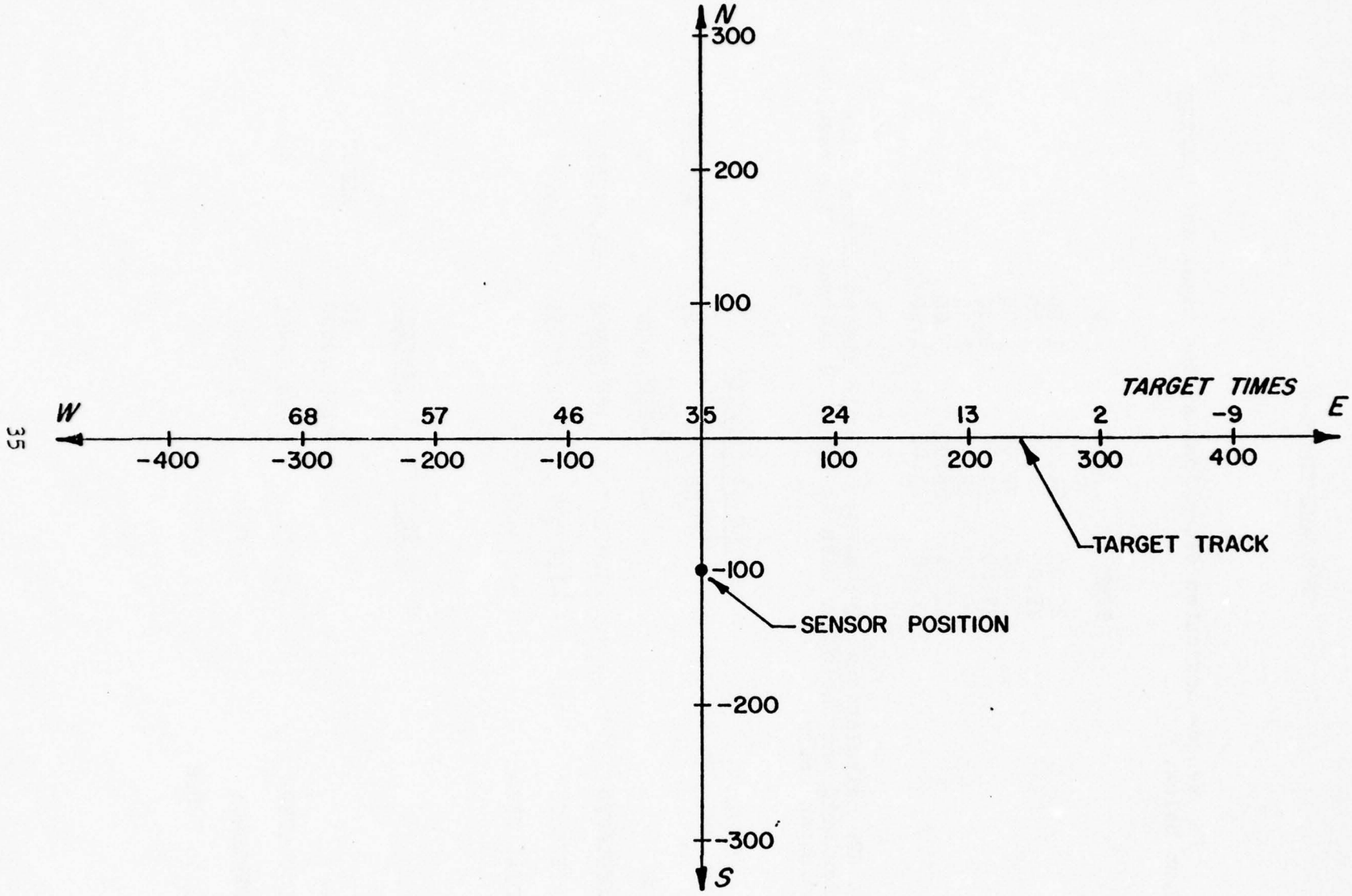

SIMULATED SPL LINES (RUN #1); BIAS = 0 DEGREES

FIGURE 12




SIMULATED SPL LINES (RUN #1); BIAS = −25 DEGREES

FIGURE 13




# SIMULATED RUN#2; TARGET TRACK

FIGURE 14

The sensor indicated detections at the times and bearings shown below.

| TIME | BEARING | W |
|---|---|---|
| 23.00 | 51.00 | 1.00 |
| 25.00 | 46.00 | 1.00 |
| 28.00 | 43.00 | 1.00 |
| 29.00 | 37.00 | 1.00 |
| 30.00 | 34.00 | 1.00 |
| 31.00 | 26.00 | 1.00 |
| 32.00 | 23.00 | 1.00 |

Two SPL calculations were made; one with the rotational bias calculated and the other with it set at 0 degrees. The results are shown below:

BIAS CALCULATED

| | X0 | Y0 | ALROT |
|---|---|---|---|
| REAL | 0 | -100.00000 | 0 |
| CALCULATED | 1.47177 | -67.59963 | -23.99265 |
| DIFFERENCE | -1.47177 | -32.40037 | 23.99265 |
| RADIAL ERROR | 32.43378 | | |

BIAS SET = 0 DEGREES

| | X0 | Y0 | ALROT |
|---|---|---|---|
| REAL | 0 | -100.00000 | 0 |
| CALCULATED | -3.67670 | -86.62461 | 0 |
| DIFFERENCE | 3.67670 | -11.37539 | 0 |
| RADIAL ERROR | 11.95482 | | |

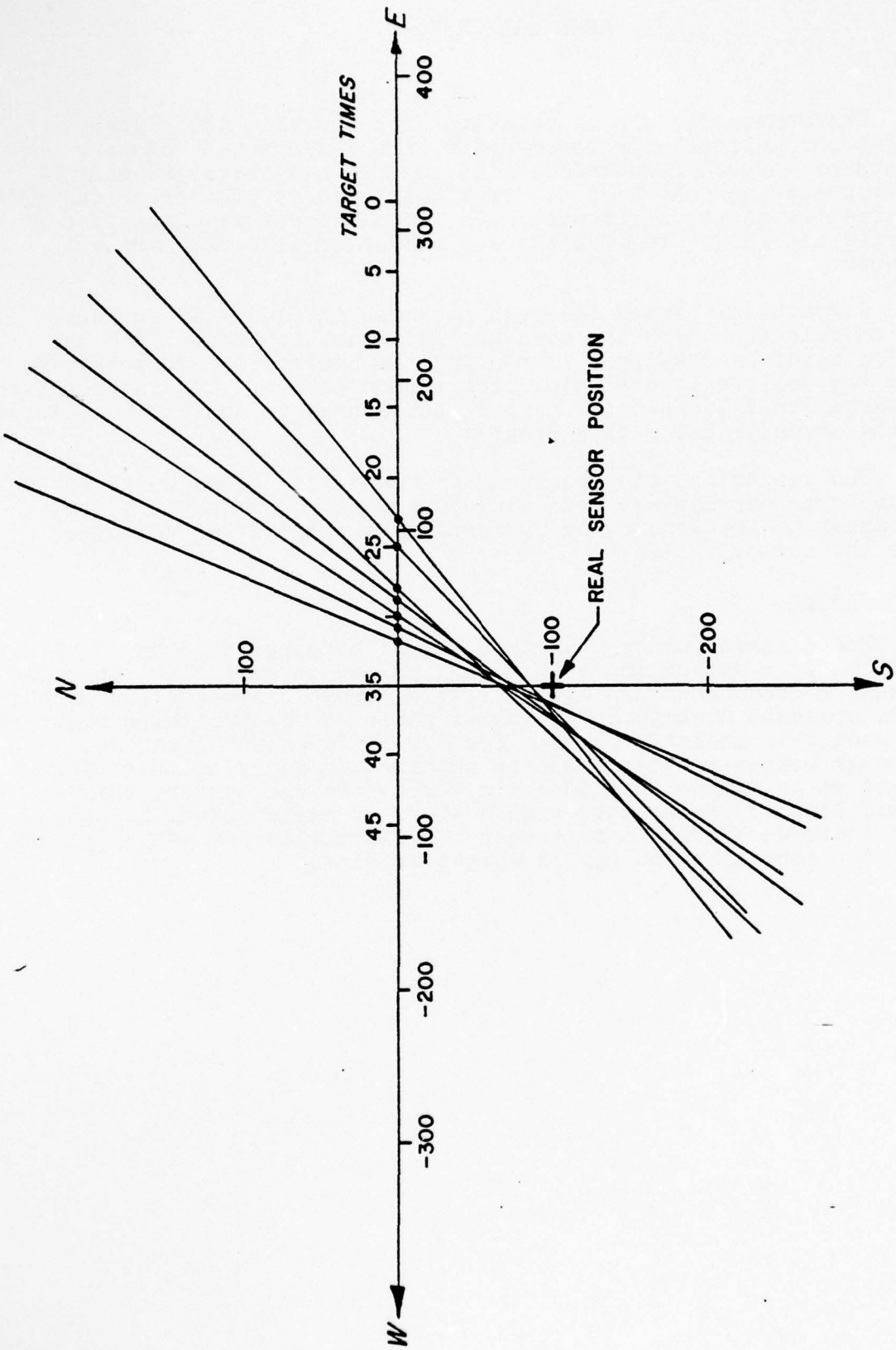
The reason the SPL calculation with the bias calculated was so poor was because the reports were few and very close together. A good bias calculation occurs when there are many reports and when the reports are widely spaced. The error occurs because the angle difference with close reports is often just a bearing bin shift and is not truly representative of bearing movement.

A graphical sensor location is shown in Figure 15 for the SPL calculation where the bias was set equal to zero. This was accomplished by drawing a line along the corrected report bearing from the vehicle location for each sensor report. The intersection of these lines locates the sensor position as is done arithmetically in the computer using this program.

The sensor location accuracy of these runs could be improved by applying appropriate weighting functions to the data, adding a compass to the sensor, or by running several target vehicles past the sensor.

## Conclusions

For a sensor which reports a target bearing angle in addition to a detection, this algorithm can determine the location of that sensor after deployment. This report discusses the method used, presents a computer program capable of accomplishing the task, and does this analysis for two sensors of "unknown" location. Although better results could be obtained by applying weighting functions to the data or adding a compass to the sensor, this method located two sensors within 40 ft of their actual location. This location algorithm increases the flexibility of sensor placement for sensors which report target bearing.




SIMULATED SPL LINES (RUN #2); BIAS = 0 DEGREES

FIGURE 15

# APPENDIX A

## ERROR ANALYSIS

There are many factors which contribute to errors in accurately locating a sensor. These include:

1. Calculation of the rotational bias of the sensor, BIAS
2. Gaussian Noise, XK
3. Target course error, ΔALA
4. Target X position error, ΔX
5. Target Y position error, ΔY
6. Target Z position error, ΔZ
7. Target Velocity error, ΔV
8. Temperature error, Δ temp
9. The bearing resolution of the sensor, BINS

Errors due to these sources were computed for 58 simulated runs under four conditions. These conditions were:

1. Rotational bias calculated, one target vehicle
2. Rotational bias set, one target vehicle
3. Rotational bias calculated, two target vehicles
4. Rotational bias set, two target vehicles

The results of these runs are shown in Table 1. Unless specified the input parameters were:

BIAS = 0 degrees

ΔXK = 0

ΔALA = 0 degrees

ΔX = 0 meters

ΔY = 0 meters

$\Delta_Z$ = 0 meters

V = 0 meters/sec

temp = 0 degrees

number of Bins = 100,000 (runs 1-47)

number of Bins = 256 (runs 53-58)

These sensor location errors were calculated for sensor positions 100, 500, and 1000 meters from the target track. The vehicle was traveling at 50 m/s at an altitude of 200 m. The range of the target track was 3000 meters. In Table 1, $DR_{100}$, $DR_{500}$, and $DR_{1000}$ refer to the distance of the sensor from the test track. All sensor location errors are in meters displayed from their actual location.

Bias Errors

In Table 1 the bias errors only affected the cases where the bias was set. In those cases, the bias was set equal to that error amount and the sensor position error was calculated. In the cases where the bias was calculated, it was calculated correctly as all other parameters were "error free." Using this correctly calculated bias the sensor position was accurate. Figure 16 shows the effect of bias errors on the SPL algorithm. The results are about the same for one or two targets, but do depend on the distance of the sensor from the target track.

Gaussian Noise

The signal received by the sensor from some target source contains noise. This noise affects the bearing angle reported by the sensor.

All noise is assumed to be Gaussian added to the real report bearing. This noise consists of an average value (Mean) and a standard deviation (Sigma). The mean is just the bias. Sigma is a function of the signal to noise ratio. The higher the ratio the lower the error. The following relationship has been observed:

$$\text{Sigma} = \frac{K_1}{S/N} \qquad (35)$$

S/N is inversely proportional to the radial distance from the sensor to the target. Equation 36 depicts this relationship. where $S/N_{cpa}$ is the signal to noise ratio at the closest point of approach (cpa).

## TABLE 1 ERROR ANALYSIS OF SENSOR POSITION LOCATER

| RUN NUMBER | ERROR TYPE | ERROR AMOUNT | ONE TARGET | | | | | | TWO TARGETS | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | WITH CALCULATED BIAS | | | WITH BIAS SET | | | WITH CALCULATED BIAS | | | WITH BIAS SET | | |
| | | | $DR_{100}$ | $DR_{500}$ | $DR_{1000}$ | $DR_{100}$ | $DR_{500}$ | $DR_{1000}$ | $DR_{100}$ | $DR_{500}$ | $DR_{1000}$ | $DR_{100}$ | $DR_{500}$ | $DR_{1000}$ |
| 1 | NONE | - | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | BIAS | 20° | - | - | - | 96 | 385 | 517 | - | - | - | 282 | 347 | 515 |
| 3 | BIAS | 10° | - | - | - | 79 | 145 | 229 | - | - | - | 68 | 141 | 258 |
| 4 | BIAS | 5° | - | - | - | 21 | 63 | 108 | - | - | - | 23 | 64 | 129 |
| 5 | BIAS | 1° | - | - | - | 3 | 12 | 21 | - | - | - | 2.6 | 12 | 26 |
| 6 | BIAS | 1° | - | - | - | 3 | 12 | 21 | - | - | - | 2.5 | 12 | 26 |
| 7 | BIAS | 5° | - | - | - | 26 | 65 | 107 | - | - | - | 17 | 62 | 126 |
| 8 | BIAS | 10° | - | - | - | 82 | 149 | 225 | - | - | - | 50 | 128 | 241 |
| 9 | BIAS | 20° | - | - | - | 299 | 362 | 519 | - | - | - | 463 | 342 | 487 |
| 10 | XK | 0.0001 | 0.2 | 0.8 | 2.4 | 0.1 | 0.5 | 1.5 | 0.2 | 0.7 | 0.3 | 0.1 | 0.3 | 1.1 |
| 11 | XK | 0.0010 | 12.1 | 9.1 | 28.9 | 11.6 | 6.5 | 16.8 | 7.6 | 9.3 | 35 | 4.5 | 9.0 | 16.4 |
| 12 | XK | 0.0050 | 416 | 216 | 509 | 265 | 150 | 363 | 127 | 32 | 203 | 337 | 33 | 85 |
| 13 | XK | 0.0100 | 6992 | 2074 | 901 | 120 | 1904 | 4023 | 1395 | 1244 | 289 | 476 | 2154 | 876 |
| 14 | ΔALA | 20° | 869 | 885 | 935 | 960 | 546 | 1263 | 901 | 1038 | 1236 | 2961 | 1301 | 1565 |
| 15 | ΔALA | 10° | 436 | 444 | 469 | 514 | 349 | 560 | 444 | 447 | 491 | 521 | 505 | 549 |
| 16 | ΔALA | 5° | 218 | 222 | 235 | 235 | 199 | 255 | 209 | 177 | 188 | 212 | 175 | 154 |
| 17 | ΔALA | 1° | 43 | 44 | 47 | 44 | 44 | 47 | 34 | 29 | 32 | 32 | 18 | 8.8 |
| 18 | ΔALA | -1° | 43 | 44 | 47 | 42 | 46 | 45 | 38 | 30 | 33 | 37 | 18 | 9.3 |
| 19 | ΔALA | -5° | 218 | 222 | 235 | 195 | 248 | 205 | 210 | 178 | 189 | 228 | 181 | 159 |
| 20 | ΔALA | -10° | 436 | 444 | 469 | 355 | 538 | 358 | 403 | 435 | 445 | 503 | 636 | 489 |
| 21 | ΔALA | -20° | 869 | 885 | 935 | 699 | 1190 | 546 | 884 | 1020 | 1040 | 504 | 1171 | 1332 |
| 22 | ΔX | 10 M | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 |
| 23 | ΔX | 5 M | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 |
| 24 | ΔX | 5 M | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 |
| 25 | ΔX | -10 M | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 |
| 26 | ΔY | 10 M | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 |
| 27 | ΔY | 5 M | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 |
| 28 | ΔY | -5 M | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 |
| 29 | ΔY | -10 M | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 |
| 30 | ΔZ | 100 M | 13.4 | 6.5 | 4.0 | 13.4 | 6.0 | 3.4 | 15.0 | 6.1 | 3.9 | 14.9 | 5.9 | 3.7 |
| 31 | ΔZ | 50 M | 6.6 | 3.0 | 1.8 | 6.6 | 2.8 | 1.5 | 6.6 | 2.8 | 1.8 | 6.6 | 2.7 | 1.7 |
| 32 | ΔZ | 10 M | 1.3 | 0.5 | 0.3 | 1.3 | 0.5 | 0.3 | 1.2 | 0.5 | 0.3 | 1.2 | 0.5 | 0.3 |
| 33 | ΔZ | -10 M | 1.3 | 0.5 | 0.3 | 1.3 | 0.5 | 0.3 | 1.2 | 0.5 | 0.3 | 1.2 | 0.5 | 0.3 |
| 34 | ΔZ | -50 M | 6.2 | 2.4 | 1.4 | 6.1 | 2.2 | 1.2 | 5.3 | 2.2 | 1.4 | 5.3 | 2.2 | 1.3 |
| 35 | ΔZ | -100 M | 11.6 | 4.1 | 2.4 | 11.5 | 3.8 | 2.1 | 9.6 | 3.7 | 2.4 | 9.5 | 3.7 | 2.2 |
| 36 | ΔV | 20 M/S | 995 | 987 | 1027 | 994 | 975 | 995 | 1035 | 1186 | 1410 | 1034 | 1173 | 1378 |
| 37 | ΔV | 10 M/S | 498 | 495 | 517 | 498 | 490 | 503 | 516 | 602 | 741 | 515 | 597 | 733 |
| 38 | ΔV | 5 M/S | 249 | 248 | 259 | 249 | 246 | 252 | 252 | 307 | 390 | 252 | 305 | 389 |
| 39 | ΔV | -5 M/S | 249 | 249 | 261 | 249 | 248 | 256 | 260 | 301 | 385 | 290 | 301 | 385 |
| 40 | ΔV | -10 M/S | 499 | 500 | 523 | 499 | 497 | 514 | 516 | 856 | 749 | 516 | 612 | 751 |
| 41 | ΔV | -20 M/S | 999 | 1005 | 1054 | 999 | 1000 | 1040 | 1038 | 1210 | 1458 | 1039 | 1205 | 2635 |
| 42 | ΔTEMP | 20°C | 1.2 | 2.6 | 4.0 | 1.3 | 2.3 | 6.0 | 1.1 | 3.0 | 6.1 | 1.1 | 3.2 | 6.4 |
| 43 | ΔTEMP | 10°C | 0.6 | 1.3 | 2.0 | 0.7 | 1.7 | 3.1 | 0.6 | 1.5 | 3.1 | 0.6 | 1.6 | 3.3 |
| 44 | ΔTEMP | 5°C | 0.3 | 0.7 | 1.0 | 0.3 | 0.9 | 1.5 | 0.3 | 0.8 | 1.6 | 0.3 | 0.8 | 1.7 |
| 45 | ΔTEMP | 5°C | 0.3 | 0.7 | 1.0 | 0.3 | 0.9 | 1.6 | 0.3 | 0.8 | 1.6 | 0.3 | 0.8 | 1.7 |
| 46 | ΔTEMP | -10°C | 0.6 | 1.4 | 2.1 | 0.7 | 1.8 | 3.2 | 0.6 | 1.6 | 3.2 | 0.6 | 1.7 | 3.4 |
| 47 | ΔTEMP | -20°C | 1.3 | 2.8 | 4.3 | 1.4 | 3.6 | 6.5 | 1.2 | 3.2 | 6.5 | 1.2 | 3.4 | 6.9 |
| 48 | # BINS | 1024 | 0.2 | 0.2 | 1.3 | 0.1 | 0.2 | 1.3 | 0.1 | 0.1 | 0.3 | 0.0 | 0.2 | 0.2 |
| 49 | # BINS | 512 | 0.6 | 1.9 | 2.6 | 0.4 | 1.8 | 2.0 | 0.3 | 1.0 | 0.5 | 0.3 | 0.6 | 0.3 |
| 50 | # BINS | 256 | 0.6 | 3.7 | 2.8 | 0.4 | 0.1 | 2.5 | 0.6 | 2.3 | 2.3 | 0.2 | 0.0 | 0.5 |
| 51 | # BINS | 128 | 0.6 | 5.7 | 18.4 | 0.1 | 5.0 | 11.6 | 0.6 | 3.2 | 10.3 | 0.1 | 1.2 | 2.1 |
| 52 | # BINS | 64 | 5.4 | 23.4 | 12.9 | 3.1 | 22.7 | 26.0 | 2.6 | 6.3 | 24.3 | 1.8 | 5.3 | 23.1 |
| 53 | XK | 0.0001 | 0.2 | 6.0 | 9.4 | 0.3 | 2.5 | 5.2 | 0.3 | 1.3 | 4.6 | 0.2 | 1.8 | 0.9 |
| 54 | XK | 0.0010 | 7.5 | 9.3 | 32 | 6.4 | 5.7 | 32 | 4.4 | 6.4 | 16 | 3.0 | 5.9 | 10.6 |
| 55 | XK | 0.0050 | 623 | 275 | 690 | 146 | 228 | 636 | 300 | 148 | 123 | 413 | 116 | 114 |
| 56 | XK | 0.0100 | 1670 | 505 | 5284 | 5687 | 689 | 2266 | 4261 | 2763 | 203 | 710 | 352 | 101 |
| 57 | BIAS | 5° | 1.0 | 5.2 | 4.0 | 239 | 63.7 | 107.5 | 0.4 | 2.1 | 1.9 | 22.5 | 63.4 | 128 |
| 58 | BIAS | 1° | 0.4 | 10.2 | 5.0 | 37 | 12.7 | 21.2 | 0.6 | 2.4 | 4.6 | 2.7 | 12.3 | 25.7 |

