MONTE-CARLO EVALUATION OF DIGITAL FILTERS FOR FIRE CONTROL SYSTEMS

Michael Gordon Ketron

NAVAL POSTGRADUATE SCHOOL

Monterey, California

THESIS

MONTE-CARLO EVALUATION OF DIGITAL FILTERS
FOR FIRE CONTROL SYSTEMS

by

Michael Gordon Ketron

June 1974

Thesis Advisor: D. E. Kirk

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) MONTE-CARLO EVALUATION OF DIGITAL FILTERS FOR FIRE CONTROL SYSTEMS		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis June 1974
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Michael Gordon Ketron		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		12. REPORT DATE June 1974
		13. NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) Naval Postgraduate School Monterey, California		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)

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

Techniques are discussed for Monte-Carlo evaluation of several filters in a digital fire control system performing against an aircraft trajectory developed to simulate a diving attack. The programs developed are designed to allow for future modeling work in a three-dimensional theater of operations with expedient selection of the

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE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

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

options that are discussed, and a comprehensive user's guide is provided. Comparison is made between a model of the filter currently used in the MK-86 Digital Fire Control System and some promising combinations of the several options available.

Monte-Carlo Evaluation of Digital Filters
for Fire Control Systems

by

Michael Gordon Ketron
Lieutenant (junior grade), United States Navy
B.S., University of Tennessee, 1971

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the
NAVAL POSTGRADUATE SCHOOL
June 1974

ABSTRACT

Techniques are discussed for Monte-Carlo evaluation of several filters in a digital fire control system performing against an aircraft trajectory developed to simulate a diving attack. The programs developed are designed to allow for future modeling work in a three-dimensional theater of operations with expedient selection of the options that are discussed, and a comprehensive user's guide is provided. Comparison is made between a model of the filter currently used in the MK-86 Digital Fire Control System and some promising combinations of the several options available.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure

I. INTRODUCTION

With the increasing cost and complexity of modern gunfire control systems there exists a correspondingly increasing need for thorough evaluation of simulated performance with accurate models before major commitment is made.

Specifically, the MK-86 Digital Gunfire Control System has come under such scrutiny and this thesis is one in a series designed to define, evaluate and offer alternative approaches to the existing system. While the original motivation for the study in this thesis was to explore alternative techniques to those used in the existing system, the resulting derivations and computer programs lend themselves to other system modeling situations within certain restrictions. The development of this thesis is intended to allow future modeling work to easily take advantage of the results contained herein.

The pattern followed in a typical cycle of the gunfire control system is the acquisition of data by a radar, analysis of the measurements in a filtering section, and the generation of gun orders in a third general section. The entire model should study all of these segments, but in order to simplify the problem each segment may be studied separately. This thesis is designed for studying the filtering section and assumes that input data is available from a

typical radar and that the results would be used by another section to generate the gun orders.

In order to evaluate possible filtering methods for analyzing the radar data it was necessary to generate realistic, simulated radar measurements for the model. Therefore, a hypothetical track was developed with the capacity for variation to provide the necessary input data. In addition, the various special features that are presented in this thesis were organized in the computer simulation for straightforward selection and implementation thus offering maximum flexibility for evaluation of the various filtering methods. This designed-in choice of options was intended, also, to lend the general program to other applicable situations.

II. PROBLEM DESCRIPTION

A. KALMAN FILTER THEORY

Sequential estimation is characterized by the serial recursive processing of observations taken in time sequence. The result of every processing cycle is the current best estimate of the vector being estimated. This estimate therefore embodies all observation data up to and including the current observation. As a new observation is made, the current estimate is updated to reflect this most recent data. In such an estimation scheme the calculations are identical in nature from cycle to cycle so they are ideally suited for implementation on a digital computer.

The Kalman filter [1] is a recursive filter of the type applicable to a digital fire control system in which discrete observations are available from the radar. The filter offers the capability of not only generating estimates of the observed system's states but, also, of predicting future system (or plant) states.

The linear discrete system for which the Kalman filter is designed is characterized by the state and output equations

$$\underline{x}(k+1) = \underline{\phi}\ x(k) + \underline{\Delta}\ \underline{u}(k) + \underline{\Gamma}\ \underline{w}(k) \quad (1)$$

$$\underline{z}(k) = \underline{H}\ \underline{x}(k) + \underline{v}(k) \quad (2)$$

where

$\underline{x}(k)$ is the n-dimensional state vector at time $t = kT$,

$\underline{u}(k)$ is the p-dimensional deterministic input vector at time $t = kT$,

$\underline{z}(k)$ is the m-dimensional vector of measurements or observations taken at time t = kT,

$\underline{w}(k)$ and $\underline{v}(k)$ are q-dimensional and m-dimensional noise processes, respectively, at time t = kT,

$\underline{\phi}$ is the n x n state transition matrix--assumed to be known,

$\underline{H}$ is the m x n observation matrix--assumed to be known,

$\underline{\Lambda}$ and $\underline{\Gamma}$ are n x p and n x q matrices, respectively, which relate the deterministic and non-deterministic forcing terms to the state vector --they are assumed to be known,

T is the time between measurements, and k is a non-negative integer.