# AFFECTS OF BIAS ON SPL ERRORS

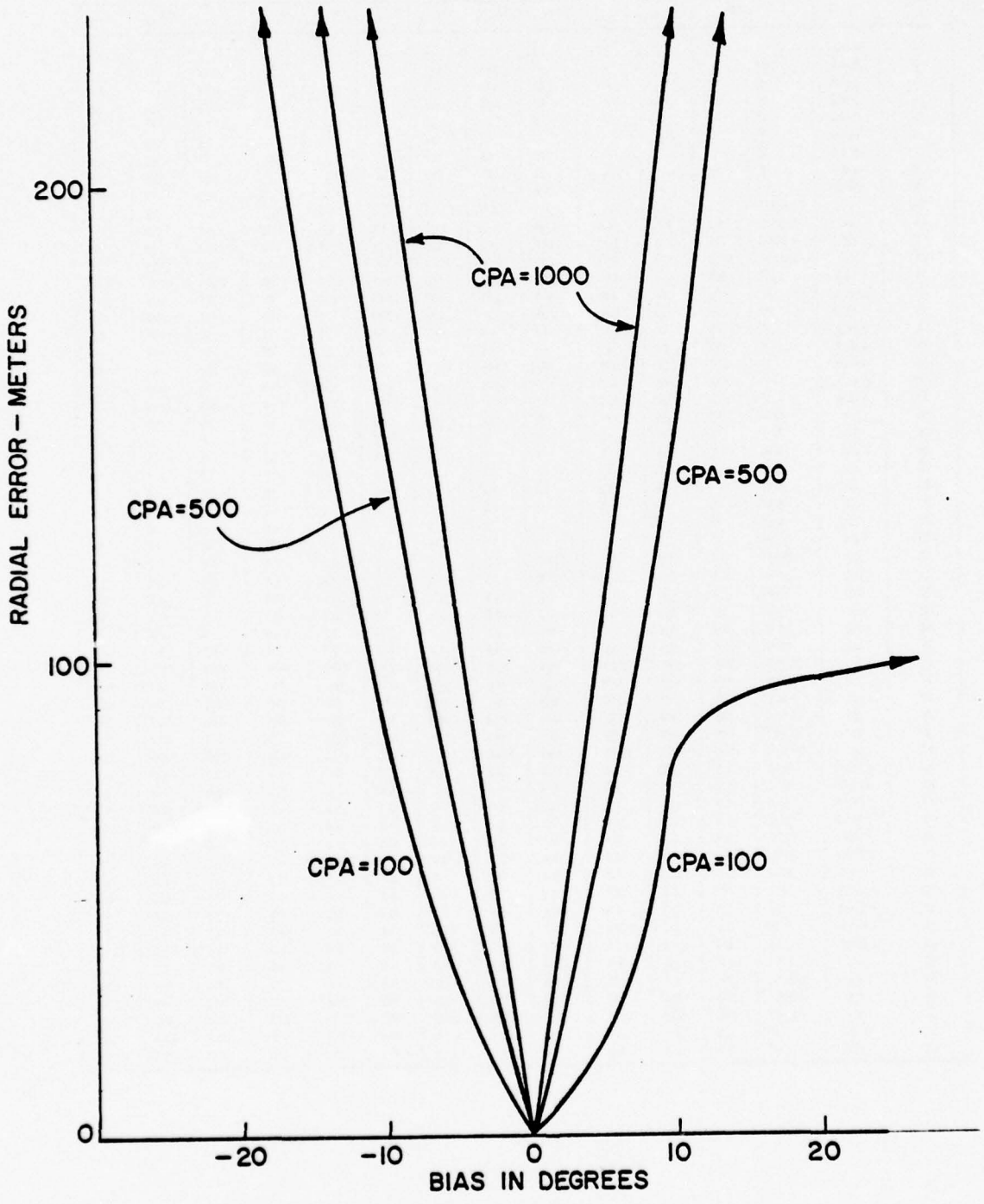





FIGURE 16

$$S/N = f(D) = \frac{K_2 + S/N_{cpa}}{\sqrt{D + 1}} \qquad (36)$$

Combining equations 35 and 36 gives:

$$Sigma = K\sqrt{D + 1} \qquad (37)$$

For the error analysis values of .0001, .001, .005, and .01 were used for K. Figure 17 plots the radial error (meters) of the sensor as a function of K. From Figure 17 it is noted that repeating the calculations for a second target greatly reduces the error from this source.

Target Course Error

This error comes about when the test vehicle is traveling in a direction slightly different than it is thought to be traveling. Figure 18 plots these errors as a function of course error (in degrees).

ΔX, ΔY Errors

This type of error occurs when the target vehicle is at a point different than it is thought to be. Equation 38 gives the radial error of the sensor as a function of the ΔX, ΔY displacement.

$$\text{radial error} = \sqrt{(\Delta X)^2 + (\Delta Y)^2} \qquad (38)$$

The amount of this error is independent of whether the rotational bias is calculated or set equal to some value in the SPL algorithm, and it is independent of the number of test vehicles provided the ΔX, ΔY displacement of each vehicle is the same.

ΔZ Errors

This error results from inaccurate estimates of the differences in altitude between the target track plane and the sensor plane. It is independent of the number of targets used to sight in the

# AFFECTS OF K (GAUSSIAN NOISE) ON SPL ERRORS

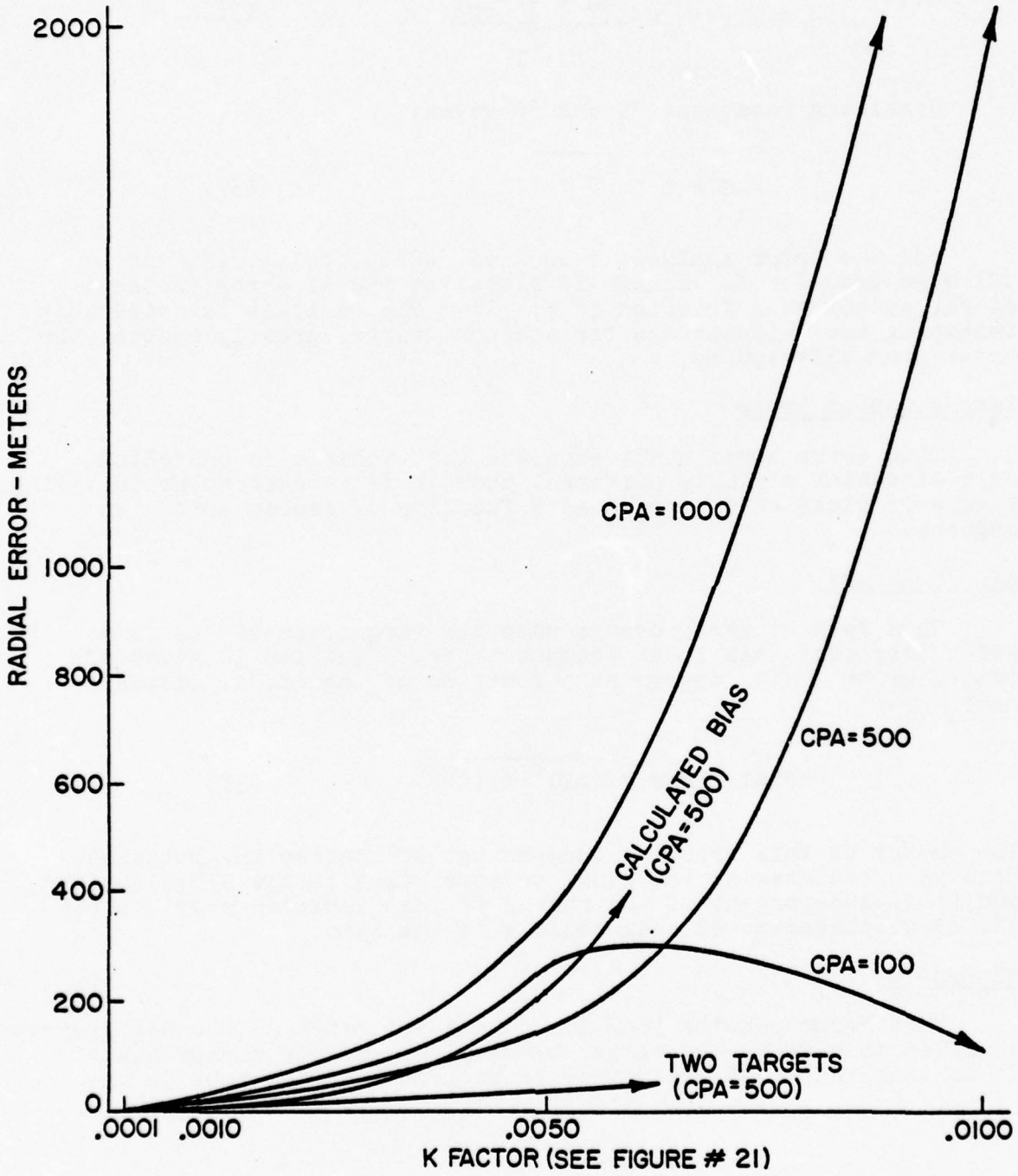





FIGURE 17

# AFFECTS OF Δ COURSE ON SPL ERRORS

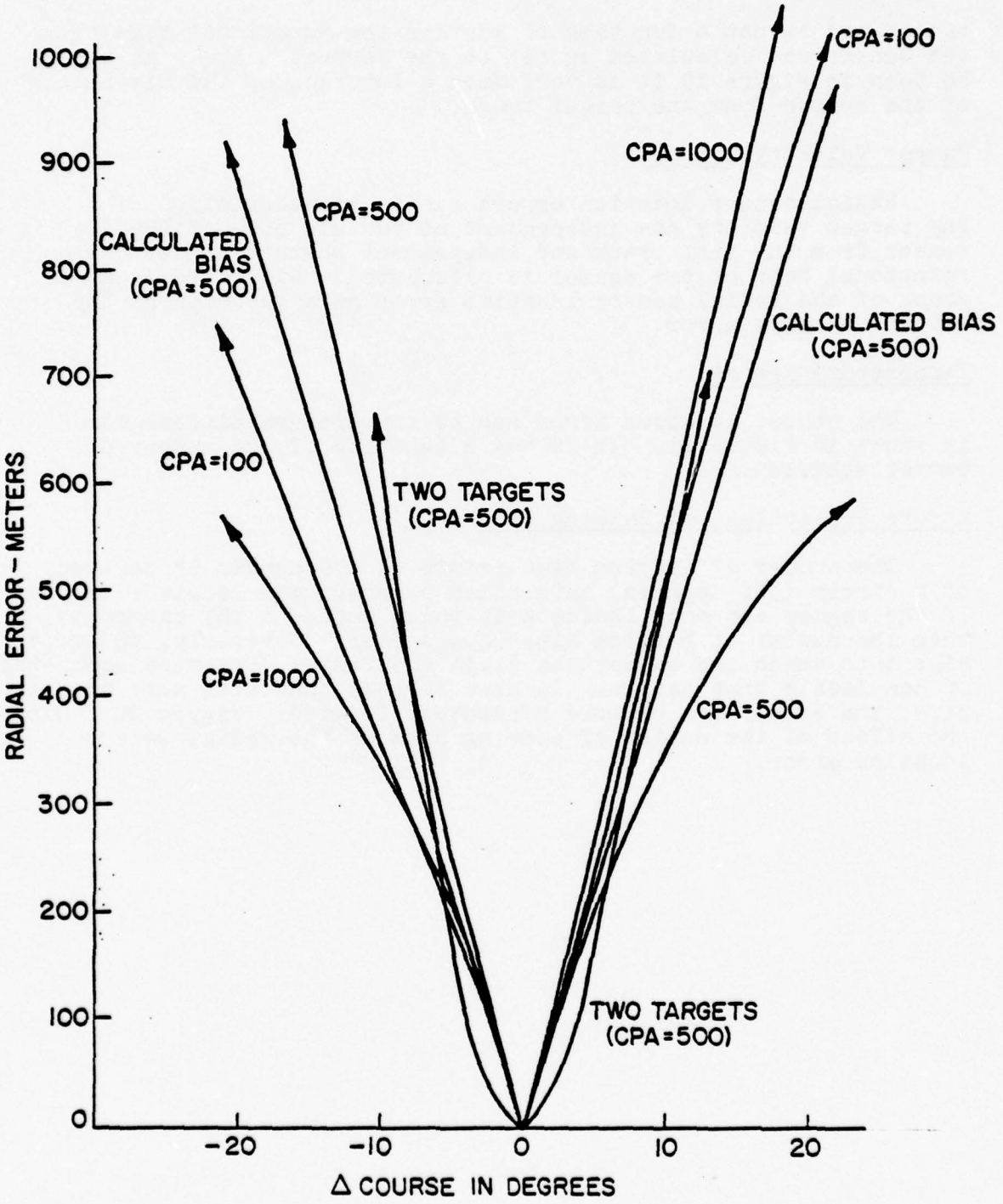





FIGURE 18

sensor and is not a function of whether the rotational bias of the sensor was calculated or set to the correct value. As can be seen in Figure 19 it is very much a function of the distance of the sensor from the target track.

Target Velocity Errors

Radial sensor location errors due to miscalculation of the target velocity are independent of the distance of the sensor from the test track and independent whether or not the rotational bias of the sensor is calculated. Figure 20 is a graph of the radial sensor location error as a function of the target velocity error.

Temperature Errors

The sensor location error due to temperature differences is shown in Figure 21. It is not a function of the number of target vehicles.

Errors Due to Lack of Bearing Bins

The number of bearing bins refers to the number of sections of a circle (360 degrees) into which a sensor can locate a target. If the sensor can only indicate in which quadrant the target is, then the number of bearing bins equals four. Obviously, the more bins into which the sensor can place the target, the more accurately it can locate that target. It also follows that with more bearing bins, the sensor can be more accurately located. Figure 22 depicts the affect of the number of bearing bins on the radial sensor location error.

# AFFECTS OF ΔZ ON SPL ERRORS

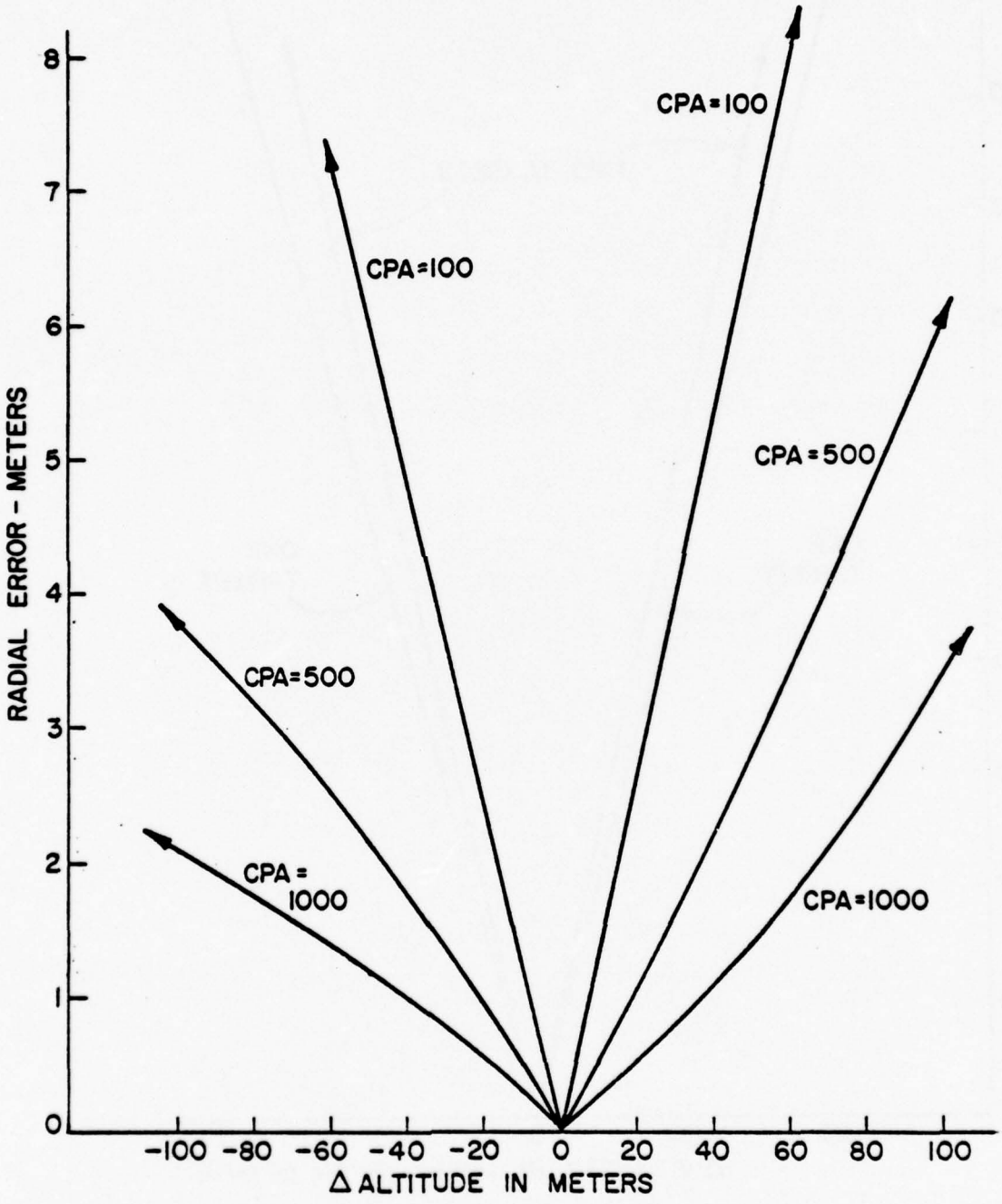





FIGURE 19

# AFFECTS OF Δ VELOCITY ON SPL ERRORS






FIGURE 20

# AFFECTS OF Δ TEMPERATURE ON SPL ERRORS

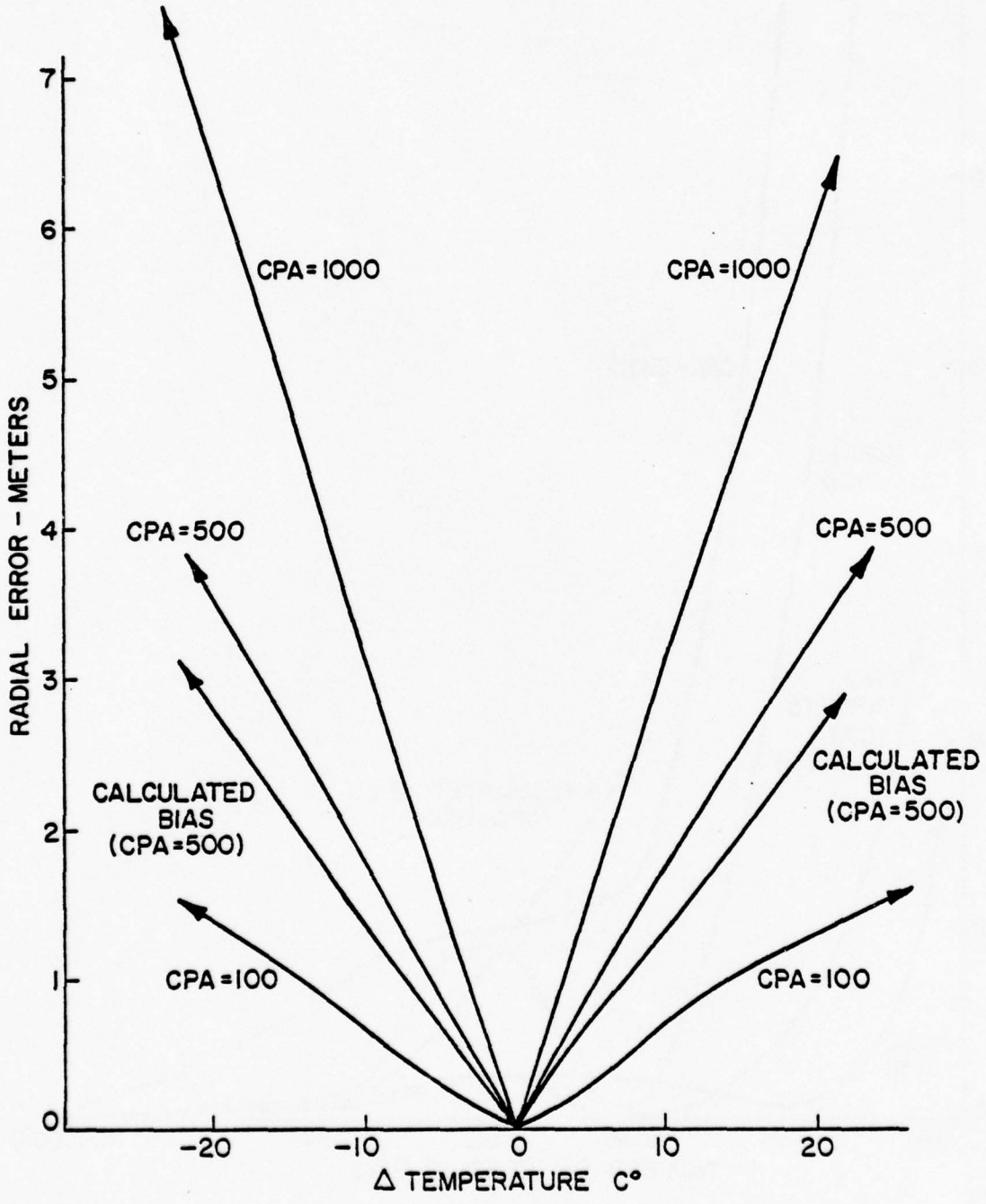





FIGURE 21

# AFFECTS OF NUMBER OF BEARING BINS ON SPL ERRORS

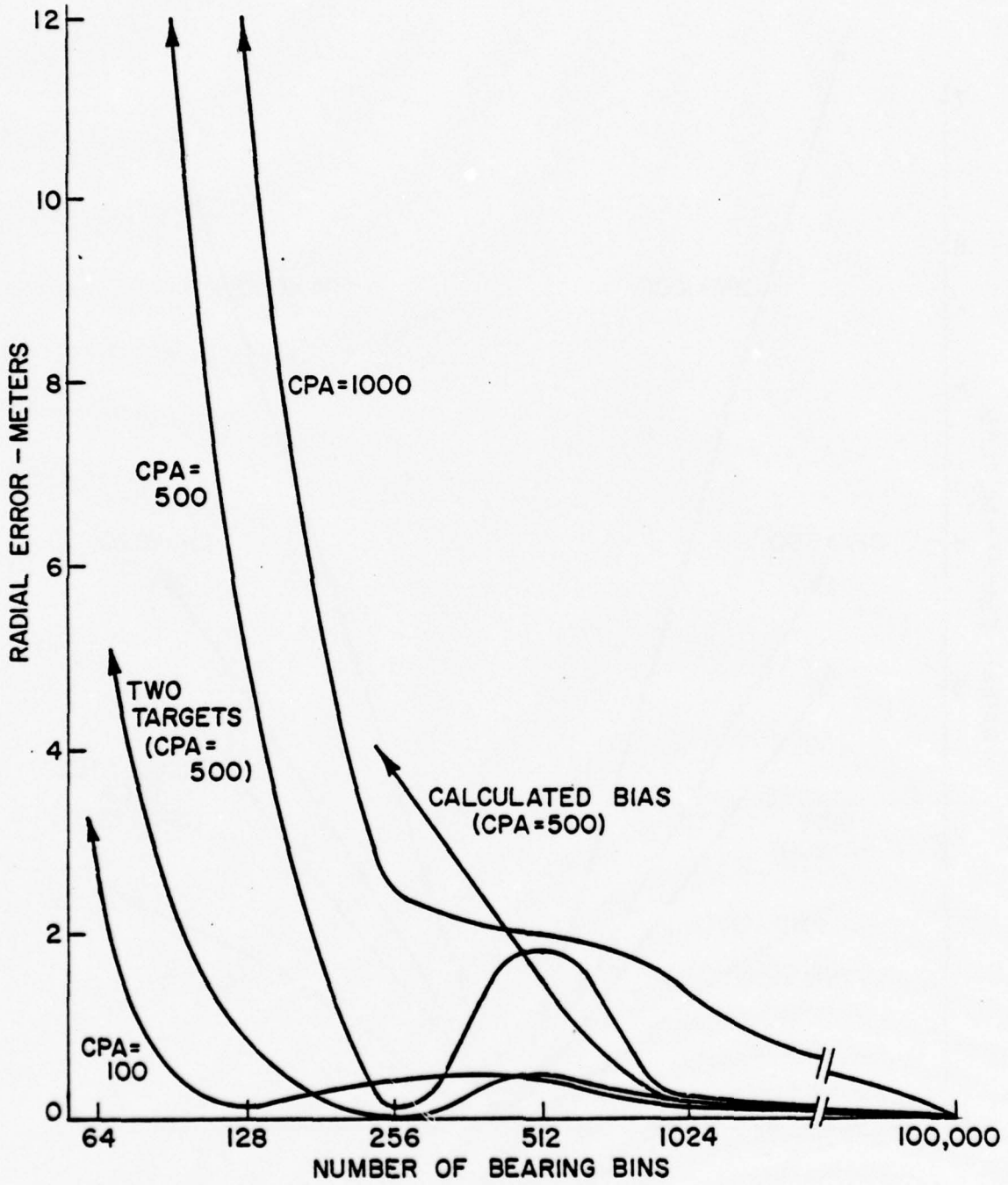





FIGURE 22

# APPENDIX B

## FORTRAN PROGRAMS AND FLOW CHARTS

Since there are many calculations involved in locating a sensor position, it is best done using a computer. Appendix B is a listing of the flow charts and the computer program used to accomplish this. The computer program is called Simulate, and it calls nine subroutines called Input, Senchk, Output, SPL, Enter, Shift, TXY, Normal and Uniform.

Program Simulate is the program which locates the sensor position. It accomplishes this by calling four subroutines; INPUT, SENCHK, SPL, and OUTPUT.

Subroutine INPUT loads the initial parameters into the program.

Subroutine SENCHK picks which sensor reports should be used in the calculation. For those reports it provides the time of report, report bearing, and a weighting function to the main program.

Subroutine SPL calculates the sensor location and the rotational bias from the data provided to it from the main program.

Subroutine OUTPUT prints out the results of the calculations.

There are five other subroutines which are used in this program. Subroutines Normal and Uniform together provide a random number uniformly distributed between 0 and 1 to Subroutine SENCHK. Subroutines Enter, Shift, and TXY simply perform minor arithmetic calculations needed in the SPL Subroutine.

The flow charts, Figures 23-30, and computer printout for these programs, Figure 31, follow.