The noise statistics are summarized below

$$E[\underline{v}(k)\ \underline{v}^T(j)] = \underline{R}(k)\ \underline{\delta}(k,j) \tag{3}$$

$$\underline{\Gamma}\ E[\underline{w}(k)\ \underline{w}^T(j)]\underline{\Gamma}^T = \underline{Q}(k)\ \underline{\delta}(k,j) \tag{4}$$

$$E[\underline{v}(k)\ \underline{w}^T(j)] = 0 \text{ for all } (k,j) \tag{5}$$

$$\underline{\delta}(k,j) = \begin{cases} 0 & k \neq j \\ 1 & k = j \end{cases} \tag{6}$$

It is assumed that the initial state is a random variable with known mean and covariance

$$E[\underline{x}(0)] = \hat{\underline{x}}_o,\ E\left\{[\underline{x}(0) - \hat{\underline{x}}_o][\underline{x}(0) - \hat{\underline{x}}_o]^T\right\} = \underline{P}_o \tag{7}$$

In addition, it is assumed that the measurement noise and initial state are uncorrelated

$$E[\underline{x}(0)\ \underline{v}^T(k)] = 0 \text{ for all } k \tag{8}$$

and that the forcing noise and initial state are uncorrelated

$$E[\underline{x}(0)\ \underline{w}^T(k)] = 0 \text{ for all } k \tag{9}$$

The Kalman filter equations are summarized below

$$\underline{G}(k) = \underline{P}(k/k-1)\ \underline{H}^T[\underline{H}\ \underline{P}(k/k-1)\ \underline{H}^T + \underline{R}(k)]^{-1} \tag{10}$$

$$\underline{P}(k/k) = (\underline{I} - \underline{G}(k)\ \underline{H})\ \underline{P}(k/k-1) \tag{11}$$

$$\underline{P}(k+1/k) = \underline{\phi}\ P(k/k)\ \underline{\phi}^T + \underline{Q}(k) \tag{12}$$

$$\hat{\underline{x}}(k/k) = \hat{\underline{x}}(k/k-1) + \underline{G}(k)[\underline{z}(k) - \underline{H}\ \hat{\underline{x}}(k/k-1)] \tag{13}$$

$$\hat{\underline{x}}(k+1/k) = \underline{\phi}\ \hat{\underline{x}}(k/k) + \underline{\Delta}\ \underline{u}(k) \tag{14}$$

where the notation (k/k-1) is defined as a condition at time $t = kT$ given information up to and including time $t = (k-1)T$. The matrices in these equations are

$\underline{G}(k)$	-	n x m gain matrix
$\underline{R}(k)$	-	m x m covariance matrix of measurement error
$\underline{P}(k/k)$	-	n x n covariance matrix of estimation error
$\underline{I}$	-	n x n identity matrix
$\underline{P}(k+1/k)$	-	n x n prediction covariance matrix
$\underline{Q}(k)$	-	n x n covariance matrix of state excitation defined as $\underline{\Gamma}\ E[\underline{w}(k)\ \underline{w}^T(k)]\ \underline{\Gamma}^T$
$\hat{\underline{x}}(k/k)$	-	n x 1 optimal (minimum variance) estimates of x(k)
$\underline{z}(k)$	-	m x 1 observation vector
$\hat{\underline{x}}(k+1/k)$	-	n x 1 optimal predicted value of $\underline{x}(k)$.

A block diagram of the discrete plant and Kalman filter is shown in Figure 1.

The Kalman filter takes advantage of all previous state measurements, $\underline{z}$, along with their respective error estimates,

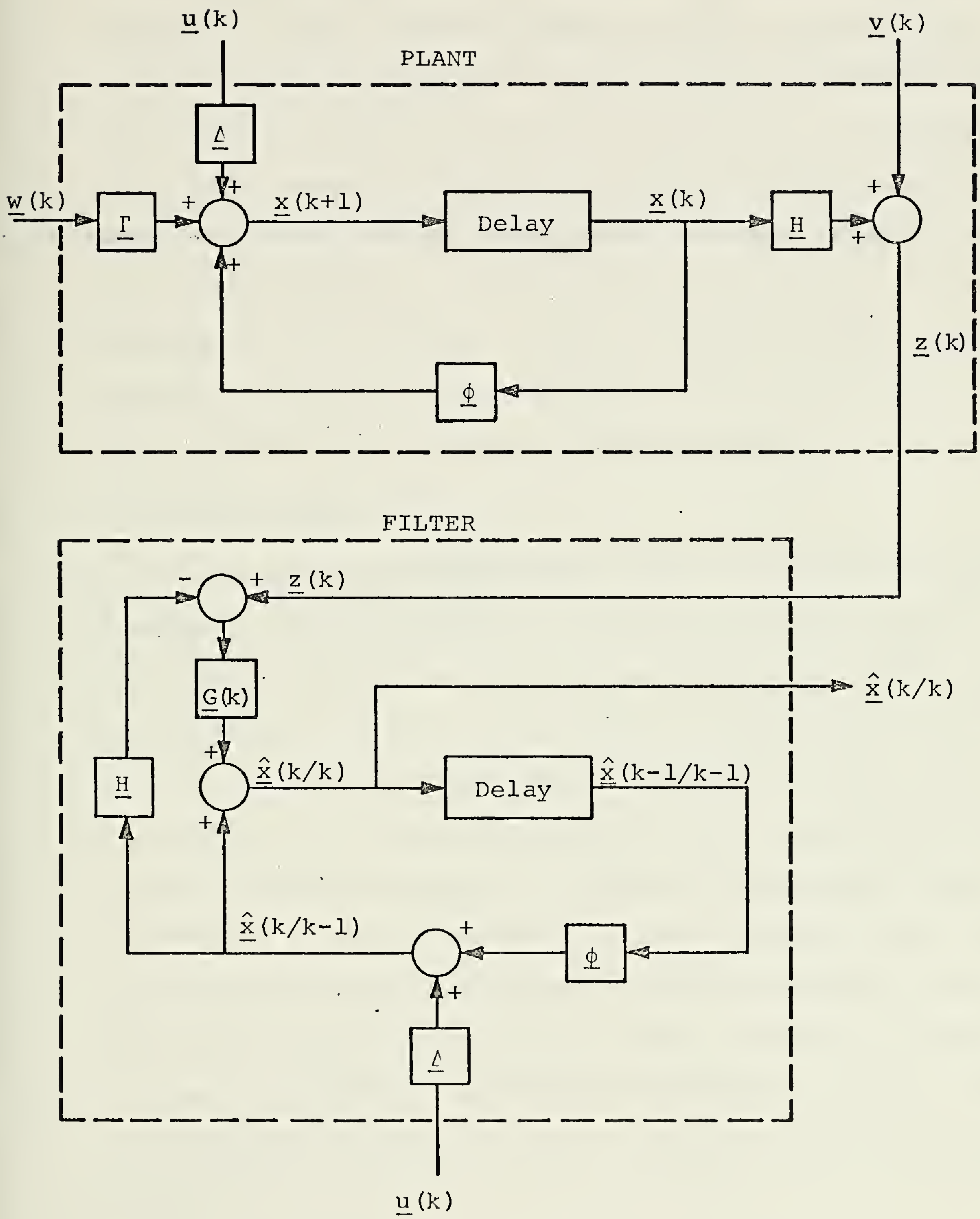


Figure 1. Block Diagram of the Discrete Plant and Kalman Filter.

and predicts ahead what the system states should be based on the state transition matrix, $\underline{\phi}$, and any known deterministic forcing $\underline{u}$. When a new measurement becomes available for current state estimation the filter takes the predicted state vector from the previous iteration, $\hat{\underline{x}}(k/k-1)$, and corrects it by some amount depending on the difference between the predicted vector and the measurement vector. Normally the amount of correction is a fraction (less than one) of the difference and is defined by the gain matrix, $\underline{G}(k)$, which has been calculated using equations (10) - (12) so that the state estimates yield minimum variance estimates.

B. SYSTEM MODELS

In order to apply the Kalman filter to a given situation a model for the plant must be specified. In the case of a fire control system the plant is defined as the target. The motion of the target is characterized by quantities such as position, velocity, acceleration, acceleration rate, etc.

For most applications the plant can be approximated as either a constant-velocity or constant-acceleration target. In terms of a three-dimensional tracking problem with filtering in an orthogonal coordinate system the resultant state vector for the constant-velocity model includes six states --three for position definition and three for velocity. The constant-acceleration model also includes three acceleration states.

In this thesis, for example, the filtering is done in an XYZ Cartesian coordinate system, shown in Figure 2, using