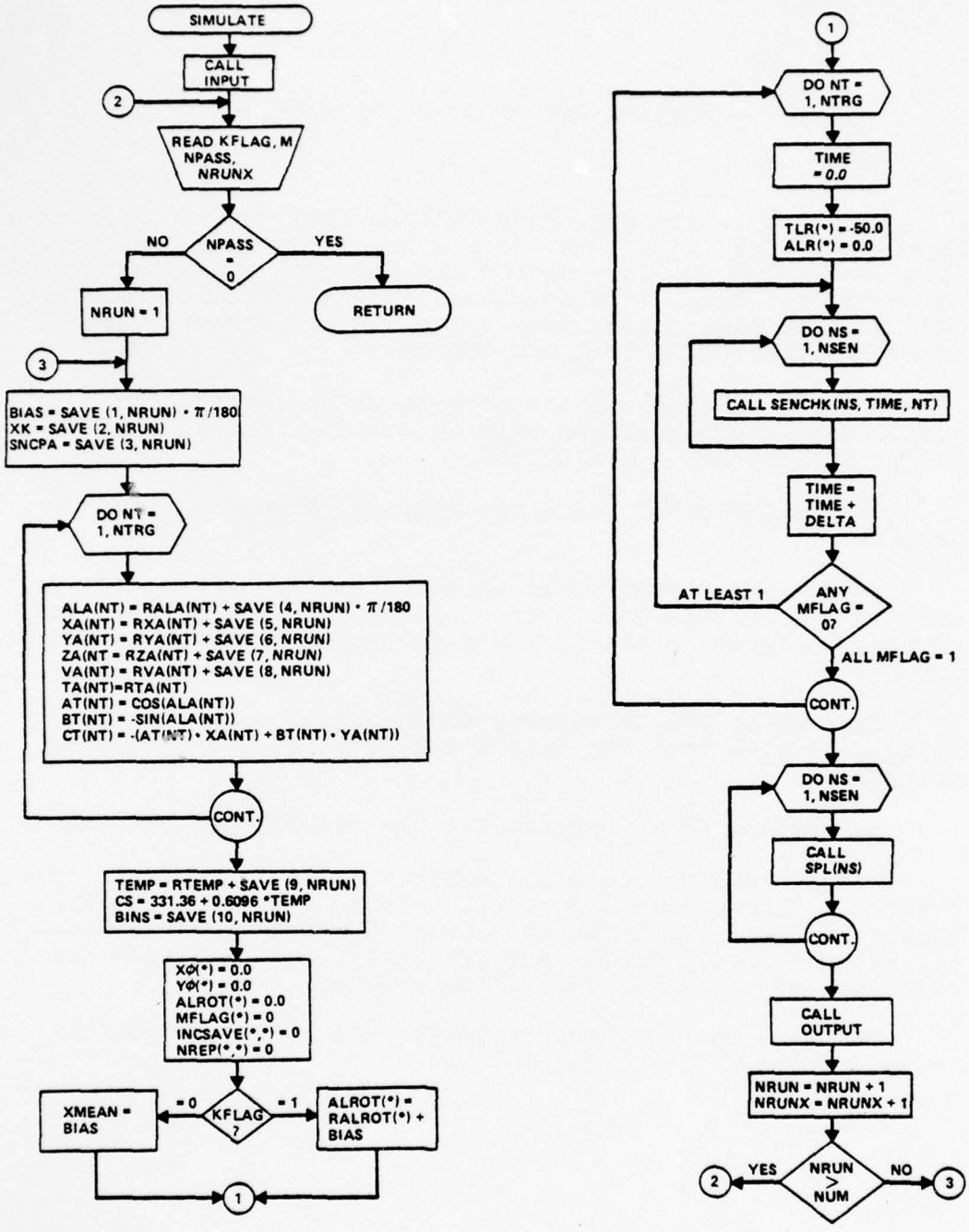



FIG. 23 FLOWCHART OF SIMULATE

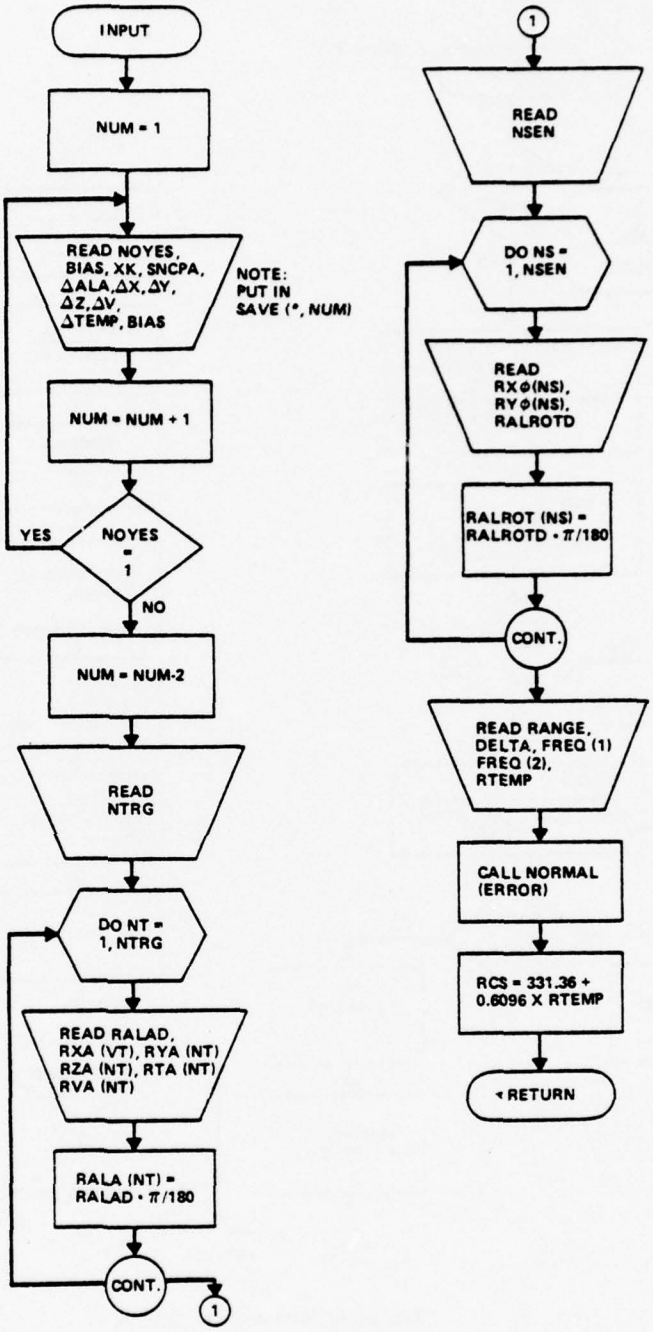



FIG. 24 FLOWCHART OF INPUT

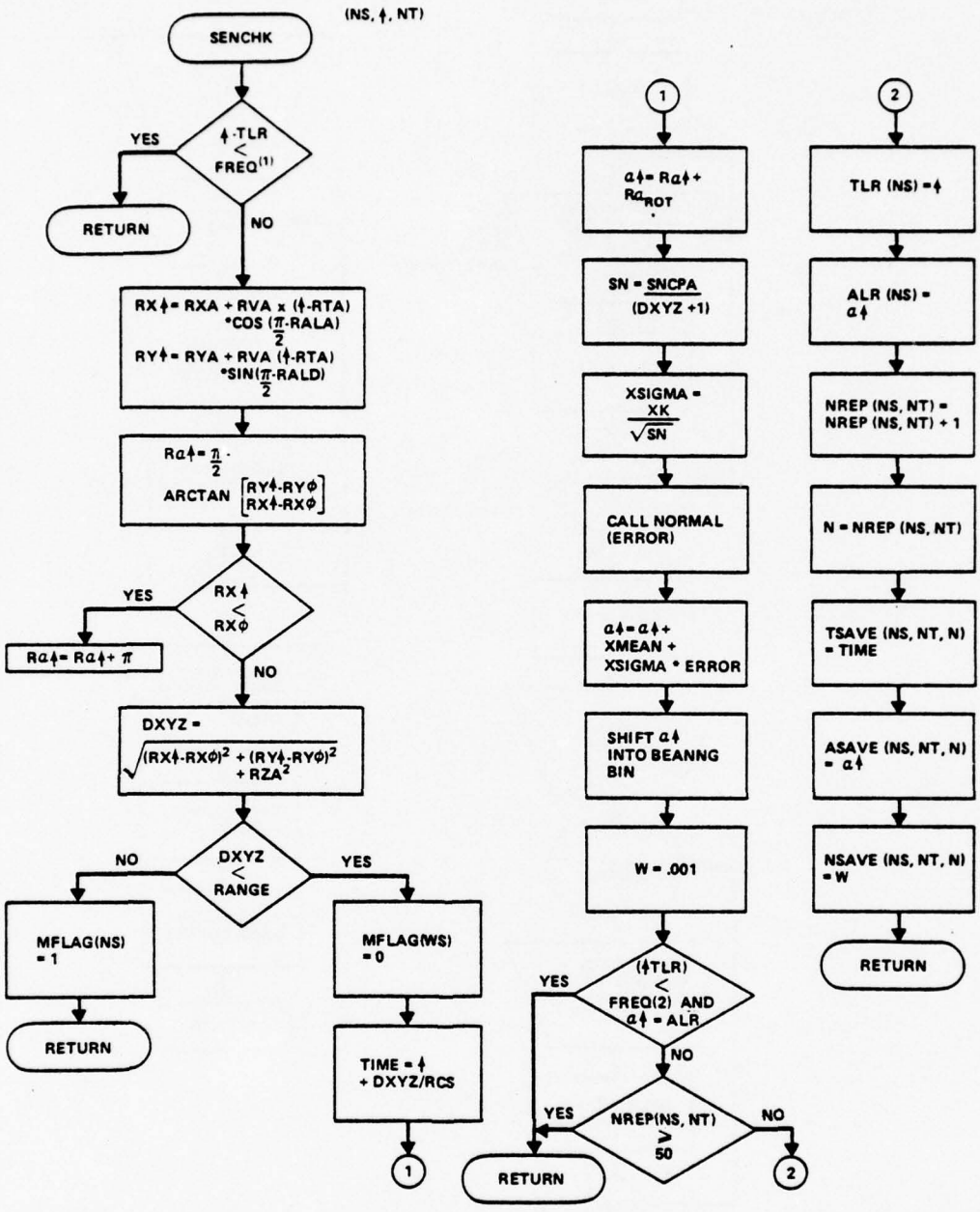



FIG. 25 FLOWCHART OF SENCHK

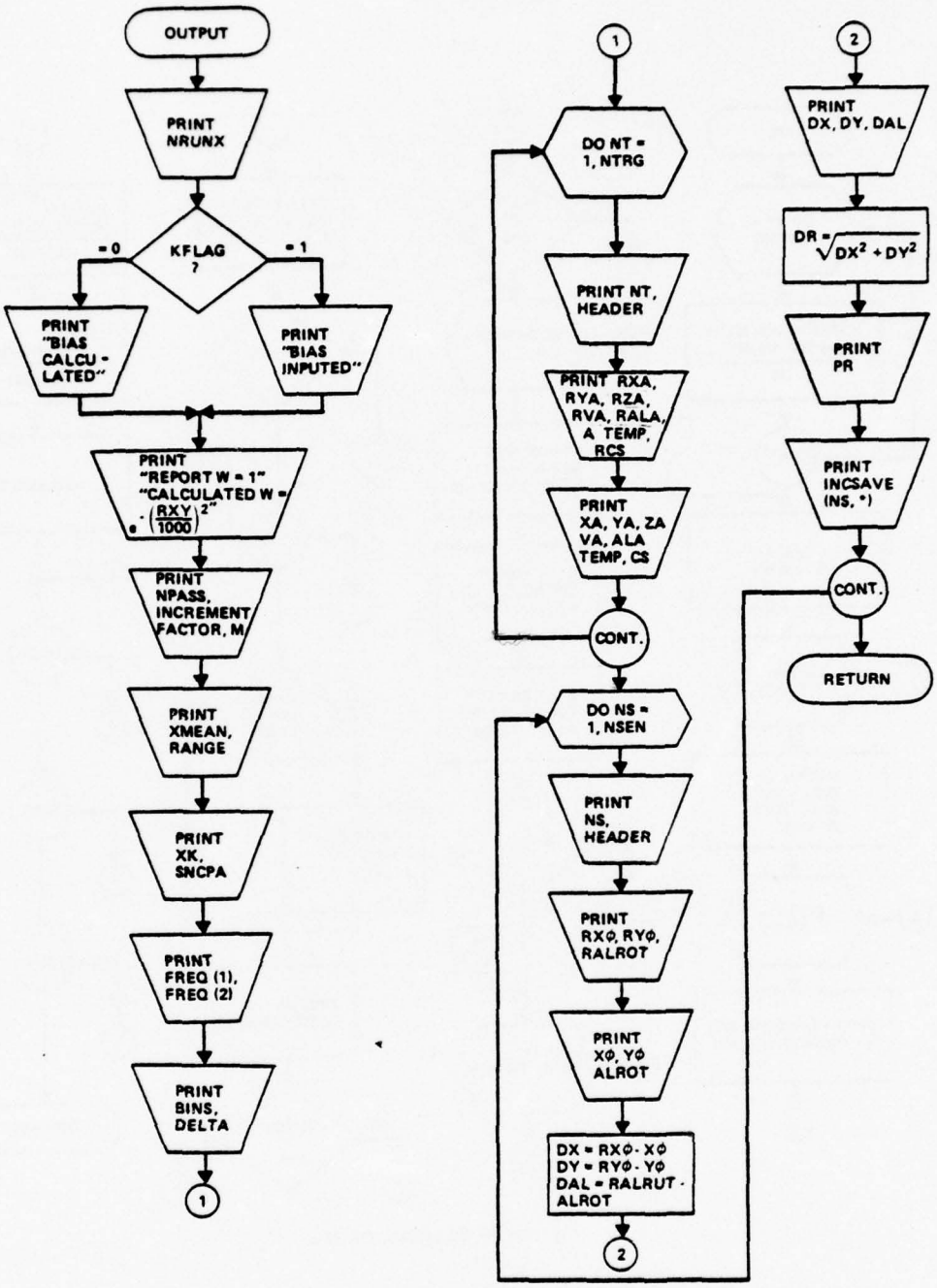



FIG. 26 FLOWCHART OF OUTPUT

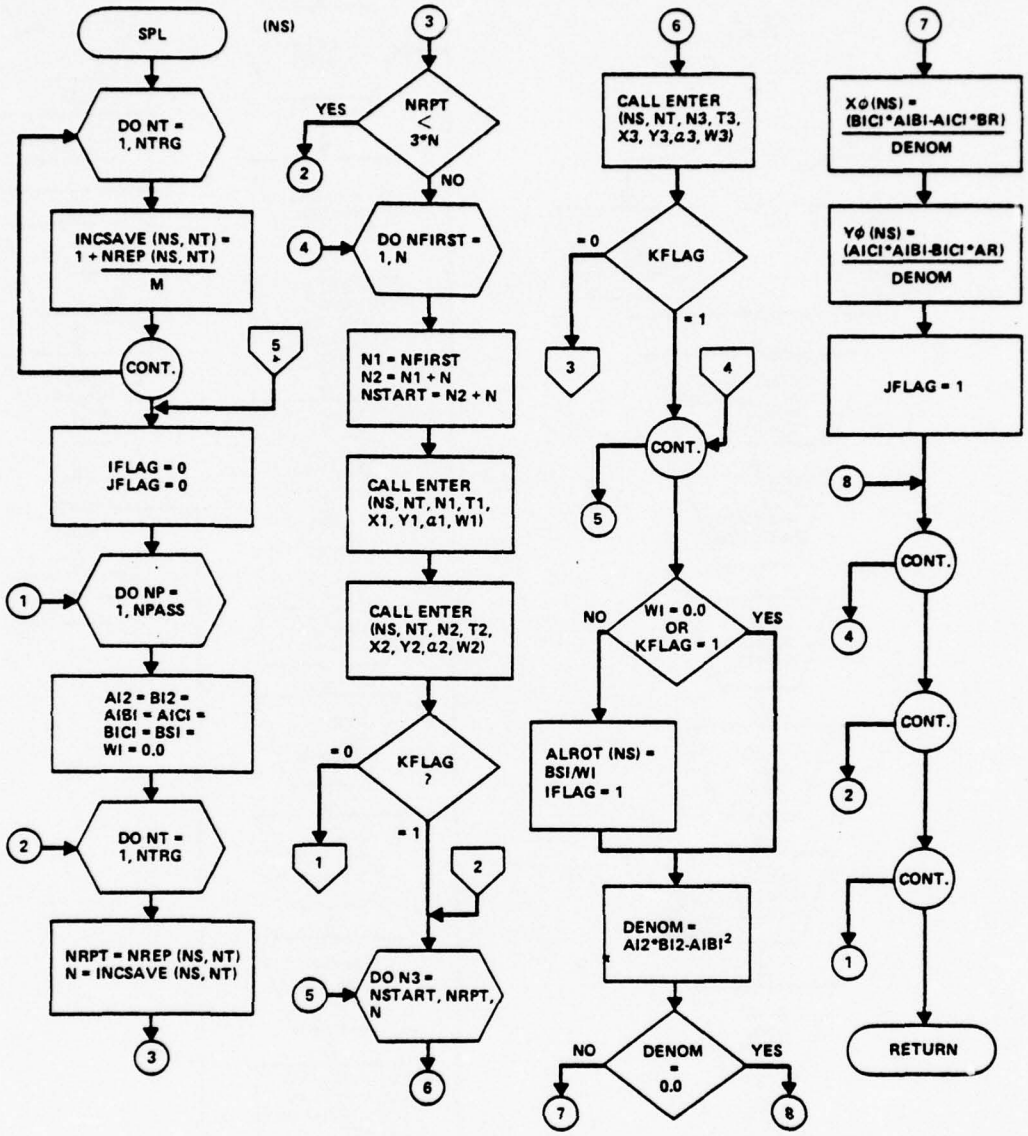



FIG. 27a FLOWCHART OF SPL

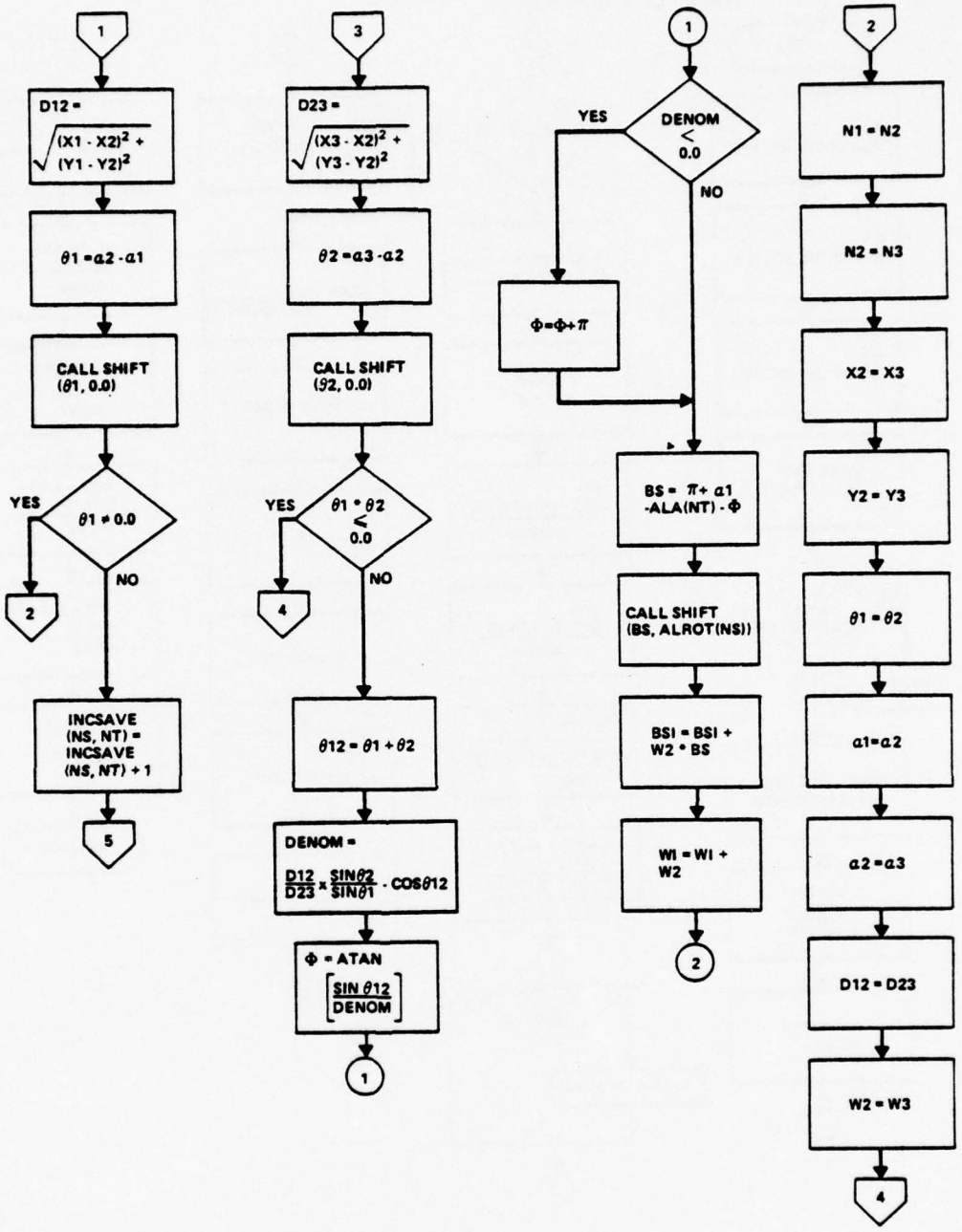



FIG. 27b FLOWCHART OF SPL (CONT.)

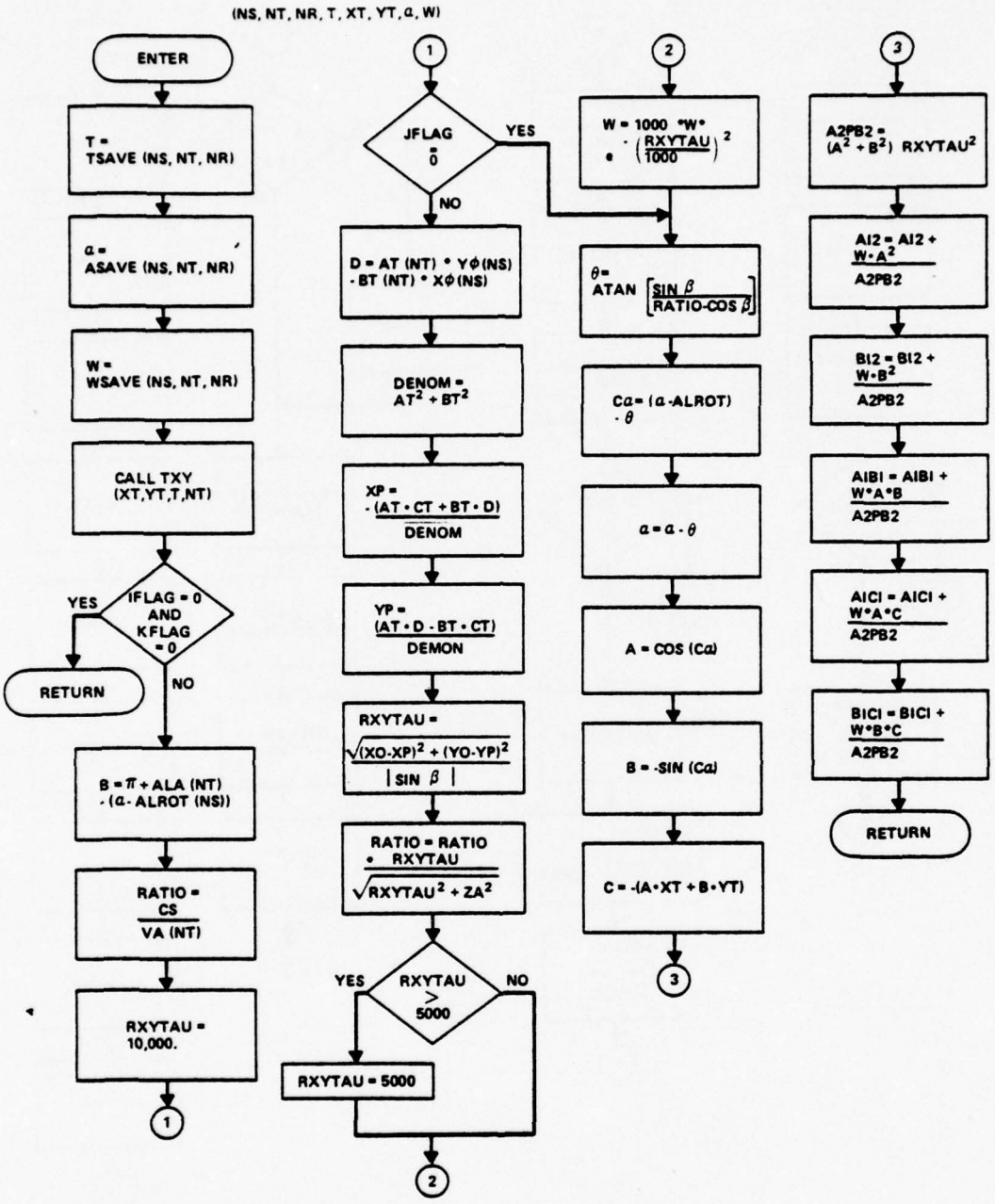



FIG. 28 FLOWCHART OF ENTER

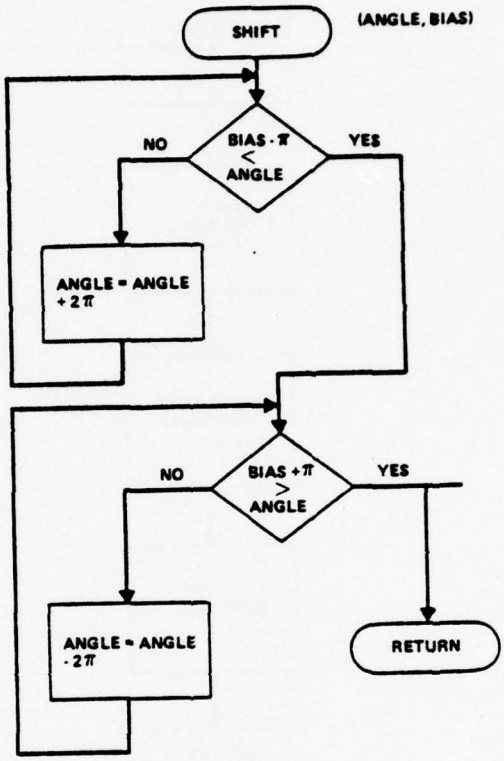



FIG. 29 FLOWCHART OF SHIFT

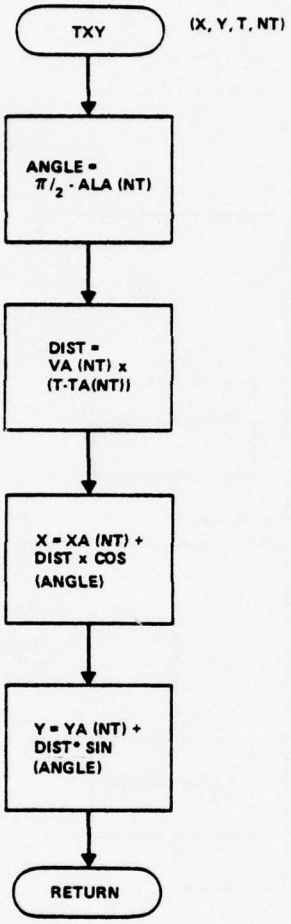



FIG. 30 FLOWCHART OF TXY

```
:DL FORTRAN (1.0)

00000            PROGRAM SIMULATE
                 C
                 C
                 C  ***************************************************************
                 C  *                                                             *
                 C  *    JONATHAN VALVANO                              03/18/75   *
                 C  *                                                             *
                 C  *    THESE PROGRAMS SIMULATE A SENSOR FIELD FOR THE           *
                 C  *    DEVELOPMENT OF A SENSOR POSITION LOCATOR (SPL).          *
                 C  *    ACOUSTIC SENSORS ARE DEPLOYED IN THE FIELD WITHOUT       *
                 C  *    THE KNOWLEDGE OF THEIR EXACT LOCATION.  THIS CAN BE DONE *
                 C  *    BY AIR DROP OR GUN DELIVERY.  THE PURPOSE OF THESE       *
                 C  *    ROUTINES IS TO LOCATE THE SENSORS ONCE THEY ARE IN THE   *
                 C  *    FIELD.  THE FOLLOWING METHOD IS USED....                 *
                 C  *         1. A TARGET, WHOSE POSITION AS A FUNCTION OF TIME   *
                 C  *            IS KNOWN, IS RUN OVER THE SENSOR FIELD.          *
                 C  *         2. THE RECIEVING UNIT LISTENS AND RECORDS THE SEQUENCE*
                 C  *            OF REPORTS BY THE SENSOR OF THE TARGET.          *
                 C  *         3. THE ARRAYS OF REPORTS PLUS THE TARGET LOCATIONS   *
                 C  *            ARE FED AS INPUTS TO THE SPL ALGORITHM.          *
                 C  *         4. SPL CALCULATES THE SENSOR LOCATION (AND BIAS ANGLE)*
                 C  *                                                             *
                 C  *    THE FOLLOWING OPTIONS ARE AVAILABLE TO SPL...            *
                 C  *         KFLAG=0 IF THE BIAS IS TO BE CALCULATED.            *
                 C  *         KFLAG=1 IF THE BIAS IS REPORTED ( INPUTED TO SPL.)  *
                 C  *         M IS THE INCREMENT FACTOR (SEE SPL)                 *
                 C  *         NPASS IS THE NUMBER OF PASSES SPL MAKES OVER REPORTS*
                 C  *                                                             *
                 C  *    THE FOLLOWING OPTIONS ARE AVAILABLE TO THE SIMULATION... *
                 C  *         NSEN       THE NUMBER OF SENSORS                    *
                 C  *         NTRG       THE NUMBER OF TARGETS                    *
                 C  *         RANGE      THE RANGE OF THE SENSORS                 *
                 C  *         BINS       THE NUMBER OF BEARING BINS OF THE SENSOR *
                 C  *         FREQ(1)    THE CYCLE TIME WHEN THE BEARING CHANGES  *
                 C  *         FREQ(2)    THE CYCLE TIME WHEN THE BEARING IS THE SAME*
                 C  *         DELTA      THE TIME INCREMENT IN THE SIMULATION     *
                 C  *         XMEAN,XK,SNCPA  ARE FOR GAUSSIAN ERROR              *
                 C  *                    MEAN=XMEAN                               *
                 C  *                    SIGMA=XK/SQRT(SNCPA/(DXYZ+1))            *
                 C  *                                                             *
                 C  ***************************************************************
                 C
00001                 COMMON /TRGT// ALA(3),XA(3),YA(3),ZA(3),
00001                1  TA(3),VA(3),AT(3),BT(3),CT(3),TEMP,CS,NTRG
00002                 COMMON /RTRGT// RALA(3),RXA(3),RYA(3),RZA(3),
00002                1  RTA(3),RVA(3),RTEMP,RCS
00003                 COMMON /SEN// NSEN,X0(3),Y0(3),ALR0(3),IFLAG,JFLAG
00004                 COMMON /RSEN// RX0(3),RY0(3),RALR0T(3)
00005                 COMMON /REP// TSAVE(3,3,50),ASAVE(3,3,50),WSAVE(3,3,50),
00005                1  NREP(3,3),INCSAVE(3,3)
00006                 COMMON /STATE// TLR(3),ALR(3),VFLAG(3),FREQ(2),
00006                1  BINS,XMEAN,XK,SNCPA,RANGE,DELTA,NUM,NRUN,NRUNX,SAVE(10,100)
00007                 COMMON /CONST// PI,PI02,PI2
00008                 COMMON /OPTIONS// KFLAG,M,NPASS
```

FIG. 31 COMPUTER PROGRAM

```
C
C    ***************************************************************
C    *                                                             *
C    *    FIRST CALL INPUT TO SET UP THE SIMULATION                *
C    *                                                             *
C    ***************************************************************
C
00009          CALL INPUT
C
C    ***************************************************************
C    *                                                             *
C    *    THEN READ THE OPTIONS FOR THIS SET OF RUNS               *
C    *    THIS IS THE START OF A SET OF RUNS WITH SEPERATE OPTIONS. *
C    *                                                             *
C    ***************************************************************
C
00010     101  READ 103, KFLAG,M,NPASS,NRUNX
00011     103  FORMAT (4I10)
00012          IF (NPASS .EQ. 0) RETURN
00014          NRUN=1
C
C
C    ***************************************************************
C    *                                                             *
C    *    SAVE CONTAINS THE SIMULATION OPTIONS FOR EACH RUN OF SET  *
C    *                                                             *
C    *    ALL ANGLE INPUTS AND OUTPUTS ARE IN DEGREES AND          *
C    *    ALL INTERNAL ANGLES ARE IN RADIANS.                      *
C    *                                                             *
C    *    ALA  IS THE INPUTED TARGET COURSE.                       *
C    *    RALA IS THE REAL    TARGET COURSE.                       *
C    *    VA  IS THE INPUTED TARGET VELOCITY.                      *
C    *    RVA IS THE REAL     TARGET VELOCITY.                     *
C    *    AT TIME TA (SECONDS THE TARGET IS AT (XA,YA,ZA) INPUTED.  *
C    *    AT TIME RTA (SEC) THE TARGET IS AT (RXA,RYA,RZA)  REAL.   *
C    *                                                             *
C    ***************************************************************
C
C
00015     110  BIAS=SAVE(1,NRUN)*PI/180.0
00016          XK=SAVE(2,NRUN)
00017          SNCHA=SAVE(3,NRUN)
00018          DO 155 NT=1,NTRG
00019          ALA(NT)=RALA(NT)+SAVE(4,NRUN)*PI/180.0
00020          XA(NT)=RXA(NT)+SAVE(5,NRUN)
00021          YA(NT)=RYA(NT)+SAVE(6,NRUN)
00022          ZA(NT)=RZA(NT)+SAVE(7,NRUN)
00023          VA(NT)=RVA(NT)+SAVE(8,NRUN)
00024          TA(NT)=RTA(NT)
C
C
C
C
C
```

```
C
C     ****************************************************************
C     *                                                              *
C     *    AT,BT,CT FORM THE LINE AT*X+BT*Y+CT=0 OF THE TARGET TRACK.*
C     *                                                              *
C     *                                                              *
C     ****************************************************************
C
00025           AT(NT)=COS(ALA(NT))
00026           BT(NT)=-SIN(ALA(NT))
00027           CT(NT)=-(AT(NT)*XA(NT)+BT(NT)*YA(NT))
00028    155    CONTINUE
C
C
C
C
C
C     ****************************************************************
C     *                                                              *
C     *    CS  IS THE INPUTED SPEED OF SOUND IN METERS/SECOND.       *
C     *    RCS IS THE REAL    SPEED OF SOUND IN METERS/SECOND.       *
C     *    TEMP  IS THE INPUTED TEMPERATURE IN CENTIGRADE.          *
C     *    RTEMP IS THE REAL    TEMPERATURE IN CENTIGRADE.          *
C     *                                                              *
C     ****************************************************************
C
00029           TEMP=RTEMP+SAVE(9,NRUN)
00030           CS=331.36+0.6096*TEMP
00031           BINS=SAVE(10,NRUN)
00032           DO 160 I=1,3
C
C     ****************************************************************
C     *                                                              *
C     *    (X0,Y0) IS THE CALCULATED SENSOR POSITION.                *
C     *    (RX0,RY0) IS THE REAL SENSOR POSITION.                    *
C     *    ALROT IS EITHER THE CALCULATED OR INPUTED SENSOR          *
C     *         ANGLE ROTATION BIAS, DEPENDING ON OPTION- KFLAG.     *
C     *    RALROT IS THE REAL SENSOR ANGLE ROTATION BIAS.            *
C     *                                                              *
C     *    MFLAG=0 WHEN THE TARGET IS WITHIN RANGE OF THAT SENSOR.   *
C     *    MFLAG=1 WHEN THE TARGET IS OUTSIDE THE RANGE OF THAT SENSOR*
C     *                                                              *
C     ****************************************************************
C
00033           X0(I)=0.0
00034           Y0(I)=0.0
00035           ALROT(I)=0.0
00036           MFLAG(I)=0
00037           IF (KFLAG .EQ. 0) XMEAN=BIAS
00039           IF (KFLAG .EQ. 1) ALROT(I)=RALROT(I)+BIAS
00041           DO 160 J=1,3
00042           INCSAVE(I,J)=0
00043    160    NREP(I,J)=0
```

```
00044                DO 125 NT=1,NTRG
00045                TIME=0.0
00046                DO 111 I=1,3
                C
                C   ***************************************************************
                C   *                                                             *
                C   *    TLR IS THE TIME OF THE LAST REPORT.                      *
                C   *    ALR IS THE ANGLE OF THE LAST REPORT.                     *
                C   *                                                             *
                C   ***************************************************************
                C
00047                TLR(I)=-50.0
00048                ALR(I)=0.0
00049         111    CONTINUE
00050         115    DO 120 NS=1,NSEN
                C
                C   ***************************************************************
                C   *                                                             *
                C   *    SENCHK WILL CHECK TO SEE IF THE SENSOR SHOULD REPORT.    *
                C   *    IF SO IT DETERMINES THE TIME OF THE REPORT AND PUTS IT   *
                C   *    IN TSAVE. IT DETERMINES THE BEARING AND PUTS IT IN ASAVE. *
                C   *    IF THE SENSOR REPORTS A WEIGHTING FACTOR IT IS  IN WSAVE. *
                C   *                                                             *
                C   ***************************************************************
                C
00051         120    CALL SENCHK(NS,TIME,NT)
00052                TIME=TIME+DELTA
00053                DO 121 NS=1,NSEN
00054                IF (MFLAG(NS) .EQ. 0) GO TO 115
00056         121    CONTINUE
00057         125    CONTINUE
                C
                C   ***************************************************************
                C   *                                                             *
                C   *    NOW THE ARRAYS (TSAVE,ASAVE,WSAVE) ARE FULL.             *
                C   *                                                             *
                C   ***************************************************************
                C
00058                DO 130 NS=1,NSEN
                C
                C   ***************************************************************
                C   *                                                             *
                C   *    CALL THE SENSOR POSITION LOCATOR (SPL).                  *
                C   *                                                             *
                C   ***************************************************************
                C
00059                CALL SPL(NS)
00060         130    CONTINUE
                C
                C
                C
                C
                C
                C
```

```
C
C     ***************************************************************
C     *                                                             *
C     *    CALL OUTPUT TO PRINT RESULTS.                            *
C     *                                                             *
C     ***************************************************************
C
00061           CALL OUTPUT
00062           NRUN=NRUN+1
00063           NRUNX=NRUNX+1
00064           IF (NRUN .GT. NUM) GO TO 101
00066           GO TO 110
00067           END
```