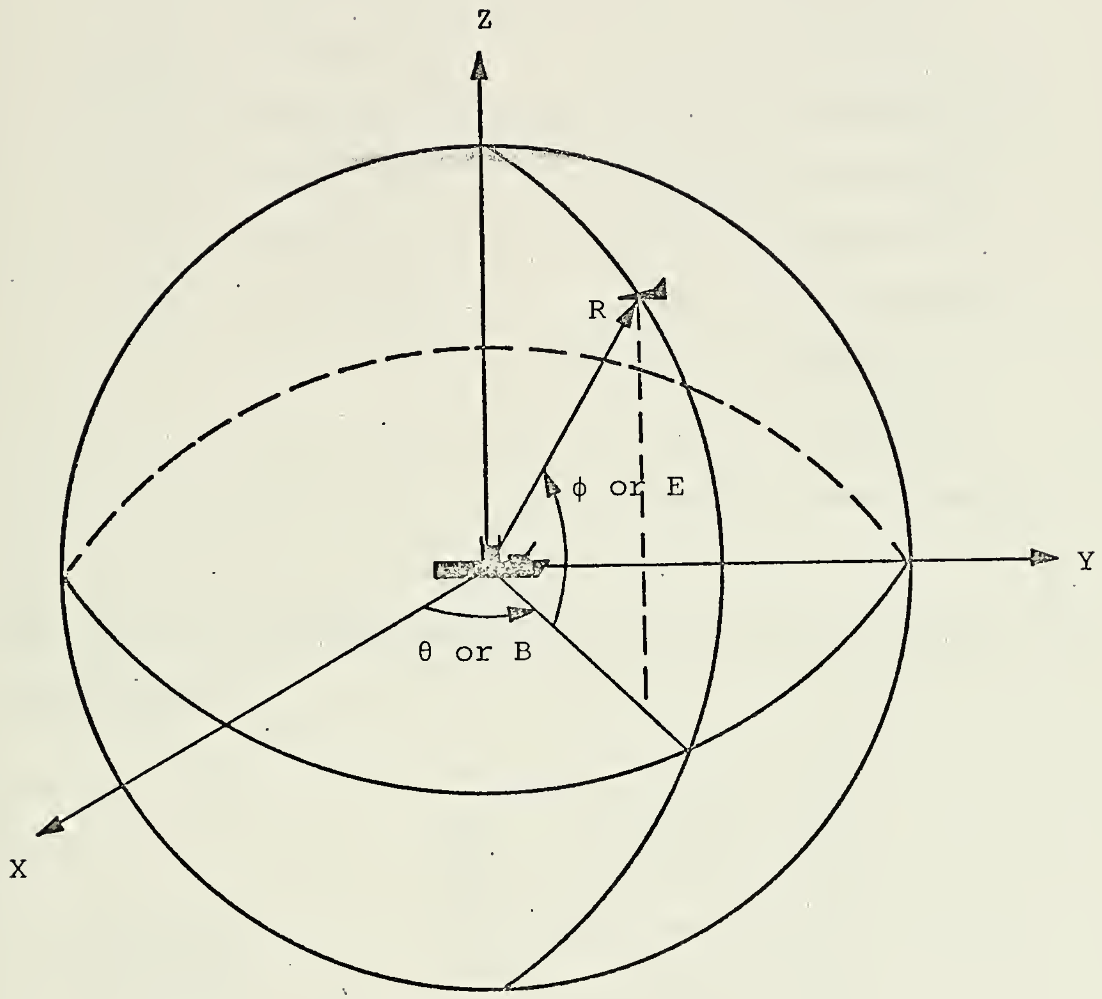


Figure 2. Coordinate System.

both constant-velocity and constant-acceleration models with the resultant state vectors

Constant-velocity Model		Constant-Acceleration Target	
x_1	x position	x_1	x position
x_2	x velocity	x_2	x velocity
x_3	y position	x_3	x acceleration
x_4	y velocity	x_4	y position
x_5	z position	x_5	y velocity
x_6	z velocity	x_6	y acceleration
		x_7	z position
		x_8	z velocity
		x_9	z acceleration

The state transition matrices, $\underline{\phi}$, for the two models are shown below

Constant-Velocity

$$\underline{\phi} = \begin{bmatrix} 1 & T & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & T & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & T \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \qquad (17)$$

$$\text{Constant Acceleration} \quad \underline{\phi} = \begin{bmatrix} 1 & T & \frac{T^2}{2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & T & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & T & \frac{T^2}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & T & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & T & \frac{T^2}{2} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & T \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{18}$$

For the models used in defining the plant in this thesis the deterministic and non-deterministic forcing terms, $\underline{\Delta}\ \underline{u}(k)$ and $\underline{\Gamma}\ \underline{w}(k)$, are assumed to be zero.

C. MONTE-CARLO SIMULATION

The basic premise upon which computer simulation is justified is that the model used for study accurately portrays the phenomenon as it occurs in practice. In order to assure that this requirement is satisfied repeated independent experiments yielding classical random-sample statistics are necessary. The requirement as it applies to actual simulation exercises implies that those parameters in the model that are variable require alteration in a random manner dictated by the expected randomness in practice.

Monte-Carlo evaluation is the process of satisfying the need for randomness in some of the model parameters and is especially applicable with computer simulation where the

necessary calculating capacity and speed is available. Further discussion of Monte-Carlo techniques and applications may be found in [2].

The following discussion presents the necessary information for deciding how many Monte Carlo runs, or iterations, are required to satisfy some desired statistical confidence in the simulation. The source of theoretical information against which empirical sampling of available random number generators are compared is [3].

The use of available random number generators, such as GAUSS or NORMAL for normally distributed random numbers, is the most straightforward method for obtaining the desired randomness to introduce into the simulation. Two factors that must be considered in evaluating the performance of a random number generator are (1) the statistics of the empirical samples compared with the desired statistics and (2) the confidence for which the results of (1) are guaranteed.

For the type of random numbers needed with the Monte-Carlo simulation of the filtering process in a fire control system the merit of acceptance is based on the actual mean and standard deviation (or variance) of the normally distributed random numbers compared to the desired, or specified, values of these quantities. Considering the mean of the distribution first, several variables can be defined that are used in calculating a desired number of samples to achieve an acceptable mean. These include

d - a margin of acceptable error of the sample mean expressed as a percentage of the desired standard deviation, such as 0.1 (10% of the standard deviation),

α - the risk involved in the result, also signifying (100 - α) per cent confidence in the result,

β - defined as (1 - α/2).

Entering Figure 3 with the value of β the variable t is defined. The number of samples necessary to give a sample mean within d of the desired mean with a known risk of α is given by

$$n = \frac{t^2 \sigma^2}{d^2} \tag{19}$$

It can be seen that if d is expressed as a percentage of σ, the actual value of the standard deviation is not required to solve for the number of samples needed.

To determine the number of samples necessary for desired statistics of the standard deviation refer to Figure 4. Here again, the margin of acceptable error is expressed as a percentage of the standard deviation, and this percentage is the parameter P scaled along the abscissa. The ordinate is labeled "degrees of freedom" and represents the number of samples needed to achieve the various confidences plotted as bars across the graph.