NOL FORTRAN DIAGNOSTIC RESULTS - FOR SIMULATE

NO ERRORS

DATA VARIABLES

| | | | | | |
|---|---|---|---|---|---|
| ALA | 00001 | 00019 | 00025 | 00026 | |
| ALR | 00006 | 00048 | | | |
| ALROT | 00003 | 00035 | 00040 | | |
| ASAVE | 00005 | | | | |
| AT | 00001 | 00025 | 00027 | | |
| BINS | 00006 | 00031 | | | |
| BT | 00001 | 00026 | 00027 | | |
| CS | 00001 | 00030 | | | |
| CT | 00001 | 00027 | | | |
| DELTA | 00006 | 00052 | | | |
| FREQ | 00006 | | | | |
| IFLAG | 00003 | | | | |
| INCSAVE | 00005 | 00042 | | | |
| JFLAG | 00003 | | | | |
| KFLAG | 00008 | 00010 | 00037 | 00039 | |
| M | 00008 | 00010 | | | |
| MFLAG | 00006 | 00036 | 00054 | | |
| NPASS | 00008 | 00010 | 00012 | | |
| NREP | 00005 | 00043 | | | |
| NRUN | 00006 | 00014 | 00015 | 00016 | 00017 |
| | 00023 | 00029 | 00031 | 00062 | 00062 |
| NRUNX | 00006 | 00010 | 00063 | 00063 | |
| NSEN | 00003 | 00050 | 00053 | 00058 | |
| NTRG | 00001 | 00018 | 00044 | | |
| NUM | 00006 | 00064 | | | |
| PI | 00007 | 00015 | 00019 | | |
| PI2 | 00007 | | | | |
| PID2 | 00007 | | | | |
| RALA | 00002 | 00019 | | | |
| RALROT | 00004 | 00040 | | | |
| RANGE | 00006 | | | | |
| RCS | 00002 | | | | |
| RTA | 00002 | 00024 | | | |
| RTEMP | 00002 | 00029 | | | |
| RVA | 00002 | 00023 | | | |
| RX0 | 00004 | | | | |
| RXA | 00002 | 00020 | | | |
| RY0 | 00004 | | | | |
| RYA | 00002 | 00021 | | | |
| RZA | 00002 | 00022 | | | |
| SAVE | 00006 | 00015 | 00016 | 00017 | 00019 |
| | 00029 | 00031 | | | |
| SNCPA | 00006 | 00017 | | | |
| TA | 00001 | 00024 | | | |
| TEMP | 00001 | 00029 | 00030 | | |
| TLR | 00006 | 00047 | | | |
| TSAVE | 00005 | | | | |
| VA | 00001 | 00023 | | | |
| WSAVE | 00005 | | | | |
| X0 | 00003 | 00033 | | | |
| XA | 00001 | 00020 | 00027 | | |
| XK | 00006 | 00016 | | | |

| | | | | | |
|---|---|---|---|---|---|
| XMEAN | 00006 | 00038 | | | |
| Y0 | 00003 | 00034 | | | |
| YA | 00001 | 00021 | 00027 | | |
| ZA | 00001 | 00022 | | | |

COMMON VARIABLES

**NONE**

PROGRAM VARIABLES

| | | | | | |
|---|---|---|---|---|---|
| BIAS | 00015 | 00038 | 00040 | | |
| COS | 00025 | | | | |
| I | 00032 | 00033 | 00034 | 00035 | 00036 |
| | 00046 | 00047 | 00048 | | |
| INPUT | 00009 | | | | |
| J | 00041 | 00042 | 00043 | | |
| NS | 00050 | 00051 | 00053 | 00054 | 00058 |
| NT | 00018 | 00019 | 00019 | 00020 | 00020 |
| | 00023 | 00023 | 00024 | 00024 | 00025 |
| | 00027 | 00027 | 00027 | 00027 | 00044 |
| OUTPUT | 00061 | | | | |
| SENCHK | 00051 | | | | |
| SIN | 00026 | | | | |
| SPL | 00059 | | | | |
| TIME | 00048 | 00051 | 00052 | 00052 | |

```
INPUT    DECK/     L.R.X
```

```
NOL FORTRAN (1.0)

00000                SUBROUTINE INPUT
                C
                C   ***************************************************************
                C   *                                                             *
                C   *    JONATHAN VALVANO                                 03/16/75 *
                C   *                                                             *
                C   *    THIS ROUTINE INPUTS INITIAL PARAMETERS.                  *
                C   *    ALL ANGLE INPUTS ARE IN DEGREES.                         *
                C   *                                                             *
                C   ***************************************************************
                C
00001                COMMON /TRGT// ALA(3),XA(3),YA(3),ZA(3),
00001               1 TA(3),VA(3),AT(3),BT(3),CT(3),TEMP,CS,NTRG
00002                COMMON /RTRGT// RALA(3),RXA(3),RYA(3),RZA(3),
00002               1 RTA(3),RVA(3),RTEMP,RCS
00003                COMMON /SEN// NSEN,X0(3),Y0(3),ALROT(3),IFLAG,JFLAG
00004                COMMON /RSEN// RX0(3),RY0(3),RALROT(3)
00005                COMMON /REP// TSAVE(3,3,50),ASAVE(3,3,50),WSAVE(3,3,50),
00005               1 NREP(3,3),INCSAVE(3,3)
00006                COMMON /STATE// TLR(3),ALR(3),MFLAG(3),FREQ(2),
00006               1 BINS,XMEAN,XK,SNCPA,RANGE,DELTA,NUM,NRUN,NRUNX,SAVE(10,100)
00007                COMMON /CONST// PI,PID2,PI2
00008                COMMON /OPTIONS// KFLAG,M,NPASS
00009                DATA /CONST/ (PI=3.1415926538)
00010                DATA /CONST/ (PID2=1.570796327)
00011                DATA /CONST/ (PI2=6.283185307)
00012                DATA /REP/ (NREP(I)=9(0))
00013                DATA /STATE/ (TLR(I)=3(-50.0))
00014                DATA /STATE/ (ALR(I)=3(0.0))
                C
                C
                C
                C
                C   ***************************************************************
                C   *                                                             *
                C   *    READ SIMULATION OPTIONS.                                 *
                C   *    NUM IS THE NUMBER OF RUNS IN A SET.                      *
                C   *                                                             *
                C   ***************************************************************
                C
00015                NUM=1
00016           279  READ 280, NOYES,(SAVE(I,NUM), I=1,10)
00017           280  FORMAT (I5,F5.0,F10.4,7F5.0,F10.0)
00018                NUM=NUM+1
00019                IF (NOYES .EQ. 1) GO TO 279
00021                NUM=NUM-2
                C
                C
                C
                C
                C
                C
```

```
NOL FORTRAN (1.0)                    SUBROUTINE INPUT

00022                READ 200, NTRG
00023       200      FORMAT (I5)
00024                DO 225 NT=1,NTRG
                C
                C    ***************************************************************
                C    *                                                             *
                C    *     READ REAL TARGET CHARACTERISTICS.                       *
                C    *                                                             *
                C    ***************************************************************
                C
00025                READ 220, RALAD,RXA(NT),RYA(NT),RZA(NT),RTA(NT),RVA(NT)
00026       220      FORMAT (6F10.2)
00027                RALA(NT)=RALAD*PI/180.0
00028       225      CONTINUE
00029                READ 230, NSEN
00030       230      FORMAT (I5)
00031                DO 250  NS=1,NSEN
                C
                C    ***************************************************************
                C    *                                                             *
                C    *     READ REAL SENSOR CHARACTERISTICS.                       *
                C    *                                                             *
                C    ***************************************************************
                C
00032                READ 240, RX0(NS),RY0(NS),RALROTD
00033       240      FORMAT (3F10.2)
00034                RALROT(NS)=RALROTD*PI/180.0
00035       250      CONTINUE
00036                READ 290, RANGE,DELTA,FREQ(1),FREQ(2),RTEMP
00037       290      FORMAT (5F10.2)
                C
                C    ***************************************************************
                C    *                                                             *
                C    *     NORMAL IS THE SUBROUTINE THAT GENERATES GAUSSIAN NOISE. *
                C    *     HERE, IT IS CALLED JUST TO INITIALIZE.                  *
                C    *                                                             *
                C    ***************************************************************
                C
00038                CALL NORMAL (ERROR)
                C
                C    RCS IS THE REAL SPEED OF SOUND.
                C
00039                RCS=331.36+0.6096*RTEMP
00040                RETURN
00041                END

            NOL FORTRAN DIAGNOSTIC RESULTS - FOR   INPUT


NO ERRORS
```

DATA VARIABLES

| | | | | | |
|---|---|---|---|---|---|
| ALA | 00001 | | | | |
| ALR | 00006 | 00014 | | | |
| ALROT | 00003 | | | | |
| ASAVE | 00005 | | | | |
| AT | 00001 | | | | |
| BINS | 00006 | | | | |
| BT | 00001 | | | | |
| CS | 00001 | | | | |
| CT | 00001 | | | | |
| DELTA | 00006 | 00036 | | | |
| FREQ | 00006 | 00036 | 00036 | | |
| IFLAG | 00003 | | | | |
| INCSAVE | 00005 | | | | |
| JFLAG | 00003 | | | | |
| KFLAG | 00008 | | | | |
| M | 00008 | | | | |
| MFLAG | 00006 | | | | |
| NPASS | 00008 | | | | |
| NREP | 00005 | 00012 | | | |
| NRUN | 00006 | | | | |
| NRUNX | 00006 | | | | |
| NSEN | 00003 | 00029 | 00031 | | |
| NTRG | 00001 | 00022 | 00024 | | |
| NUM | 00006 | 00015 | 00016 | 00018 | 00018 |
| PI | 00007 | 00009 | 00027 | 00034 | |
| PI2 | 00007 | 00011 | | | |
| PID2 | 00007 | 00010 | | | |
| RALA | 00002 | 00027 | | | |
| RALROT | 00004 | 00034 | | | |
| RANGE | 00006 | 00036 | | | |
| RCS | 00002 | 00039 | | | |
| RTA | 00002 | 00025 | | | |
| RTEMP | 00002 | 00036 | 00039 | | |
| RVA | 00002 | 00025 | | | |
| RX0 | 00004 | 00032 | | | |
| RXA | 00002 | 00025 | | | |
| RY0 | 00004 | 00032 | | | |
| RYA | 00002 | 00025 | | | |
| RZA | 00002 | 00025 | | | |
| SAVE | 00006 | 00016 | | | |
| SNCPA | 00006 | | | | |
| TA | 00001 | | | | |
| TEMP | 00001 | | | | |
| TLR | 00006 | 00013 | | | |
| TSAVE | 00005 | | | | |
| VA | 00001 | | | | |
| WSAVE | 00005 | | | | |
| X0 | 00003 | | | | |
| XA | 00001 | | | | |
| XR | 00006 | | | | |
| XMEAN | 00006 | | | | |
| Y0 | 00003 | | | | |

SUBROUTINE INPUT

| | |
|---|---|
| YA | 00001 |
| ZA | 00001 |

COMMON VARIABLES

**NONE**

PROGRAM VARIABLES

| | | | | | |
|---|---|---|---|---|---|
| ERROR | 00038 | | | | |
| I | 00016 | 00016 | | | |
| NORMAL | 00038 | | | | |
| NOYES | 00016 | 00019 | | | |
| NS | 00031 | 00032 | 00032 | 00034 | |
| NT | 00024 | 00025 | 00025 | 00025 | 00025 |
| RALAD | 00025 | 00027 | | | |
| RALROTD | 00032 | 00034 | | | |

```
SENCHK   DECK/      L.R.X
```

```
NOL FORTRAN (1.0)

00000                SUBROUTINE SENCHK (NS,TAU,NT)
                C
                C   ***************************************************************
                C   *                                                             *
                C   *    JONATHAN VALVANO                                 03/18/75 *
                C   *                                                             *
                C   *    THIS ROUTINE CHECKS TO SEE IF THE SENSOR SHOULD           *
                C   *    REPORT AT THIS TIME. IF IT DECIDES TO REPORT THEN IT MAKES *
                C   *    THE APPROPRIATE ENTRIES INTO THE ARRAYS. (TSAVE,ASAVE,WSAVE*
                C   *                                                             *
                C   ***************************************************************
                C
00001                COMMON /RTRGT// RALA(3),RXA(3),RYA(3),RZA(3),
00001               1 RTA(3),RVA(3),RTEMP,PCS
00002                COMMON /RSEN// RX0(3),RY0(3),RALROT(3)
00003                COMMON /REP// TSAVE(3,3,50),ASAVE(3,3,50),WSAVE(3,3,50),
00003               1 NREP(3,3),INCSAVE(3,3)
00004                COMMON /STATE// TLR(3),ALR(3),MFLAG(3),FREQ(2),
00004               1 BINS,XMEAN,XK,SNCPA,RANGE,DELTA,NUM,NRUN,NRUNX,SAVE(10,100)
00005                COMMON /CONST// PI,PID2,PI2
                C
                C   ***************************************************************
                C   *                                                             *
                C   *    THE SENSOR CAN ONLY REPORT EVERY FREQ(1) SECONDS IF THE  *
                C   *    TARGET IS CHANGING BINS. IF IT HAS NOT BEEN AT LEAST     *
                C   *    FREQ(1) SECONDS, CONTROL IS RETURNED TO SIMULATE,        *
                C   *    AND THE SENSOR DOES NOT REPORT.                          *
                C   *                                                             *
                C   ***************************************************************
                C
00006                IF ((TAU-TLR(NS)) .LT. FREQ(1)) RETURN
                C
                C   ***************************************************************
                C   *                                                             *
                C   *    TIME TAU IS WHEN THE SOUND ORIGINATES FROM THE TARGET.   *
                C   *    RXTAU=REAL X POSITION OF THE TARGET AT TIME TAU.         *
                C   *    RYTAU=REAL Y POSITION OF THE TARGET AT TIME TAU.         *
                C   *    RALTAU=REAL ALPHA ANGLE TO THE TARGET AT TIME TAU.       *
                C   *                                                             *
                C   ***************************************************************
                C
00008                RXTAU=RXA(NT)+RVA(NT)*(TAU-RTA(NT))*COS(PID2-RALA(NT))
00009                RYTAU=RYA(NT)+RVA(NT)*(TAU-RTA(NT))*SIN(PID2-RALA(NT))
00010                RALTAU=PID2  -ATAN((RYTAU-RY0(NS))/(RXTAU-RX0(NS)))
00011                IF (RXTAU .LT. RX0(NS)) RALTAU=RALTAU+PI
                C
                C   ***************************************************************
                C   *                                                             *
                C   *    DXYZ IS THE DISTANCE FROM THE SENSOR TO THE TARGET. IT IS *
                C   *    THIS DISTANCE THAT THE SOUND TRAVELS (FROM TARGET TO SENSOR*
                C   *                                                             *
                C   ***************************************************************
                C
00013                DXYZ=SQRT((RXTAU-RX0(NS))**2+(RYTAU-RY0(NS))**2+RZA(NT)**2)
```

```
NOL FORTRAN (1.0)                SUBROUTINE SENCHK (NS,TAU,NT)

C
C   ****************************************************************
C   *                                                              *
C   *     IF THE TARGET IS OUT OF RANGE FOR THIS SENSOR,           *
C   *     SET MFLAG TO 1 AND RETURN.     NO REPORT.                *
C   *                                                              *
C   ****************************************************************
C
00014               IF (DXYZ .LT. RANGE) GO TO 300
00016               MFLAG(NS)=1
00017               RETURN
00018        300    MFLAG(NS)=0
C
C   ****************************************************************
C   *                                                              *
C   *     TIME IS THE TIME WHEN THE SOUND REACHS THE SENSOR.       *
C   *     IT IS THE REPORT TIME.                                   *
C   *                                                              *
C   ****************************************************************
C
00019               TIME=TAU+DXYZ/RCS
C
C   ****************************************************************
C   *                                                              *
C   *     ALTAU WILL BE THE REPORT BEARING                         *
C   *                                                              *
C   ****************************************************************
C
00020               ALTAU=RALTAU+RALROT(NS)
C
C   ****************************************************************
C   *                                                              *
C   *     CALCULATE SIGNAL/NOISE (SN)                              *
C   *                                                              *
C   *     ADD GAUSSIAN ERROR.                                      *
C   *                                                              *
C   ****************************************************************
C
00021               SN=SNCPA/(DXYZ+1.0)
00022               XSIGMA=XK/SQRT(SN)
00023               CALL NORMAL (ERROR)
00024               ALTAU=ALTAU+XMEAN+XSIGMA*ERROR
C
C   ****************************************************************
C   *                                                              *
C   *     ALTAU IS ADJUSTED TO FIT INTO A BEARING BIN.             *
C   *                                                              *
C   ****************************************************************
C
00025               ALTAU=(PI2/BINS)*FLOAT(IFIX(BINS*ALTAU/PI2+0.5))
C
C
C
C
```

```
NOL FORTRAN (1.0)                    SUBROUTINE SENCHK (NS,TAU,NT)

                C
                C   ***************************************************************
                C   *                                                             *
                C   *     W IS THE WEIGHTING FACTOR PASSED FROM THE SENSOR.        *
                C   *                                                             *
                C   ***************************************************************
                C
00026                   W=.001
                C
                C   ***************************************************************
                C   *                                                             *
                C   *     IF THE BEARING HAS NOT CHANGED SINCE THE LAST REPORT     *
                C   *     (ALTAU=ALR), THEN THE SENSOR MUST WAIT AT LEAST FREQ(2)  *
                C   *     SINCE THE LAST REPORT.  WE KNOW THE SENSOR HAS WAITED    *
                C   *     FREQ(1) SECONDS (FROM ABOVE), SO WE WILL REPORT IF EITHER *
                C   *               1. THE BEARING HAS CHANGED                    *
                C   *          OR   2. THE TIME DIFFERENCE .GT. FREQ(2)           *
                C   *                                                             *
                C   ***************************************************************
                C
00027                   IF (( TAU-TLR(NS)) .LT. FREQ(2) .AND.
00027                1  ALTAU .EQ. ALR(NS)) RETURN
                C
                C   ***************************************************************
                C   *                                                             *
                C   *     THE SENSOR IS NOW REPORTING.                            *
                C   *                                                             *
                C   ***************************************************************
                C
00029                   IF (NREP(NS,NT) .GE. 50) RETURN
00031                   TLR(NS)=TAU
00032                   ALR(NS)=ALTAU
00033                   NREP(NS,NT)=NREP(NS,NT)+1
00034                   N=NREP(NS,NT)
00035                   TSAVE(NS,NT,N)=TIME
00036                   ASAVE(NS,NT,N)=ALTAU
00037                   WSAVE(NS,NT,N)=W
00038                   RETURN
00039                   END

              NOL FORTRAN DIAGNOSTIC RESULTS - FOR   SENCHK

NO ERRORS
```