Working through an example shows how the curves are used. Suppose it is desired to execute enough Monte-Carlo runs to assure 90% confidence in sample statistics having an actual mean and an actual standard deviation within 10% of its desired value. Considering the mean first:

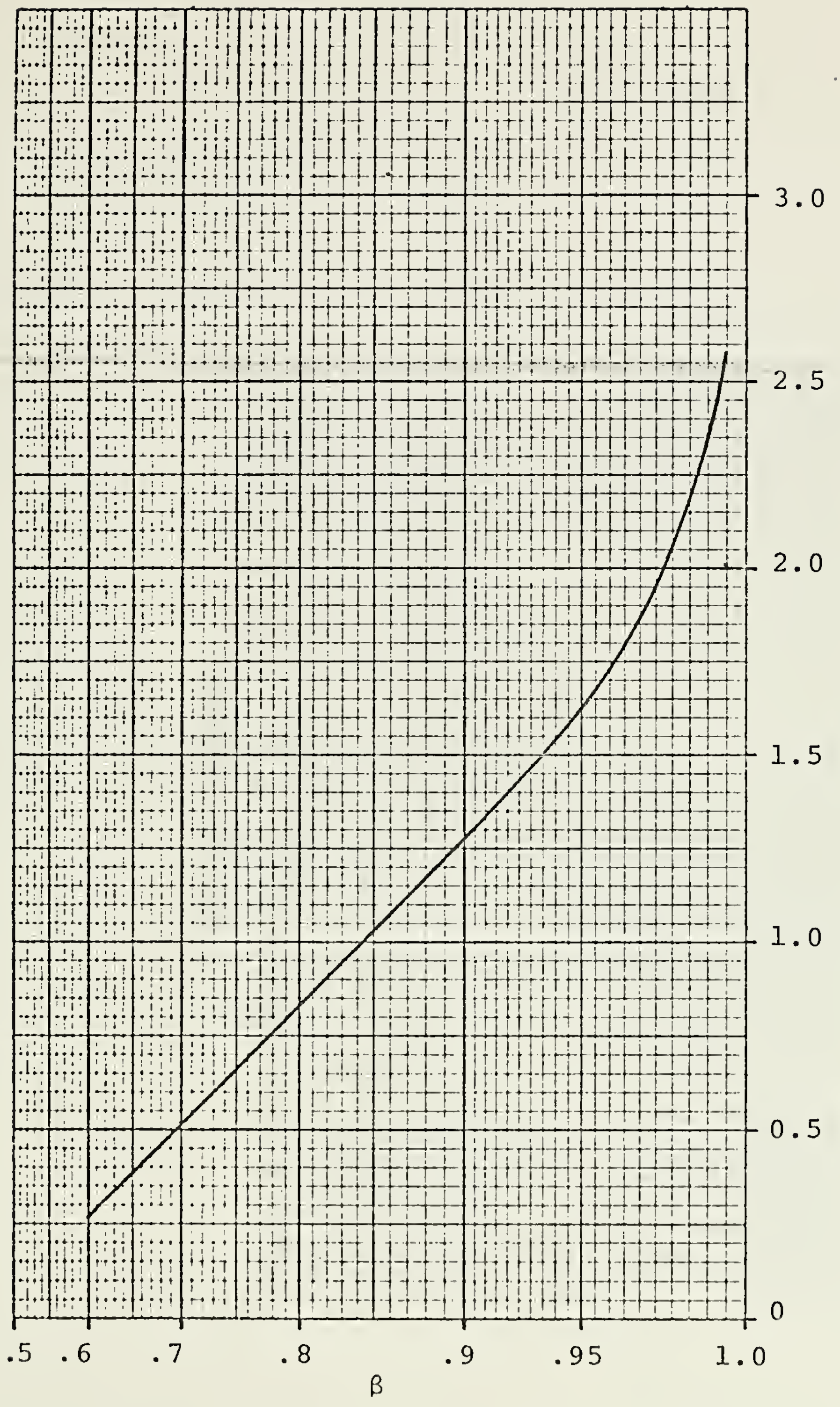


Figure 3. t - β Diagram.

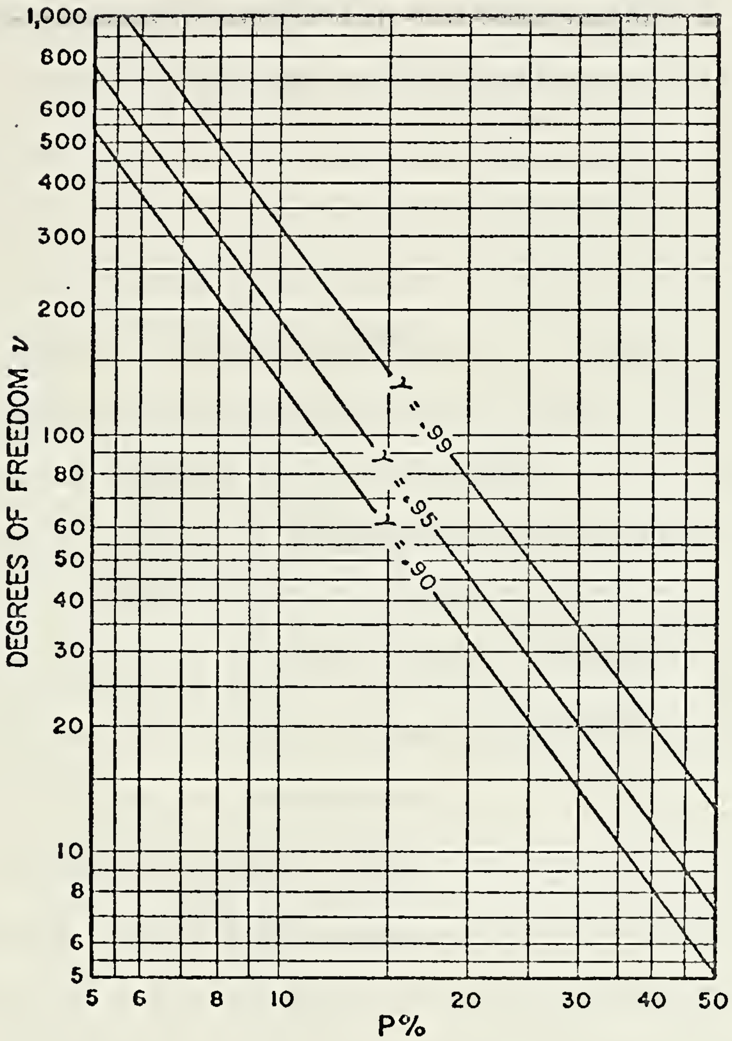


Figure 4. Number of Samples Required to Estimate the Standard Deviation within P Percent of Its True Value with Confidence Coefficient γ.

$$\beta = 1 - \frac{\alpha}{2} = 1 - \frac{.1}{2} = .95 \quad (20)$$

From Figure 3 the value of t is found to be 1.645, and

$$n = \frac{(1.645)^2\sigma^2}{(.1)^2\sigma^2} = 270 \quad (21)$$

Referring to Figure 4 to find the number of samples required for the desired standard deviation results gives n = 140. Therefore, in order to satisfy all restrictions 270 samples would be required.

Verification of these theoretical results involved sampling two subroutines available in the NPS Computer Facility Sublibrary, GAUSS and NORMAL. The approach used was to sample each number generator for 1000 ensembles of several numbers of samples in each ensemble. The number 1000 was chosen to obtain steady-state values in the statistics determined for the runs of length 50, 75, 100, 150, 200, 500, 700, 1000, and 5000 samples each. A summary of the results is included as Table I,[1] and the FORTRAN program for the study may be found in Appendix A.

It may be noted in Table I that the results for the normally distributed random number generator GAUSS are incomplete, and this is a consequence of the substantially greater execution time required than for NORMAL. Whereas NORMAL used

[1]Theoretical results are limited in accuracy to one's ability to read Figure 3. Also, percent of standard deviations within criteria of desired standard deviations are not shown for theoretical calculations because (1) Figure 4 is not complete and (2) the limiting factor in satisfactory M-C simulation is the satisfaction of requirements for the mean, not for the standard deviation.

TABLE I

COMPARISON OF EMPIRICAL AND THEORETICAL NORMALLY DISTRIBUTED RANDOM NUMBERS

Number of Samples	Percent Criteria	Theoretical $\underline{M}^a$	NORMAL $\underline{M}^a$	NORMAL $\underline{SD}^b$	GAUSS $\underline{M}^a$	GAUSS $\underline{SD}^b$
50	5	28.0	27.0	32.7	27.4	37.2
	10	52.4	51.9	63.7	52.4	64.8
	15	71.2	69.0	82.8	72.3	83.9
	20	84.2	83.9	94.3	85.7	94.5
	25	92.4	92.9	98.1	94.2	98.4
75	5	33.6	33.7	42.2	34.7	44.2
	10	62.0	60.4	75.1	60.6	76.6
	15	80.8	80.2	82.5	81.3	93.2
	20	91.6	91.9	98.5	92.0	99.0
	25	96.8	97.0	99.8	97.6	99.9
100	5	38.6	39.0	49.8	37.3	50.7
	10	68.0	67.9	83.2	66.9	83.3
	15	86.8	86.3	96.0	87.1	97.2
	20	95.4	95.1	99.2	95.7	99.6
	25	98.8	98.3	100.0	99.1	100.0
150	5	47.0	45.2	60.0	45.3	60.8
	10	78.0	78.3	89.6	78.2	92.1
	15	93.4	91.9	99.0	94.2	99.3
	20	98.4	98.0	99.9	99.0	100.0
	25	99.8	99.7	100.0	99.9	100.0
200	5	52.0	52.7	66.8	52.6	68.7
	10	84.2	82.2	95.3	83.6	95.2
	15	96.4	96.0	99.8	96.3	99.9
	20	99.6	99.4	100.0	99.1	100.0
	25	100.0	99.9	100.0	99.9	100.0
500	5	73.4	75.3	87.0	72.6	88.9
	10	97.0	96.7	99.9	97.0	99.9
	15	100.0	99.9	100.0	99.9	100.0
	20	100.0	100.0	100.0	100.0	100.0
700	5	81.4	81.0	93.5	81.5	93.7
	10	99.2	99.6	100.0	98.6	100.0
	15	100.0	100.0	100.0	100.0	100.0
1000	5	89.0	90.4	96.2	-	-
	10	100.0	99.9	100.0	-	-
	15	100.0	100.0	100.0	-	-
5000	5	100.0	100.0	100.0	-	-

Note (a) - Percent of Means within percent criteria of desired standard deviation

Note (b) - Percent of Standard Deviations within same percent criteria as (a)

approximately 25 minutes of execution time to carry out the entire analysis, GAUSS required 90 minutes to evaluate up through 700 sample ensemble runs. For this reason NORMAL is recommended when normally distributed random numbers are desired.