DATA VARIABLES

| | | | | | |
|---|---|---|---|---|---|
| ALR | 00004 | 00027 | 00032 | | |
| ASAVE | 00003 | 00036 | | | |
| BINS | 00004 | 00025 | 00025 | | |
| DELTA | 00004 | | | | |
| FREQ | 00004 | 00006 | 00027 | | |
| INCSAVE | 00003 | | | | |
| MFLAG | 00004 | 00016 | 00018 | | |
| NREP | 00003 | 00029 | 00033 | 00033 | 00034 |
| NRUN | 00004 | | | | |
| NRUNX | 00004 | | | | |
| NUM | 00004 | | | | |
| PI | 00005 | 00012 | | | |
| PI2 | 00005 | 00025 | 00025 | | |
| PID2 | 00005 | 00008 | 00009 | 00010 | |
| RALA | 00001 | 00008 | 00009 | | |
| RALROT | 00002 | 00020 | | | |
| RANGE | 00004 | 00014 | | | |
| RCS | 00001 | 00019 | | | |
| RTA | 00001 | 00008 | 00009 | | |
| RTEMP | 00001 | | | | |
| RVA | 00001 | 00008 | 00009 | | |
| RX0 | 00002 | 00010 | 00011 | 00013 | |
| RXA | 00001 | 00008 | | | |
| RY0 | 00002 | 00010 | 00013 | | |
| RYA | 00001 | 00009 | | | |
| RZA | 00001 | 00013 | | | |
| SAVE | 00004 | | | | |
| SNCPA | 00004 | 00021 | | | |
| TLR | 00004 | 00006 | 00027 | 00031 | |
| TSAVE | 00003 | 00035 | | | |
| WSAVE | 00003 | 00037 | | | |
| XK | 00004 | 00022 | | | |
| XMEAN | 00004 | 00024 | | | |

COMMON VARIABLES

**NONE**

PROGRAM VARIABLES

| | | | | | |
|---|---|---|---|---|---|
| ALTAU | 00020 | 00024 | 00024 | 00025 | 00025 |
| ATAN | 00010 | | | | |
| COS | 00008 | | | | |
| DXYZ | 00013 | 00014 | 00019 | 00021 | |
| ERROR | 00023 | 00024 | | | |
| FLOAT | 00025 | | | | |
| IFIX | 00025 | | | | |
| N | 00034 | 00035 | 00036 | 00037 | |
| NORMAL | 00023 | | | | |
| NS | 00000 | 00006 | 00010 | 00010 | 00011 |

NOL FORTRAN (1.0)  SUBROUTINE SENCHK (NS,TAU,NT)

| | | | | | |
|---|---|---|---|---|---|
| | 00020 | 00027 | 00027 | 00029 | 00031 |
| | 00035 | 00036 | 00037 | | |
| NT | 00000 | 00008 | 00008 | 00008 | 00008 |
| | 00013 | 00029 | 00033 | 00033 | 00034 |
| RALTAU | 00010 | 00012 | 00012 | 00020 | |
| RXTAU | 00008 | 00010 | 00011 | 00013 | |
| RYTAU | 00009 | 00010 | 00013 | | |
| SIN | 00009 | | | | |
| SN | 00021 | 00022 | | | |
| SQRT | 00013 | 00022 | | | |
| TAU | 00000 | 00006 | 00008 | 00009 | 00019 |
| TIME | 00019 | 00035 | | | |
| W | 00026 | 00037 | | | |
| XSIGMA | 00022 | 00024 | | | |

OUTPUT DECK/ L.R.X

```
NOL FORTRAN (1.0)

00000                SUBROUTINE OUTPUT
                C
                C   ***********************************************************************
                C   *                                                                     *
                C   *     JONATHAN VALVANO                                  03/16/75      *
                C   *                                                                     *
                C   *     THIS ROUTINE OUTPUTS RESULTS.                                   *
                C   *     ALL ANGLE OUTPUTS ARE IN DEGREES.                               *
                C   *                                                                     *
                C   ***********************************************************************
                C
00001                COMMON /TRGT// ALA(3),XA(3),YA(3),ZA(3),
00001               1   TA(3),VA(3),AT(3),BT(3),CT(3),TEMP,CS,NTRG
00002                COMMON /RTRGT// RALA(3),RXA(3),RYA(3),RZA(3),
00002               1   RTA(3),RVA(3),RTEMP,RCS
00003                COMMON /SEN// NSEN,X0(3),Y0(3),ALROT(3),IFLAG,JFLAG
0000*                COMMON /RSEN// RX0(3),RY0(3),RALROT(3)
00005                COMMON /REP// TSAVE(3,3,50),ASAVE(3,3,50),WSAVE(3,3,50),
00005               1   NREP(3,3),TNCSAVE(3,3)
00006                COMMON /STATE// TLR(3),ALR(3),MFLAG(3),FREQ(2),
00006               1   BINS,XMEAN,XK,SNCPA,RANGE,DELTA,NUM,NRUN,NRUNX,SAVE(10,100)
00007                COMMON /OPTIONS// KFLAG,M,NPASS
00008                COMMON /CONST// PI,PI02,PI2
00009                PRINT 604, NRUNX
00010           604  FORMAT (1H1,50X,5HRUN  ,I5)
00011                IF (KFLAG .EQ. 0) PRINT 611
00013           611  FORMAT (1H3,10X,15HBIAS CALCULATED)
00014                IF (KFLAG .EQ. 1) PRINT 612
00016           612  FORMAT (1H3,10X,12HBIAS INPUTED)
00017                PRINT 615
00018           615  FORMAT (1H ,10X,14HREPORTED   W=1,17X,
00018               1   32HCALCULATED W=EXP(-(RXY/1000)**2))
00019                PRINT 614, NPASS,M
00020           614  FORMAT (1H ,10X,18HNUMBER OF PASSES  ,I8,5X,
00020               1   21HINCREMENT FACTOR M   ,I8)
00021                XMEAND=XMEAN*180.0/PI
00022                PRINT 602,XMEAND,RANGE
00023           602  FORMAT (1H ,10X,4HMEAN,14X,F8.1,5X,5HRANGE,16X,F8.0)
00024                PRINT 603, XK,SNCPA
00025           603  FORMAT (1H ,10X,8HK FACTOR,10X,F8.4,5X,10HS/N AT CPA,11X,F8.2)
00026                PRINT 600, FREQ(1),FREQ(2)
00027           600  FORMAT (1H ,10X,18HCHANGE CYCLE TIME ,F8.2,
00027               1   5X,21HNO CHANGE CYCLE TIME ,F8.2)
00028                PRINT 601,  BINS,DELTA
00029           601  FORMAT (1H ,10X,18HNUMBER OF BINS    ,F8.0,5X,
00029               1   21HTIME INCREMENT       ,F8.2)
                C
                C
                C
                C
                C
                C
                C
                C
```



COPY AVAILABLE TO DDC DOES NOT
PERMIT FULLY LEGIBLE PRODUCTION

```
NOL FORTRAN (1.0)                    SUBROUTINE OUTPUT


            C
00030                 DO 610 NT=1,NTRG
00031                 PRINT 605, NT
00032       605       FORMAT (1H3,10X,5HTARG ,I5,
00032             1   56H       XA        YA        ZA        VA  COURSE     TEMP        CS)
00033                 RALAU=RALA(NT)*180.0/PI
00034                 PRINT 606, RXA(NT),RYA(NT),RZA(NT),RVA(NT),RALAU,RTEMP,RCS
00035       606       FORMAT (1H ,10X,10HREAL       ,7F8.2)
00036                 ALAU=ALA(NT)*180.0/PI
00037                 PRINT 607, XA(NT),YA(NT),ZA(NT),VA(NT),ALAU,TEMP,CS
00038       607       FORMAT (1H ,10X,10HINPUTED    ,7F8.2)
00039       610       CONTINUE
00040                 DO 650 NS=1,NSEN
00041                 PRINT 620, NS
00042       620       FORMAT (1H3,10X,14HSENSOR NUMBER ,I5)
00043                 PRINT 621
00044       621       FORMAT (1H ,10X,20X,
00044             1   36H            X0            Y0        ALROT)
00045                 RALROTD=RALROT(NS)*180.0/PI
00046                 PRINT 622, RX0(NS),RY0(NS),RALROTD
00047       622       FORMAT (1H ,10X,4HREAL,16X,3F12.5)
00048                 ALROTD=ALROT(NS)*180.0/PI
00049                 PRINT 623, X0(NS),Y0(NS),ALROTD
00050       623       FORMAT (1H ,10X,10HCALCULATED,10X,3F12.5)
00051                 DX=RX0(NS)-X0(NS)
00052                 DY=RY0(NS)-Y0(NS)
00053                 DAL=RALROTD-ALROTD
00054                 PRINT 624, DX,DY,DAL
00055       624       FORMAT (1H ,10X,10HDIFFERENCE,10X,3F12.5)
00056                 DR=SQRT(DX**2+DY**2)
00057                 PRINT 625, DR
00058       625       FORMAT (1H0,10X,12HRADIAL ERROR,8X,F12.5)
00059                 PRINT 616, (INCSAVE(NS,I), I=1,3)
00060       616       FORMAT (1H ,10X,10HINCREMENTS,10X,3I12)
00061       650       CONTINUE
00062                 RETURN
00063                 END

            NOL FORTRAN DIAGNOSTIC RESULTS - FOR   OUTPUT


NO ERRORS
```



COPY AVAILABLE TO DDC DOES NOT
PERMIT FULLY LEGIBLE PRODUCTION

DATA VARIABLES

| | | | | | |
|---|---|---|---|---|---|
| ALA | 00001 | 00036 | | | |
| ALR | 00006 | | | | |
| ALROT | 00003 | 00048 | | | |
| ASAVE | 00005 | | | | |
| AT | 00001 | | | | |
| BINS | 00006 | 00028 | | | |
| BT | 00001 | | | | |
| CS | 00001 | 00037 | | | |
| CT | 00001 | | | | |
| DELTA | 00006 | 00028 | | | |
| FREQ | 00006 | 00026 | 00026 | | |
| IFLAG | 00003 | | | | |
| INCSAVE | 00005 | 00059 | | | |
| JFLAG | 00003 | | | | |
| KFLAG | 00007 | 00011 | 00014 | | |
| M | 00007 | 00019 | | | |
| MFLAG | 00006 | | | | |
| NPASS | 00007 | 00019 | | | |
| NREP | 00005 | | | | |
| NRUN | 00006 | | | | |
| NRUNX | 00006 | 00009 | | | |
| NSEN | 00003 | 00040 | | | |
| NTRG | 00001 | 00030 | | | |
| NUM | 00006 | | | | |
| PI | 00008 | 00021 | 00033 | 00036 | 00045 |
| PI2 | 00008 | | | | |
| PID2 | 00008 | | | | |
| RALA | 00002 | 00033 | | | |
| RALROT | 00004 | 00045 | | | |
| RANGE | 00006 | 00022 | | | |
| RCS | 00002 | 00034 | | | |
| RTA | 00002 | | | | |
| RTEMP | 00002 | 00034 | | | |
| RVA | 00002 | 00034 | | | |
| RXO | 00004 | 00046 | 00051 | | |
| RXA | 00002 | 00034 | | | |
| RYO | 00004 | 00046 | 00052 | | |
| RYA | 00002 | 00034 | | | |
| RZA | 00002 | 00034 | | | |
| SAVE | 00006 | | | | |
| SNCPA | 00006 | 00024 | | | |
| TA | 00001 | | | | |
| TEMP | 00001 | 00037 | | | |
| TLR | 00006 | | | | |
| TSAVE | 00005 | | | | |
| VA | 00001 | 00037 | | | |
| WSAVE | 00005 | | | | |
| XO | 00003 | 00049 | 00051 | | |
| XA | 00001 | 00037 | | | |
| XK | 00006 | 00024 | | | |
| XMEAN | 00006 | 00021 | | | |
| YO | 00003 | 00049 | 00052 | | |



COPY AVAILABLE TO DDC DOES NOT PERMIT FULLY LEGIBLE PRODUCTION

| | | | | | |
|---|---|---|---|---|---|
| YA | 00001 | 00037 | | | |
| ZA | 00001 | 00037 | | | |

COMMON VARIABLES

**NONE**

PROGRAM VARIABLES

| | | | | | |
|---|---|---|---|---|---|
| ALAD | 00036 | 00037 | | | |
| ALROTD | 00048 | 00049 | 00053 | | |
| DAL | 00053 | 00054 | | | |
| DR | 00056 | 00057 | | | |
| DX | 00051 | 00054 | 00056 | | |
| DY | 00052 | 00054 | 00056 | | |
| I | 00059 | 00059 | | | |
| NS | 00040 | 00041 | 00045 | 00046 | 00046 |
| | 00051 | 00052 | 00052 | 00059 | |
| NT | 00030 | 00031 | 00033 | 00034 | 00034 |
| | 00037 | 00037 | 00037 | | |
| RALAD | 00033 | 00034 | | | |
| RALROTD | 00045 | 00046 | 00053 | | |
| SQRT | 00056 | * | | | |
| XMEAND | 00021 | 00022 | | | |



COPY AVAILABLE TO DDC DOES NOT PERMIT FULLY LEGIBLE PRODUCTION

SPL DECK/ L,R,X



COPY AVAILABLE TO DDC DOES NOT
PERMIT FULLY LEGIBLE PRODUCTION

```
NOL FORTRAN (1.0)

00000                SUBROUTINE SPL(NS)
                C
                C
                C    ********************************************************************
                C    *                                                                  *
                C    *    JONATHAN VALVANO                                  03/18/75    *
                C    *                                                                  *
                C    *                                                                  *
                C    *    THIS IS THE SENSOR POSITION LOCATOR ROUTINE.                  *
                C    *                                                                  *
                C    *                                                                  *
                C    *    INPUTS              NS        NUMBER OF SENSOR              *
                C    *                        NPASS     NUMBER OF PASSES USED         *
                C    *                        NTRG      NUMBER OF TARGETS             *
                C    *                        TSAVE(NS,NT,NR)=TIME OF REPORT          *
                C    *                        ASAVE(NS,NT,NR)=ANGLE OF REPORT         *
                C    *                        WSAVE(NS,NT,NR)=W OF REPORT             *
                C    *                        NREP(NS,NT)=NUMBER OF REPORTS           *
                C    *                                                                *
                C    *    OUTPUTS             X0(NS)    X POSITION OF THE SENSOR      *
                C    *                        Y0(NS)    Y POSITION OF THE SENSOR      *
                C    *                        ALROT(NS) ANGLE ROTATION BIAS           *
                C    *                                                                *
                C    *                                                                *
                C    *    GIVEN THE REPORTS SAVED IN THE ARRAYS THIS ROUTINE          *
                C    *    CALCULATES THE SENSOR POSITION AND BIAS ANGLE.              *
                C    *    THE BIAS IS THE FIRST THING CALCULATED. IT USES THREE       *
                C    *    REPORTS AND USING  ANGLE DIFFERENCES CALCULATES THE BIAS.   *
                C    *    THE SENSOR POSITION  IS CALCULATED BY  A LEAST  SQUARES     *
                C    *    FIT OF A POINT TO THE MANY LINES (ONE FOR EACH REPORT)      *
                C    *    FROM THE TARGET TO THE SENSOR.                              *
                C    *    ENTER CALCULATES THESE LINES AND MAKES THE APPROPRIATE      *
                C    *    SUMS INTO AI2,BI2,AIBI,AICI,BICI.  THE SENSOR POSITION      *
                C    *    IS A SIMPLE RELATION OF THESE VARIABLES.                    *
                C    *                                                                *
                C    *    THERE IS THE POSSIBILITY OF INPUTED THE BIAS (ALROT.)       *
                C    *    IF THIS OPTION IS TAKEN, KFLAG SHOULD BE SET (KFLAG=1).     *
                C    *    IF KLAG IS RESET (KFLAG=0) THEN THE PROGRAM WILL CALCULATE  *
                C    *    ITS OWN BIAS.                                               *
                C    *                                                                *
                C    ********************************************************************
                C
00001                COMMON /TRGT// ALA(3),XA(3),YA(3),ZA(3),TA(3),VA(3),
00001               1   AT(3),BT(3),CT(3),TEMP,CS,NTRG
00002                COMMON /SEN// NSEN,X0(3),Y0(3),ALROT(3),IFLAG,JFLAG
00003                COMMON /REP// TSAVE(3,3,50),ASAVE(3,3,50),WSAVE(3,3,50),
00003               1   NREP(3,3),INCSAVE(3,3)
00004                COMMON /CONST// PI,PID2,PI2
00005                COMMON /SUMS// AI2,BI2,AIBI,AICI,BICI
00006                COMMON /OPTIONS// KFLAG,M,NPASS
                C
                C
                C
                C
```



COPY AVAILABLE TO DDC DOES NOT PERMIT FULLY LEGIBLE PRODUCTION

```
                      SUBROUTINE SPL(NS)

C     ****************************************************************
C     *                                                              *
C     *     WHEN CALCULATING THE BIAS IT IS BETTER NOT TO USE        *
C     *     SEQUENCIAL REPORTS BECAUSE THE ANGLE DIFFERENCES ARE     *
C     *     EITHER TOO SMALL OR ZERO.  THEREFORE WE WILL INCREMENT   *
C     *     THROUGH THE ARRAYS  STEPPING MORE THAN ONE. INC IS THE   *
C     *     VALUE OF THE INCREMENT FOR EACH TARGET THE SENSOR SAW.   *
C     *     IF INC(1)=4 THEN WHEN WE LOOK AT THE REPORTS OF TARGET 1,*
C     *     WE WILL STEP THROUGH BY 4 (1,5,9,13,...).                *
C     *     IN ORDER TO USE AS MUCH OF THE DATA AS POSSIBLE,  WE     *
C     *     WILL STEP THROUGH THE ARRAYS AGAIN BUT STARTING WITH 2.  *
C     *     THEN 3.  THEN 4.  IF NREP(NS,1)=50 THEN INC(1) WOULD BE 13 *
C     *     AND WE WOULD LOOK AT THE REPORTS IN THIS ORDER---        *
C     *     1,14,27,40  2,15,28,41 3,16,29,41 .... 13,26,39          *
C     *     BIAS CALCULATIONS USE REPORTS IN GROUPS OF THREE.         *
C     *     POSITION CALCULATION S USE EACH REPORT SEPERATELY.       *
C     *                                                              *
C     ****************************************************************
C
00007                 DO 400 NT=1,NTRG
00008                 INCSAVE(NS,NT)=1+NREP(NS,NT)/M
00009          400    CONTINUE
C
C     ****************************************************************
C     *                                                              *
C     *     THE POSITION CALCULATION CAN NOT BE DONE WITHOUT  A BIAS *
C     *     CALCULATION . BUT THE BIAS CALCULATION CAN BE DONE WITH  *
C     *     OR WITHOUT A POSITION CALCULATION . THE BIAS CALCULATION *
C     *     IS MORE ACCURATE WHEN  POSITION IS KNOWN.  TO KEEP       *
C     *     TRACK OF  WHICH CALCULATIONS HAVE BEEN DONE FLAGS        *
C     *     ARE USED.              IFLAG=0   MEANS NO BIAS CALCULATION *
C     *                            IFLAG=1   MEANS A BIAS CALCULATION  *
C     *                            JFLAG=0   MEANS NO POSITION CALCULATION *
C     *                            JFLAG=1   MEANS A POSITION CALCULATION *
C     *                            KFLAG=0   MEANS BIAS TO BE CALCULATED *
C     *                            KFLAG=1   MEANS BIAS HAS BEEN INPUTED *
C     *                                                              *
C     ****************************************************************
00010          401    IFLAG=0
00011                 JFLAG=0
00012                 DO 490 NP=1,NPASS
C     ****************************************************************
C     *                                                              *
C     *     EACH PASS STARTS WITH FRESH  SUMS BUT USES THE CALCULATIONS *
C     *     FROM THE PREVIOUS PASS.                                  *
C     *                                                              *
C     ****************************************************************
00013                 AI2=0.0
00014                 BI2=0.0
00015                 AIBI=0.0
00016                 AICI=0.0
00017                 BICI=0.0
00018                 BSI=0.0
00019                 WI=0.0
```