III. DEVELOPMENT OF MONTE-CARLO SIMULATION PROGRAM (MCSP)

A. TRACK MODEL

In order to provide realistic data for testing filter performance against an airborne target it was necessary to develop a model based on a conventional dive bomb approach by a modern aircraft against a seaborne target. From knowledgeable sources the parameters of such an approach were obtained and, in accordance with the 4-Hz sampling rate of the radar, used to develop a simulation including such features as a constant-acceleration dive, a pull-out following ordinance release, and evasive maneuvers while withdrawing in a gradual climb. The individual segments of the 233-point track will be discussed as well as the methods one may use to develop other tracks.

While in actual operation a radar measures range, bearing, and elevation (RBE), and perhaps rates as well, convenience dictated development of the track in an XYZ coordinate system centered at the ship, or radar,[2] as shown in Figure 2. Such parameters as ship's speed, bearing, roll, pitch, etc. were not considered in the simulation to avoid model complexity. These quantities, while offering potential sources of error to any overall tracking system, were avoided to concentrate on the filter performance and not the various correction factors applied to input measurement data.

---

[2]The radar and the gun are assumed to be at the center of the ship, or origin of coordinate system, and at sea level.

Satisfying the requirement for randomness in the track data points, the variable factors mentioned earlier for Monte-Carlo simulation, involved, when using one basic track, the addition of randomly distributed Gaussian measurement noise to every point. An alternative approach might have been to run the simulation with many independent tracks randomly distributed about the expected track, but this would have caused much additional complexity and increased execution time in the simulation.

In order to realistically generate noisy measurements the track's XYZ data points, after being read by the Monte-Carlo Simulation Program (MCSP), are converted to RBE, the desired Gaussian measurement noise with zero mean and appropriate standard deviation is added to each data point, conversion back into the same XYZ coordinate system takes place, and the data is available as simulated coordinate-converted output from the radar ready for analysis.

The original track is layed out with the aircraft approaching from the +X direction with initial radar contact initiated at a range of 7200 yards and altitude of 4333 yards. Its initial velocity is approximately 250 knots, and in the scenario of the attack has just climbed to the given altitude from a low or medium level approach in preparation for the dive phase of the attack. A summary of the track segments follows:

Track Points	Segment Description	G Turn	Comments
1 - 33	Initial Dive	1	Assumes 40° Dive Angle
34 - 113	Accelerating Dive	-	139.16 yd/sec-249.75 yd/sec
114 - 137	Pullout	4	To altitude of 958 yards
138 - 141	Starboard Turn	2	
142 - 145	Port Turn	2	
146 - 149	Port Turn	2	
150 - 157	Port Turn	3.5	Begin 10° Climb -
158 - 165	Starboard Turn	3.5	Velocity of 249.76
166 - 173	Port Turn	3.0	yd/sec is maintained
174 - 181	Starboard Turn	3.0	
182 - 193	Port Turn	3.0	
194 - 205	Starboard Turn	2.5	
206 - 221	Port Turn	2.5	
222 - 233	Starboard Turn	2.0	

Note that the port turn at points 142 - 145 is followed by another similar turn from points 146 - 149. The need for the additional segment rather than creating just one turn from points 142 - 149 will become clear later in the discussion.

The geometric relationships used to translate the desired track into mathematical terms include both basic, intuitive approaches and some requiring direct application to a given target orientation. These general techniques will now be discussed; the implementation of the particular track segments in a FORTRAN program may be found in Appendix B.

The following sub-sections portray the various possible maneuvers and track orientations used in developing the track for the MCSP.

A.1 The initial conditions include the starting point anywhere in the coordinate system with the velocity vector parallel to an axis. Specified parameters for a turn can include angle of turn, time of execution, tangential velocity,

constant radial acceleration, and final translation with respect to starting point. (Examples are: Points 1 - 33, 138-141, 146 - 149)

The geometrical arrangement is shown in Figure 5, and the following pertinent definitions and equations apply: a is the radial acceleration, vel is the tangential velocity, and T is the time of maneuver; thus

$$a = \frac{32.2}{3} \text{ yds/sec}^2 \cdot \text{G of turn} = \text{vel}^2/R \,, \tag{22}$$

$$C = (4h(2R-h))^{\frac{1}{2}}, \; S = R\theta = 2R\sin^{-1}C/2R,$$
$$T = S/2 \cdot \text{vel} \,. \tag{23}$$

Using the pertinent equations certain parameters of the turn can be specified while assuring that the other parameters are reasonable. This results in values for R, θ, and the number of track points in the segment and provides incrementally growing Δθ's. Then each new track point is given by

$$X_n = X_o \pm R \sin \Delta\theta \tag{24}$$

$$Y_n = Y_o \pm (R - R \cos \Delta\theta) \,. \tag{25}$$

Note: The optional signs above are dependent upon the orientation of the maneuver on the axes.