COPY AVAILABLE TO DDC DOES NOT PERMIT FULLY LEGIBLE PRODUCTION

```
NOL FORTRAN (1.0)                    SUBROUTINE SPL(NS)

C
C
C
C    ********************************************************************
C    *                                                                  *
C    *    IF DATA FROM TWO OR MORE TARGETS IS AVAILABLE THEN            *
C    *    CALCULATIONS ARE GENERALLY MORE ACCURATE.                     *
C    *    HERE, WE LOOP THROUGH FOR EACH TARGET.                        *
C    *                                                                  *
C    ********************************************************************
C
C
00020          DO 480 NT=1,NTRG
C
C
C    ********************************************************************
C    *                                                                  *
C    *    NRPT IS THE NUMBER OF REPORTS  OF THIS TARGET.                *
C    *    N  IS THE INCREMENT USED  WITH THE REPORTS OF THIS TARGET.    *
C    *                                                                  *
C    ********************************************************************
C
C
00021          NRPT=NREP(NS,NT)
00022          N=INCSAVE(NS,NT)
C
C    ********************************************************************
C    *                                                                  *
C    *    IF WE ARE GOING TO INCREMENT BY N THEN THERE MUST BE          *
C    *    AT LEAST 3*N POINTS IN THE ARRAYS ,IF NOT GO TO NEXT TARGET   *
C    *                                                                  *
C    ********************************************************************
C
00023          IF (NRPT .LT. 3*N) GO TO 480
C
C
C
```

```
NOL FORTRAN (1.0)                    SUBROUTINE SPL(NS)

C
C
00025          DO 470 NFIRST=1,N
C
C
C    ********************************************************************
C    *                                                                  *
C    *                                                                  *
C    *     N1 IS THE NUMBER OF THE FIRST REPORT.                        *
C    *     N2 IS THE NUMBER OF THE SECOND REPORT.                       *
C    *     N3 IS THE NUMBER OR THE THIRD REPORT..                       *
C    *                                                                  *
C    *     T1 IS THE TIME OF THE FIRST REPORT.                          *
C    *     T2 IS THE TIME OF THE SECOND REPORT.                         *
C    *     T3 IS THE TIME OF THE THIRD REPORT .                         *
C    *                                                                  *
C    *     (X1,Y1) IS THE TARGET POSITION AT THE FIRST REPORT.          *
C    *     (X2,Y2) IS THE TARGET POSITION AT THE SECOND REPORT.         *
C    *     (X3,Y3) IS THE TARGET POSITION AT THE THIRD REPORT.          *
C    *                                                                  *
C    *     ALPHA1 IS THE REPORTED BEARING OF THE FIRST REPORT.          *
C    *     ALPHA2 IS THE REPORTED BEARING OF THE SECOND REPORT.         *
C    *     ALPHA3 IS THE REPORTED BEARING OF THE THIRD REPORT.          *
C    *                                                                  *
C    *     W1 IS THE WEIGHTING FACTOR ASSOCIATED WITH THE FIRST REPORT  *
C    *     W2 IS THE WEIGHTING FACTOR ASSOCIATED WITH THE SECOND REPORT *
C    *     W3 IS THE WEIGHTING FACTOR ASSOCIATED WITH THE THIRD REPORT  *
C    *                                                                  *
C    *     THETA1 IS THE ANGLE DIFFERENCE BETWEEN FIRST AND SECOND      *
C    *     THETA2 IS THE ANGLE DIFFERENCE BETWEEN THE SECOND AND THIRD  *
C    *                                                                  *
C    *     D12 IS THE DISTANCE TRAVELLED BETWEEN  THE FIRST AND SECOND  *
C    *     D23 IS THE DISTANCE TRAVELLED BETWEEN THE SECOND AND THIRD   *
C    *                                                                  *
C    *                                                                  *
C    ********************************************************************
C
C
00026          N1=NFIRST
00027          N2=N1+N
00028          NSTART=N2+N
00029          CALL ENTER (NS,NT,N1,T1,X1,Y1,ALPHA1,W1)
00030          CALL ENTER (NS,NT,N2,T2,X2,Y2,ALPHA2,W2)
00031          IF (KFLAG .EQ. 1) GO TO 410
00033          D12=SQRT((X1-X2)**2+(Y1-Y2)**2)
00034          THETA1=ALPHA2-ALPHA1
C
C    ********************************************************************
C    *                                                                  *
C    *     SHIFT WILL SHIFT THETA1 BETWEEN -PI AND +PI                  *
C    *                                                                  *
C    ********************************************************************
C
00035          CALL SHIFT (THETA1,0.0)
```



COPY AVAILABLE TO DDC DOES NOT
PERMIT FULLY LEGIBLE PRODUCTION

```
NOL FORTRAN (1.0)                    SUBROUTINE SPL(NS)

C
C
C   ***********************************************************************
C   *                                                                     *
C   *                                                                     *
C   *     THE BIAS ALGORITHM WILL NOT WORK IF THETA1 EQUALS ZERO.         *
C   *     THE INCREMENT IS INCREASED AND WE START OVER AGAIN.             *
C   *                                                                     *
C   *                                                                     *
C   ***********************************************************************
C
C
00036               IF (THETA1 .NE. 0.0) GO TO 410
00038               INCSAVE(NS,NT)=INCSAVE(NS,NT)+1
00039               GO TO 401
00040        410    DO 460 N3=NSTART,NRPT,N
00041               CALL ENTER (NS,NT,N3,T3,X3,Y3,ALPHA3,W3)
00042               IF (KFLAG .EQ. 1) GO TO 460
00044               D23=SQRT((X3-X2)**2+(Y3-Y2)**2)
00045               THETA2=ALPHA3-ALPHA2
C
C
C   ***********************************************************************
C   *                                                                     *
C   *     SHIFT WILL SHIFT THETA2 BETWEEN -PI AND +PI                     *
C   *                                                                     *
C   ***********************************************************************
C
C
00046               CALL SHIFT (THETA2,0.0)
C
C
C   ***********************************************************************
C   *                                                                     *
C   *                                                                     *
C   *     THE ALGORITHM WILL NOT WORK IF                                  *
C   *               THETA1 EQUALS ZERO                                    *
C   *               THETA2 EQUALS ZERO                                    *
C   *               THETA1 AND THETA2 HAVE OPPOSITE SIGNS.                *
C   *     IF ANY OF THESE FAULTS OCCUR THE ALGORITM IS SKIPPED.           *
C   *                                                                     *
C   *                                                                     *
C   ***********************************************************************
C
C
00047               IF (THETA1*THETA2 .LE. 0.0) GO TO 460
C
C
C
C
C
C
C
```



COPY AVAILABLE TO DDC DOES NOT PERMIT FULLY LEGIBLE PRODUCTION

```
C
C
C     ********************************************************************
C     *                                                                  *
C     *                                                                  *
C     *     THIS IS THE BIAS CALCULATION.                                *
C     *                                                                  *
C     *                                                                  *
C     ********************************************************************
C
00049           THETA12=THETA1+THETA2
00050           DENOM=D12*SIN(THETA2)/(D23*SIN(THETA1))-COS(THETA12)
00051           PHI=ATAN(SIN(THETA12)/DENOM)
00052           IF (DENOM .LT. 0.0) PHI=PHI+PI
00054           BS=PI+ALPHA1-ALA(NT)-PHI
C
C     ********************************************************************
C     *                                                                  *
C     *     SHIFT WILL SHIFT BS BTWEEN (ALROT-PI) AND (ALROT+PI)         *
C     *                                                                  *
C     *     EACH BIAS CALCULATION (BS) IS AVERAGED INTO BSI WITH A       *
C     *     WEIGHTING FACTOR OF W2.                                      *
C     *                                                                  *
C     ********************************************************************
C
00055           CALL SHIFT (BS,ALROT(NS))
00056           BSI=BSI+W2*BS
00057           WI=WI+W2
C
C     ********************************************************************
C     *                                                                  *
C     *     THIS FOLLOWING VARIABLES ARE BUMPED DOWN SO                  *
C     *     THEY WILL NOT HAVE TO BE RECALCULATED.                       *
C     *                                                                  *
C     ********************************************************************
C
00058           N1=N2
00059           N2=N3
00060           X2=X3
00061           Y2=Y3
00062           THETA1=THETA2
00063           ALPHA1=ALPHA2
00064           ALPHA2=ALPHA3
00065           D12=D23
00066           W2=W3
00067    460    CONTINUE
C
C
C
C
C
C
C
```



COPY AVAILABLE TO DDC DOES NOT
PERMIT FULLY LEGIBLE PRODUCTION

```
NOL FORTRAN (1.0)                    SUBROUTINE SPL(NS)

C
C   ***************************************************************
C   *                                                             *
C   *     WE HAVE NOW FINISHED ONE LOOP THROUGH THE ARRAYS.       *
C   *     CONTROL WILL BE HERE ABOUT INC(NT) TIMES FOR EACH       *
C   *     TARGET AND FOR EACH PASS AND FOR EACH SENSOR.           *
C   *                                                             *
C   ***************************************************************
C
C   ***************************************************************
C   *                                                             *
C   *     IF WI EQUALS ZERO THAT MEANS NO BIAS CALCULATIONS HAVE  *
C   *     BEEN DONE. IF THIS IS SO THEN NO AVERAGING CAN BE DONE. *
C   *                                                             *
C   ***************************************************************
C
00068              IF (WI .EQ. 0.0 .OR. KFLAG .EQ. 1) GO TO 465
C
C   ***************************************************************
C   *                                                             *
C   *     CALCULATE THE AVERAGE BIAS AND SET IFLAG.               *
C   *                                                             *
C   ***************************************************************
C
00070              ALROT(NS)=RSI/WI
00071              IFLAG=1
C
00072       465    DENOM=AI2**2*I2-AIBI**2
C
C   ***************************************************************
C   *                                                             *
C   *     IF DENOM EQUALS ZERO THEN WE HAVE CALCULATED ONLY ONE OR*
C   *     NO LINES THROUGH THE SENSOR. IN EITHER CASE A SENSOR    *
C   *     POSITION CALCULATION CAN NOT BE DONE AND IS SKIPPED.    *
C   *                                                             *
C   ***************************************************************
C
00073              IF (DENOM .EQ. 0.0) GO TO 470
C
C   ***************************************************************
C   *                                                             *
C   *     THE SENSOR POSITION LOCATION IS CALCULATED. AND JFLAG SET.*
C   *                                                             *
C   ***************************************************************
C
00075              X0(NS)=(BICI*AIBI-AICI*BI2)/DENOM
00076              Y0(NS)=(AICI*AIBI-BICI*AI2)/DENOM
00077              JFLAG=1
00078       470    CONTINUE
00079       480    CONTINUE
00080       490    CONTINUE
00081              RETURN
00082              END

NOL FORTRAN DIAGNOSTIC RESULTS - FOR   SPL
```



NO ERRORS

COPY AVAILABLE TO DDC DOES NOT PERMIT FULLY LEGIBLE PRODUCTION

DATA VARIABLES

| | | | | | |
|---|---|---|---|---|---|
| AI2 | 00005 | 00013 | 00072 | 00076 | |
| AIBI | 00005 | 00015 | 00072 | 00075 | 00076 |
| AICI | 00005 | 00016 | 00075 | 00076 | |
| ALA | 00001 | 00054 | | | |
| ALROT | 00002 | 00055 | 00070 | | |
| ASAVE | 00003 | | | | |
| AT | 00001 | | | | |
| BI2 | 00005 | 00014 | 00072 | 00075 | |
| BICI | 00005 | 00017 | 00075 | 00076 | |
| BT | 00001 | | | | |
| CS | 00001 | | | | |
| CT | 00001 | | | | |
| IFLAG | 00002 | 00010 | 00071 | | |
| INCSAVE | 00003 | 00008 | 00022 | 00038 | 00038 |
| JFLAG | 00002 | 00011 | 00077 | | |
| KFLAG | 00006 | 00031 | 00042 | 00068 | |
| M | 00006 | 00008 | | | |
| NPASS | 00006 | 00012 | | | |
| NREP | 00003 | 00008 | 00021 | | |
| NSEN | 00002 | | | | |
| NTRG | 00001 | 00007 | 00020 | | |
| PI | 00004 | 00053 | 00054 | | |
| PI2 | 00004 | | | | |
| PID2 | 00004 | | | | |
| TA | 00001 | | | | |
| TEMP | 00001 | | | | |
| TSAVE | 00003 | | | | |
| VA | 00001 | | | | |
| WSAVE | 00003 | | | | |
| X0 | 00002 | 00075 | | | |
| XA | 00001 | | | | |
| Y0 | 00002 | 00076 | | | |
| YA | 00001 | | | | |
| ZA | 00001 | | | | |

COMMON VARIABLES

**NONE**

PROGRAM VARIABLES

| | | | | | |
|---|---|---|---|---|---|
| ALPHA1 | 00029 | 00034 | 00054 | 00063 | |
| ALPHA2 | 00030 | 00034 | 00045 | 00063 | 00064 |
| ALPHA3 | 00041 | 00045 | 00064 | | |
| ATAN | 00051 | | | | |
| BS | 00054 | 00055 | 00056 | | |
| BSI | 00018 | 00056 | 00056 | 00070 | |
| COS | 00050 | | | | |
| D12 | 00033 | 00050 | 00065 | | |
| D23 | 00044 | 00050 | 00065 | | |



COPY AVAILABLE TO DDC DOES NOT PERMIT FULLY LEGIBLE PRODUCTION

| | | | | | |
|---|---|---|---|---|---|
| DENOM | 00050 | 00051 | 00052 | 00072 | 00073 |
| ENTER | 00029 | 00030 | 00041 | | |
| N | 00022 | 00023 | 00025 | 00027 | 00028 |
| N1 | 00026 | 00027 | 00029 | 00058 | |
| N2 | 00027 | 00028 | 00030 | 00058 | 00059 |
| N3 | 00040 | 00041 | 00059 | | |
| NFIRST | 00025 | 00026 | | | |
| NP | 00012 | | | | |
| NRPT | 00021 | 00023 | 00040 | | |
| NS | 00000 | 00008 | 00008 | 00021 | 00022 |
| | 00041 | 00055 | 00070 | 00075 | 00076 |
| NSTART | 00028 | 00040 | | | |
| NT | 00007 | 00008 | 00008 | 00020 | 00021 |
| | 00038 | 00041 | 00054 | | |
| PHI | 00051 | 00053 | 00053 | 00054 | |
| SHIFT | 00035 | 00046 | 00055 | | |
| SIN | 00050 | 00050 | 00051 | | |
| SQRT | 00033 | 00044 | | | |
| T1 | 00029 | | | | |
| T2 | 00030 | | | | |
| T3 | 00041 | | | | |
| THETA1 | 00034 | 00035 | 00036 | 00047 | 00049 |
| THETA12 | 00049 | 00050 | 00051 | | |
| THETA2 | 00045 | 00046 | 00047 | 00049 | 00050 |
| W1 | 00029 | | | | |
| W2 | 00030 | 00056 | 00057 | 00066 | |
| W3 | 00041 | 00066 | | | |
| WI | 00019 | 00057 | 00057 | 00066 | 00070 |
| X1 | 00029 | 00033 | | | |
| X2 | 00030 | 00033 | 00044 | 00060 | |
| X3 | 00041 | 00044 | 00060 | | |
| Y1 | 00029 | 00033 | | | |
| Y2 | 00030 | 00033 | 00044 | 00061 | |
| Y3 | 00041 | 00044 | 00061 | | |



COPY AVAILABLE TO DDC DOES NOT PERMIT FULLY LEGIBLE PRODUCTION

AD-A032 850 NAVAL SURFACE WEAPONS CENTER WHITE OAK LAB SILVER SP--ETC F/G 17/1
SENSOR POSITION LOCATOR.(U)
SEP 75 J VALVANO

UNCLASSIFIED NSWC/WOL/TR 76-8 NL

2 OF 2
AD
A032 850



END

DATE
FILMED
1-77

```
ENTER    DECK/    L.R.X
```



COPY AVAILABLE TO DDC DOES NOT
PERMIT FULLY LEGIBLE PRODUCTION

```
NOL FORTRAN (1.0)

00000            SUBROUTINE ENTER (NS,NT,NR,T,XT,YT,ALPHA,W)
                 C
                 C
                 C  ***************************************************************
                 C  *                                                             *
                 C  *    JONATHAN VALVANO                              03/15/75   *
                 C  *                                                             *
                 C  *                                                             *
                 C  *    ENTER IS THE ROUTINE THAT CALCULATES THE LINE FROM       *
                 C  *    THE TARGET TO THE SENSOR AND MAKES THE APPROPRIATES SUMS  *
                 C  *    USED IN THE SPL ALGORITHM.                               *
                 C  *                                                             *
                 C  *                                                             *
                 C  *    INPUTS    NS THE NUMBER OF THE SENSOR                    *
                 C  *    NT IS THE NUMBER OF THE TARGET                           *
                 C  *    NR THE NUMBER OF THE REPORT                              *
                 C  *    TSAVE(NS,NT,NR) THE TIME OF THIS REPORT                  *
                 C  *    ASAVE(NS,NT,NR) THE BEARING OF THE REPORT                *
                 C  *    WSAVE(NS,NT,NR) THE WEIGHTING FACTOR PASSED BY THE SENSOR*
                 C  *    AT,BT,CT WHICH FORM THELINE AT*X+BT*Y+CT=0 OF TARGET TRACK*
                 C  *    ZA(NT) WHICH IS THE TARGETS ALTITUDE                     *
                 C  *    TXY WHICH RETURNS THE TARGETS POSITION AS A FUNCTION OF T*
                 C  *    ALROT WHICH  IS THE SENSORS ANGLE ROTATION BIAS          *
                 C  *                                                             *
                 C  *                                                             *
                 C  *    OUTPUTS   T THE TIME OF THE REPORT                       *
                 C  *    XT,YT THE POSITION OF THE TARGET AT THE TIME OF THE REPORT*
                 C  *    ALPHA THE REPORT BEARING (CORRECTED FOR THE SPEED OF SOUND)*
                 C  *    W THE COMBINED WEIGHTING FACTOR REPORTED AND CALCULATED  *
                 C  *    SUMS IN AI2,BI2,AIBI,AICI,BICI  FOR SPL ALGORITHM        *
                 C  *                                                             *
                 C  *                                                             *
                 C  ***************************************************************
                 C
                 C  ***************************************************************
                 C  *                                                             *
                 C  *    THESE ARE THE COMMON STATEMENTS.                         *
                 C  *                                                             *
                 C  ***************************************************************
                 C
00001                 COMMON /TRGT// ALA(3),XA(3),YA(3),ZA(3),TA(3),VA(3),
00001                1  AT(3),BT(3),CT(3),TEMP,CS,NTRG
00002                 COMMON /SEN// NSEN,X0(3),Y0(3),ALROT(3),IFLAG,JFLAG
00003                 COMMON /REP// TSAVE(3,3,50),ASAVE(3,3,50),WSAVE(3,3,50),
00003                1  NREP(3,3),INCSAVE(3,3)
00004                 COMMON /CONST// PI,PID2,PI2
00005                 COMMON /SUMS// AI2,BI2,AIBI,AICI,BICI
00006                 COMMON /OPTIONS// KFLAG,M,NPASS
                 C
                 C
                 C
                 C
                 C
```



COPY AVAILABLE TO DDC DOES NOT
PERMIT FULLY LEGIBLE PRODUCTION

```
NOL FORTRAN (1.0)                    SUBROUTINE ENTER (NS,NT,NR,T,XT,YT,ALPHA,W)

00007                T=TSAVE(NS,NT,NR)
00008                ALPHA=ASAVE(NS,NT,NR)
00009                W=WSAVE(NS,NT,NR)
00010                CALL TAY (XT,YT,T,NT)
               C
               C     ****************************************************************
               C     *                                                              *
               C     *     IF WE ARE GOING TO CALCULATE THE BIAS (KFLAG=0) AND      *
               C     *     IF A BIAS CALCULATION HAS NOT BEEN DONE THEN RETURN      *
               C     *                                                              *
               C     ****************************************************************
               C
00011                IF (IFLAG .EQ. 0 .AND. KFLAG .EQ. 0) RETURN
               C
               C     ****************************************************************
               C     *                                                              *
               C     *     THE RATIO IS USED IN THE ALGORITHM TO ACCOUNT FOR THE    *
               C     *     SPEED OF SOUND . THE TARGET SPEED (VA) MUST BE LESS THAN *
               C     *     THE SPEED OF SOUND(CS). THIS ALONE CORRECTS ONLY THE     *
               C     *     TWO DIMENSIONAL CASE. IF A SENSOR CALCULATION HAS BEEN   *
               C     *     DONE BEFORE (JFLAG=1) THEN THIS ALGORITHM CAN CORRECT    *
               C     *     THE THREE DIMENSIONAL CASE. IF NO SENSOR POSITION        *
               C     *     LOCATION HAS BEEN DONE (JFLAG=0) THEN  JUST USE  THE 2-D.*
               C     *                                                              *
               C     ****************************************************************
               C
00013                BETA=PI+ALA(NT)-(ALPHA-ALROT(NS))
00014                RATIO=CS/VA(NT)
00015                RXYTAU=10000.0
00016                IF (JFLAG .EQ. 0) GO TO 500
               C
               C     ****************************************************************
               C     *                                                              *
               C     *     (XP,YP) IS THE POINT ON THE TARGET TRACK AT CPA          *
               C     *     THE POINT IS USED IN THE ALGORITHM TO CORRECT FOR THE     *
               C     *     SPEED OF SOUND DUE TO THE ALTITUDE.  THE RATIO IS        *
               C     *     RECALCULATED  TAKING THE ALTITUDE INTO EFFECT.           *
               C     *     RXYTAU IS THE TWO DIMENSIONAL RADIAL DISTANCE FROM       *
               C     *     THE SENSOR TO THE TARGET AT TIME TAU (TIME WHEN THE SOUND *
               C     *     ORIGINATED.)  RXYTAU IS USED TO CONSTRUCT A W.           *
               C     *                                                              *
               C     ****************************************************************
               C
00018                D=AT(NT)*Y0(NS)-BT(NT)*X0(NS)
00019                DENOM=AT(NT)**2+BT(NT)**2
00020                XP=-(AT(NT)*CT(NT)+BT(NT)*D)/DENOM
00021                YP=(AT(NT)*D-BT(NT)*CT(NT))/DENOM
00022                RXYTAU=(SQRT((X0(NS)-XP)**2+(Y0(NS)-YP)**2))/
00022              1   ABS(SIN(BETA))
00023                RATIO=RATIO*RXYTAU/SQRT(ZA(NT)**2+RXYTAU**2)
00024                IF (RXYTAU .GT. 5000.0) RXYTAU=5000.0
00026                W=1000.0*W*EXP(-(RXYTAU/1000.0)**2)
               C
               C
```



COPY AVAILABLE TO DDC DOES NOT
PERMIT FULLY LEGIBLE PRODUCTION

```
NOL FORTRAN (1.0)                    SUBROUTINE ENTER (NS,NT,NR,T,XT,YT,ALPHA,X)

C     ****************************************************************
C     *                                                              *
C     *    THETA IS THE CORRECTION ANGLE DUE TO THE SPEED OF SOUND.  *
C     *    CALPHA IS THE CORRECTED  REPORT BEARING.                  *
C     *    ALPHA  IS CORRECTED AND PASSED BACK TO THE BIAS ALGORITHM *
C     *                                                              *
C     *    CALPHA IS USED TO CALCULATE A,B,C OF THE LINE A*X+B*Y+C=0 *
C     *    THIS LINE GOES THROUGH THE SENSOR AND THE TARGET          *
C     *    THE ALGORITHM WILL EVENTUALLY (INSPL)  FIND THE POINT     *
C     *    (A0,Y0)  WHICH MINIMIZES THE DISTANCE FROM THAT POINT     *
C     *    TO EACH LINE. A WEIGHTING FACTOR IS USED IN THE LEAST     *
C     *    SQUARES ALGORITHM.                                        *
C     *                                                              *
C     ****************************************************************
C
00027      500  THETA=ATAN(SIN(BETA)/(RATIO-COS(BETA)))
00028           CALPHA=(ALPHA-ALROT(NS))-THETA
00029           ALPHA=ALPHA-THETA
C
C     ****************************************************************
C     *                                                              *
C     *    CALCULATE A,B,C TO FORM THE LINE  A*X+B*Y+C=0             *
C     *                                                              *
C     ****************************************************************
C
00030           A=COS(CALPHA)
00031           B=-SIN(CALPHA)
00032           C=-(A*XT+B*YT)
C
C     ****************************************************************
C     *                                                              *
C     *    MAKE THE SUMS.                                            *
C     *                                                              *
C     ****************************************************************
C
00033           A2PB2=(A**2+B**2)*RXYT&U**2
00034           AI2=AI2+W*A**2/A2PB2
00035           BI2=BI2+W*B**2/A2PB2
00036           AIBI=AIBI+W*A*B/A2PB2
00037           AICI=AICI+W*A*C/A2PB2
00038           BICI=BICI+W*B*C/A2PB2
00039           RETURN
00040           END

          NOL FORTRAN DIAGNOSTIC RESULTS - FOR   ENTER

NO ERRORS
```



COPY AVAILABLE TO DDC DOES NOT
PERMIT FULLY LEGIBLE PRODUCTION

NOL FORTRAN (1.0) SUBROUTINE ENTER (NS,AT,NR,T,XT,YT,ALPHA,V)

DATA VARIABLES

| | | | | | |
|---|---|---|---|---|---|
| AI2 | 00005 | 00034 | 00034 | | |
| AIBI | 00005 | 00036 | 00036 | | |
| AICI | 00005 | 00037 | 00037 | | |
| ALA | 00001 | 00013 | | | |
| ALROT | 00002 | 00013 | 00028 | | |
| ASAVE | 00003 | 00008 | | | |
| AT | 00001 | 00018 | 00019 | 00020 | 00021 |
| BI2 | 00005 | 00035 | 00035 | | |
| BICI | 00005 | 00038 | 00038 | | |
| BT | 00001 | 00018 | 00019 | 00020 | 00021 |
| CS | 00001 | 00014 | | | |
| CT | 00001 | 00020 | 00021 | | |
| IFLAG | 00002 | 00011 | | | |
| INCSAVE | 00003 | | | | |
| JFLAG | 00002 | 00016 | | | |
| KFLAG | 00006 | 00011 | | | |
| M | 00006 | | | | |
| NPASS | 00006 | | | | |
| NREP | 00003 | | | | |
| NSEN | 00002 | | | | |
| NTRG | 00001 | | | | |
| PI | 00004 | 00013 | | | |
| PI2 | 00004 | | | | |
| PIO2 | 00004 | | | | |
| TA | 00001 | | | | |
| TEMP | 00001 | | | | |
| TSAVE | 00003 | 00007 | | | |
| VA | 00001 | 00014 | | | |
| WSAVE | 00003 | 00009 | | | |
| X0 | 00002 | 00018 | 00022 | | |
| XA | 00001 | | | | |
| Y0 | 00002 | 00018 | 00022 | | |
| YA | 00001 | | | | |
| ZA | 00001 | 00023 | | | |

COMMON VARIABLES

**NONE**

PROGRAM VARIABLES

| | | | | | |
|---|---|---|---|---|---|
| A | 00030 | 00032 | 00033 | 00034 | 00036 |
| A2PB2 | 00033 | 00034 | 00035 | 00036 | 00037 |
| ABS | 00022 | | | | |
| ALPHA | 00000 | 00008 | 00013 | 00028 | 00029 |
| ATAN | 00027 | | | | |
| B | 00031 | 00032 | 00033 | 00035 | 00036 |
| BETA | 00013 | 00022 | 00027 | 00027 | |
| C | 00032 | 00037 | 00038 | | |
| CALPHA | 00028 | 00030 | 00031 | | |



COPY AVAILABLE TO DDC DOES NOT PERMIT FULLY LEGIBLE PRODUCTION

| | | | | | |
|---|---|---|---|---|---|
| COS | 00027 | 00030 | | | |
| D | 00018 | 00020 | 00021 | | |
| DENOM | 00019 | 00020 | 00021 | | |
| EXP | 00026 | | | | |
| NR | 00000 | 00007 | 00008 | 00009 | |
| NS | 00000 | 00007 | 00008 | 00009 | 00013 |
| | 00028 | | | | |
| NT | 00000 | 00007 | 00008 | 00009 | 00010 |
| | 00019 | 00019 | 00020 | 00020 | 00020 |
| RATIO | 00014 | 00023 | 00023 | 00027 | |
| RXYTAU | 00015 | 00022 | 00023 | 00023 | 00024 |
| SIN | 00022 | 00027 | 00031 | | |
| SQRT | 00022 | 00023 | | | |
| T | 00000 | 00007 | 00010 | | |
| THETA | 00027 | 00028 | 00029 | | |
| TXY | 00010 | | | | |
| W | 00000 | 00009 | 00026 | 00026 | 00034 |
| XP | 00020 | 00022 | | | |
| XT | 00000 | 00010 | 00032 | | |
| YP | 00021 | 00022 | | | |
| YT | 00000 | 00010 | 00032 | | |



COPY AVAILABLE TO DDC DOES NOT PERMIT FULLY LEGIBLE PRODUCTION

SHIFT DECK/ L.R.X



COPY AVAILABLE TO DDC DOES NOT
PERMIT FULLY LEGIBLE PRODUCTION

```
IOL FORTRAN (1.0)

    00000               SUBROUTINE SHIFT (ANGLE,BIAS)
                C
                C   ***********************************************************
                C   *                                                         *
                C   *    JONATHAN VALVANO                          03/16/75   *
                C   *                                                         *
                C   *                                                         *
                C   *    THIS ROUTINE SHIFTS THE ANGLE BETWEEN                *
                C   *    (BIAS-PI) TO (BIAS+PI)                               *
                C   *                                                         *
                C   *                                                         *
                C   ***********************************************************
                C
    00001               COMMON /CONST// PI,PI02,PI2
                C
    00002        800    IF (BIAS-PI .LT. ANGLE) GO TO 810
    00004               ANGLE=ANGLE+PI2
    00005               GO TO 800
    00006        810    IF (BIAS+PI .GT. ANGLE) RETURN
    00008               ANGLE=ANGLE-PI2
    00009               GO TO 810
    00010               END

               NOL FORTRAN DIAGNOSTIC RESULTS - FOR   SHIFT

IO ERRORS
```



COPY AVAILABLE TO DDC DOES NOT
PERMIT FULLY LEGIBLE PRODUCTION

NOL FORTRAN (1.0)  SUBROUTINE SHIFT (ANGLE,BIAS)

DATA VARIABLES

| | | | | | |
|---|---|---|---|---|---|
| PI | 00001 | 00002 | 00006 | | |
| PI2 | 00001 | 00004 | 00008 | | |
| PID2 | 00001 | | | | |

COMMON VARIABLES

**NONE**

PROGRAM VARIABLES

| | | | | | |
|---|---|---|---|---|---|
| ANGLE | 00000 | 00002 | 00004 | 00004 | 00006 |
| BIAS | 00000 | 00002 | 00006 | | |



COPY AVAILABLE TO DDC DOES NOT PERMIT FULLY LEGIBLE PRODUCTION

```
TXY      DECK/    L.R.X
```



COPY AVAILABLE TO DDC DOES NOT
PERMIT FULLY LEGIBLE PRODUCTION

```
NOL FORTRAN (1.0)

00000                SUBROUTINE TXY (X,Y,T,NT)
                C
                C
                C    ********************************************************************
                C    *                                                                  *
                C    *    JONATHAN VALVANO                                  03/18/75    *
                C    *                                                                  *
                C    *                                                                  *
                C         THIS ROUTINE CALCULATES THE TARGET POSITION.
                C         THE POSITION  IS THOUGHT TO BE TRUE.
                C    *                                                                  *
                C    *                                                                  *
                C    ********************************************************************
                C
00001                COMMON /TRGT// ALA(3),XA(3),YA(3),ZA(3),
00001               1    TA(3),VA(3),AT(3),RT(3),CT(3),TEMP,CS,NTRG
00002                COMMON /CONST// PI,PID2,PI2
00003                ANGLE=PID2-ALA(NT)
00004                DIST=VA(NT)*(T-TA(NT))
00005                X=XA(NT)+DIST*COS(ANGLE)
00006                Y=YA(NT)+DIST*SIN(ANGLE)
00007                RETURN
00008                END

      \     NOL FORTRAN DIAGNOSTIC RESULTS - FOR   TXY

NO ERRORS
```



COPY AVAILABLE TO DDC DOES NOT
PERMIT FULLY LEGIBLE PRODUCTION

NOL FORTRAN (1.0) SUBROUTINE TXY (X,Y,T,NT)

DATA VARIABLES

| | | |
|---|---|---|
| ALA | 00001 | 00003 |
| AT | 00001 | |
| BT | 00001 | |
| CS | 00001 | |
| CT | 00001 | |
| NTRG | 00001 | |
| PI | 00002 | |
| PI2 | 00002 | |
| PID2 | 00002 | 00003 |
| TA | 00001 | 00004 |
| TEMP | 00001 | |
| VA | 00001 | 00004 |
| XA | 00001 | 00005 |
| YA | 00001 | 00006 |
| ZA | 00001 | |

COMMON VARIABLES

**NONE**

PROGRAM VARIABLES

| | | | | | |
|---|---|---|---|---|---|
| ANGLE | 00003 | 00005 | 00006 | | |
| COS | 00005 | | | | |
| DIST | 00004 | 00005 | 00006 | | |
| NT | 00000 | 00003 | 00004 | 00004 | 00005 |
| SIN | 00006 | | | | |
| T | 00000 | 00004 | | | |
| X | 00000 | 00005 | | | |
| Y | 00000 | 00006 | | | |



COPY AVAILABLE TO DDC DOES NOT PERMIT FULLY LEGIBLE PRODUCTION

TXY DECK/ L•R•X
NORMAL DECK/ L•R•X

000



COPY AVAILABLE TO DDC DOES NOT
PERMIT FULLY LEGIBLE PRODUCTION

```
L FORTRAN (1.0)

 00000            SUBROUTINE NORMAL(X)
                  C
                  C FORTRAN CALL LINE:
                  C
                  C     CALL NORMAL(X)
                  C
                  C THIS PROGRAM USES THE POLAR METHOD TO GENERATE NORMALLY DISTRIBUTED
                  C RANDOM NUMBERS OF ZERO MEAN AND UNIT VARIANCE.  THE NUMBERS ARE
                  C GENERATED TWO AT A TIME, SO EVERY OTHER CALL REQUIRES NO WORK.
                  C THE LOCAL INTEGER MODE VARIABLE M WILL NORMALLY NOT BE ONE AT LOAD
                  C TIME.  IF THIS IS NOT THE CASE FOR A PARTICULAR FORTRAN SYSTEM, THE
                  C USER CAN INITIALIZE THE PROGRAM VARIABLES BY MAKING AN INITIAL CALL
                  C AND DISREGARDING THE RESULT.
                  C
                  C THE SUBROUTINE UNIFORM USED IN THE PROGRAM SHOULD GENERATE RANDOM
                  C NUMBERS UNIFORMLY DISTRIBUTED BETWEEN ZERO AND ONE.
                  C
 00001                IF(M .EQ. 1) GO TO 20
 00003                M=1
 00004             10 CALL UNIFORM(Y)
 00005                U1=2.*Y-1.
 00006                CALL UNIFORM(Y)
 00007                U2=2.*Y-1.
 00008                S=U1*U1+U2*U2
 00009                IF(S .GE. 1.) GO TO 10
 00011                S=SQRTF(-2.*ALOG(S)/S)
 00012                X=U1*S
 00013                RETURN
 00014             20 M=0
 00015                X=U2*S
 00016                RETURN
 00017                END

               NOL FORTRAN DIAGNOSTIC RESULTS - FOR   NORMAL


ERRORS
```



COPY AVAILABLE TO DDC DOES NOT
PERMIT FULLY LEGIBLE PRODUCTION

NOL FORTRAN (1.0)  SUBROUTINE NORMAL(X)

DATA VARIABLES

**NONE**

COMMON VARIABLES

**NONE**

PROGRAM VARIABLES

| | | | | | |
|---|---|---|---|---|---|
| ALOG | 00011 | | | | |
| M | 00001 | 00003 | 00014 | | |
| S | 00008 | 00009 | 00011 | 00011 | 00011 |
| SQRTF | 00011 | | | | |
| U1 | 00005 | 00002 | 00008 | 00012 | |
| U2 | 00007 | 00000 | 00006 | 00015 | |
| UNIFORM | 00004 | 00000 | | | |
| X | 00000 | 00012 | 00015 | | |
| Y | 00004 | 00005 | 00006 | .00007 | |



COPY AVAILABLE TO DDC DOES NOT PERMIT FULLY LEGIBLE PRODUCTION

COPY AVAILABLE TO DDC DOES NOT
PERMIT FULLY LEGIBLE PRODUCTION

```
NOL COMPASS (1.0)
  UNIFORM  DECK/      L.X
```



COPY AVAILABLE TO DDC DOES NOT
PERMIT FULLY LEGIBLE PRODUCTION

NOL COMPASS (1.0)  UNIFORM

ENTRY POINT SYMBOLS
UNIFORM  00000

PROGRAM  00052
COMMON
DATA



COPY AVAILABLE TO DDC DOES NOT
PERMIT FULLY LEGIBLE PRODUCTION

```
                ENTRY     UNIFORM
```

FORTRAN CALL LINE:

```
                CALL UNIFORM(X)
```

IF X IS GREATER THAN OR EQUAL TO ZERO WHEN THE CALL IS EXECUTED, X IS REPLACED WITH A RANDOM NUMBER UNIFORMLY DISTRIBUTED BETWEEN 0 AND 1. THE COMPUTATION USES 35 BIT INTEGER ARITHMETIC, AND CONVERTS TO FLOATING POINT. THE ALGORITHM IS THE LINEAR CONGRUENTIAL METHOD WITH A MODULUS OF 2**35, A MULTIPLIER OF 5**15, AND AN INCREMENT OF 7261067065. THESE PARAMETERS HAVE BEEN EXTENSIVELY TESTED, AND YIELD A MAXIMUM LENGTH SEQUENCE WITH GOOD SPECTRAL PROPERTIES.

IF X IS LESS THAN ZERO WHEN THE CALL IS EXECUTED, -X IS CONVERTED TO A 35 BIT FIXED POINT FRACTION, AND THE MOST SIGNIFICANT 35 BITS ARE USED TO INITIALIZE THE RANDOM NUMBER GENERATOR. IF -X IS GREATER THAN OR EQUAL TO 1 OR LESS THAN 2**-35, THE SEQUENCE WILL START WITH ZERO. THE STARTING VALUE IS CONVERTED TO FLOATING POINT AND RETURNED IN X, SO THAT X IS ALWAYS A FRACTION BETWEEN 0 AND 1 ON RETURN.

ENTER, GET X, AND BRANCH TO INITIALIZATION IF NEGATIVE.

```
 77777 0  UNIFORM  UJP       **       ENTRY POINT
P00000 1          LDI       UNIFORM,1 PARAMETER ADDRESS TO B1
 00000 1          LDAQ,I    0,1      X TO AQ
P00024 3          AZJ,LT    INIT     INITIALIZE IF X LT ZERO
```

IF X GE ZERO ON CALL, GENERATE A NEW RANDOM NUMBER.

```
P00042 0          LDAQ      Y        NEWY=(A*OLDY+2*C)MOD2**36
P00044 0          MUAQ      A
 00000 3          EAQ
 07777 2          ANA       7777B
P00046 0          ADAQ      C2
 07777 2          ANA       7777B
P00042 0  CONT    STAQ      Y        SAVE Y FOR NEXT TIME
```

CONVERT THE 36 BIT NUMBER IN AQ TO A FLOATING POINT FRACTION.

```
 02013 2          SCAQ      2013B,2  SCALE AQ,BIASED EXPONENT TO B2
P00022 0          AZJ,EQ    RETURN   ZERO IS A SPECIAL CASE
 02000 2          ISG       2000B,2  TEST FOR NON-NEGATIVE EXPONENT
 77776 2          INI       -1,2     ADJUST BIAS IF NEGATIVE
 00031 0          SHAQ      25       MERGE FRACTION AND EXPONENT
 40000 2          AIA       2
 00014 0          SHAQ      12       AQ IS NOW A FLOATING POINT FRACTION
```

STORE AQ IN X, AND RETURN TO CALLING PROGRAM.

```
 00000 1  RETURN   STAQ,I    0,1      STORE AQ IN X
```



COPY AVAILABLE TO DDC DOES NOT
PERMIT FULLY LEGIBLE PRODUCTION

```
00001 1           UJP       1,1       RETURN TO PARAMETER ADDRESS + 1

                  IF X LT ZERO ON CALL, CONVERT -X TO FIXED POINT AND USE IT TO
                  INITIALIZE THE RANDOM NUMBER GENERATOR.

77777 0  INIT     XOA,S     77777B    NEGATE AQ
77777 1           XOQ,S     77777B
00044 0           SHAQ      36        BIASED EXPONENT TO 82
00000 2           TAI       2
03777 2           ANI       3777B,2
00014 0           SHAQ      12        FRACTION TO AQ
07777 2           ANA       77777B
02001 2           ISG       20018,2   SKIP IF EXPONENT GE 1
01735 2           ISG       1735B,2   SKIP IF EXPONENT GE -34
P00051 0          LDAQ      ZERO      FORCE ZERO IF EXPONENT OUT OF BOUNDS
01777 2           ISG       17778,2   SKIP IF EXPONENT NON-NEGATIVE
76000 3           SHAQ      -1777B,2  EXPONENT IS SHIFT COUNT, IF NEGATIVE
77776 1           ANQ,S     7777OB    CLEAR LSB TO RETAIN 35 BITS
P00012 0          UJP       CONT      USE AQ FOR NEW Y IN MAIN LOOP

                  CONSTANTS AND VARIBLES

000               Y         OCT       0,0
000
432               A         OCT       3432,7724461B   THIS IS 5**15
215
541               C2        OCT       1541,45427232   THIS IS 2*72610670B5
232
000               ZERO      OCT       0,0
000

                            END

                  NUMBER OF LINES WITH DIAGNOSTICS        0
```

COPY AVAILABLE TO DDC DOES NOT PERMIT FULLY LEGIBLE PRODUCTION

## DISTRIBUTION

| | Copies |
|---|---|
| Defense Documentation Center<br>Cameron Station<br>Alexandria, VA 22314 | 12 |
| MITRE Corporation<br>1820 Dolley Madison Blvd<br>McLean, VA 22101<br>Attn: C. L. Woodbridge | |
| ADTC/AFATL<br>Eglin AFB, FL 32542<br>Attn: J. G. Constantine (DLJM) | |
| Army Mobility Equipment Research & Development Command<br>Fort Belvoir, VA 22060<br>Attn: Library | |
| Army Engineer Waterways Experiment Station<br>P.O. Box 631<br>Vicksburg, MS 39180<br>Attn: Dr. Lundien | |
| Army Electronics Command AMSEL-CT-I<br>Fort Monmouth, NJ 07703<br>Attn: A. Zarin | |
| Naval Sea Systems Command<br>Washington, D. C. 20360<br>Attn: SEA-03B | |

NSWC(W)-5605/1 (Rev. 1-75)

TO AID IN UPDATING THE DISTRIBUTION LIST FOR NAVAL SURFACE WEAPONS CENTER, WHITE OAK LABORATORY TECHNICAL REPORTS PLEASE COMPLETE THE FORM BELOW:

TO ALL HOLDERS OF NSWC/WOL/TR 76-8
by Joseph Reid Code WA-22
DO NOT RETURN THIS FORM IF ALL INFORMATION IS CURRENT

A. FACILITY NAME AND ADDRESS *(OLD) (Show Zip Code)*

NEW ADDRESS *(Show Zip Code)*

B. ATTENTION LINE ADDRESSES:

C.

☐ REMOVE THIS FACILITY FROM THE DISTRIBUTION LIST FOR TECHNICAL REPORTS ON THIS SUBJECT.

D.
NUMBER OF COPIES DESIRED _____

GPO 886-970