<u>A.2</u> The initial conditions include the starting point anywhere with the velocity vector at some angle, τ, with respect to an axis. The turn is in the plane of two of the axes with certain parameters known, such as those mentioned in the previous sub-section. Four types of turn are possible in this scenario:

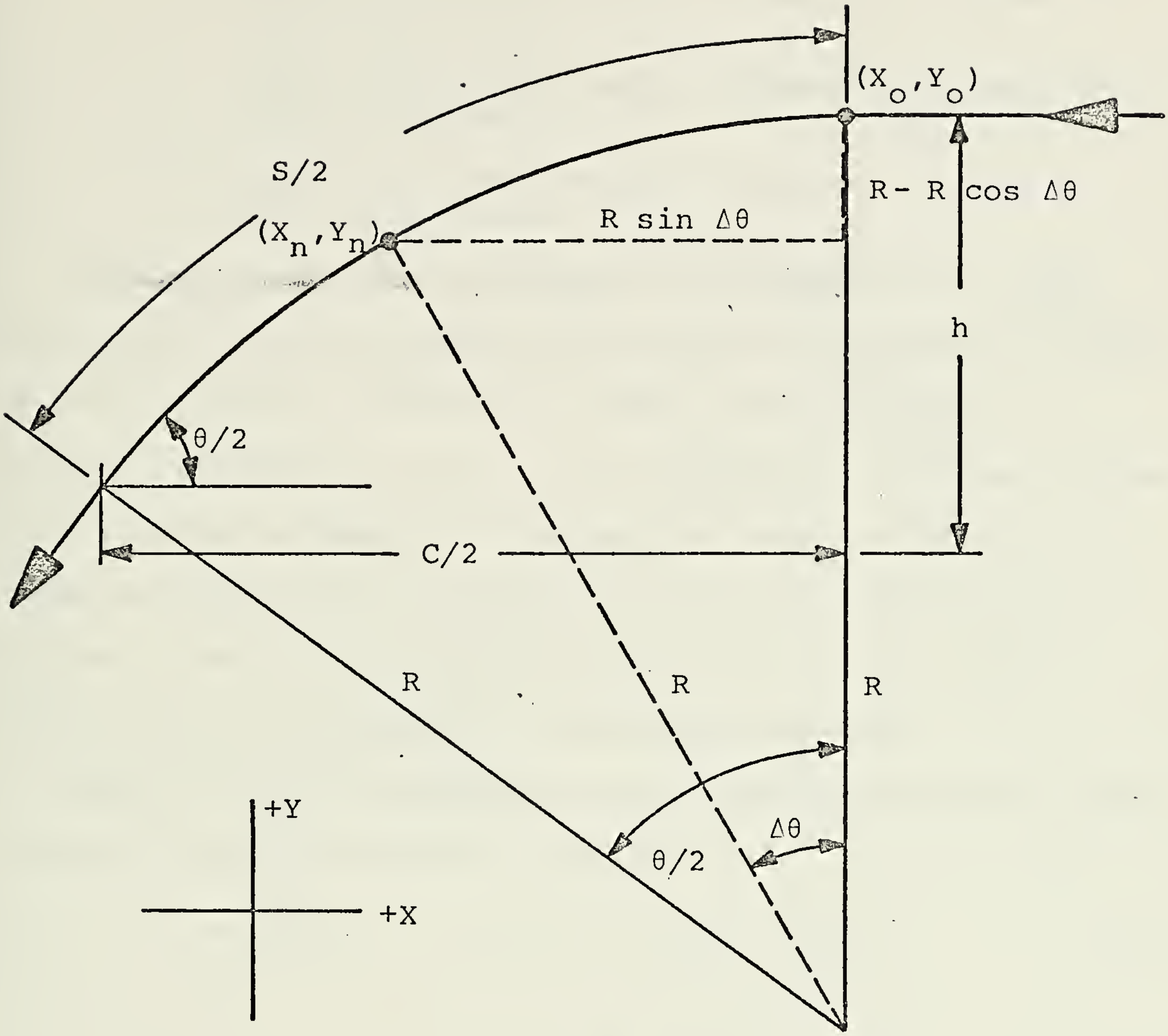


Figure 5. Geometry for Turn with Velocity Vector Initially Parallel to Axis.

a - Initial velocity vector τ degrees to right of axis; right turn,

b - Initial velocity vector τ degrees to right of axis; left turn,

c - Initial velocity vector τ degrees to left of axis; right turn,

d - Initial velocity vector τ degrees to left of axis; left turn.

The same pertinent equations as presented in A.1 are valid here, and the geometrical layouts are shown in Figures 6 and 7. Figure 6 applies to cases a and d above, and Figure 7 to cases b and c. For comparison all cases will be based on the assumption of an initial velocity vector at τ degrees with respect to the X axis and the turn in the XY plane. However, the procedure is directly applicable to other axes and planes with similar orientation.

The following development for obtaining the desired quantities $\Delta\phi$ and C' applies to Figure 6.

$$\alpha = \frac{180-\Delta\theta}{2}, \ \lambda = \beta-\alpha, \ \beta = 180 - (\tau+\Delta\theta) \tag{26}$$

$$\therefore \lambda = 180 - \tau - \Delta\theta - 90 + \frac{\Delta\theta}{2} = 90 - \tau - \frac{\Delta\theta}{2} \tag{27}$$

and
$$\Delta\phi = 90 - \lambda = \tau + \frac{\Delta\theta}{2} \tag{28}$$

Also, it can be seen that $C' = 2 R \sin (\Delta\theta/2)$.

Similarly, the following development applies to Figure 7.

$$\alpha = \frac{180-\Delta\theta}{2}, \ \beta = 90 + (90-\tau) = 90 - \frac{\Delta\theta}{2} \tag{29}$$

$$\Delta\phi = \beta - \alpha = 180 - \tau - 90 + \frac{\Delta\theta}{2} = 90 - \tau + \frac{\Delta\theta}{2} \tag{30}$$

Again, it can be seen that $C' = 2 R \sin (\Delta\theta/2)$.

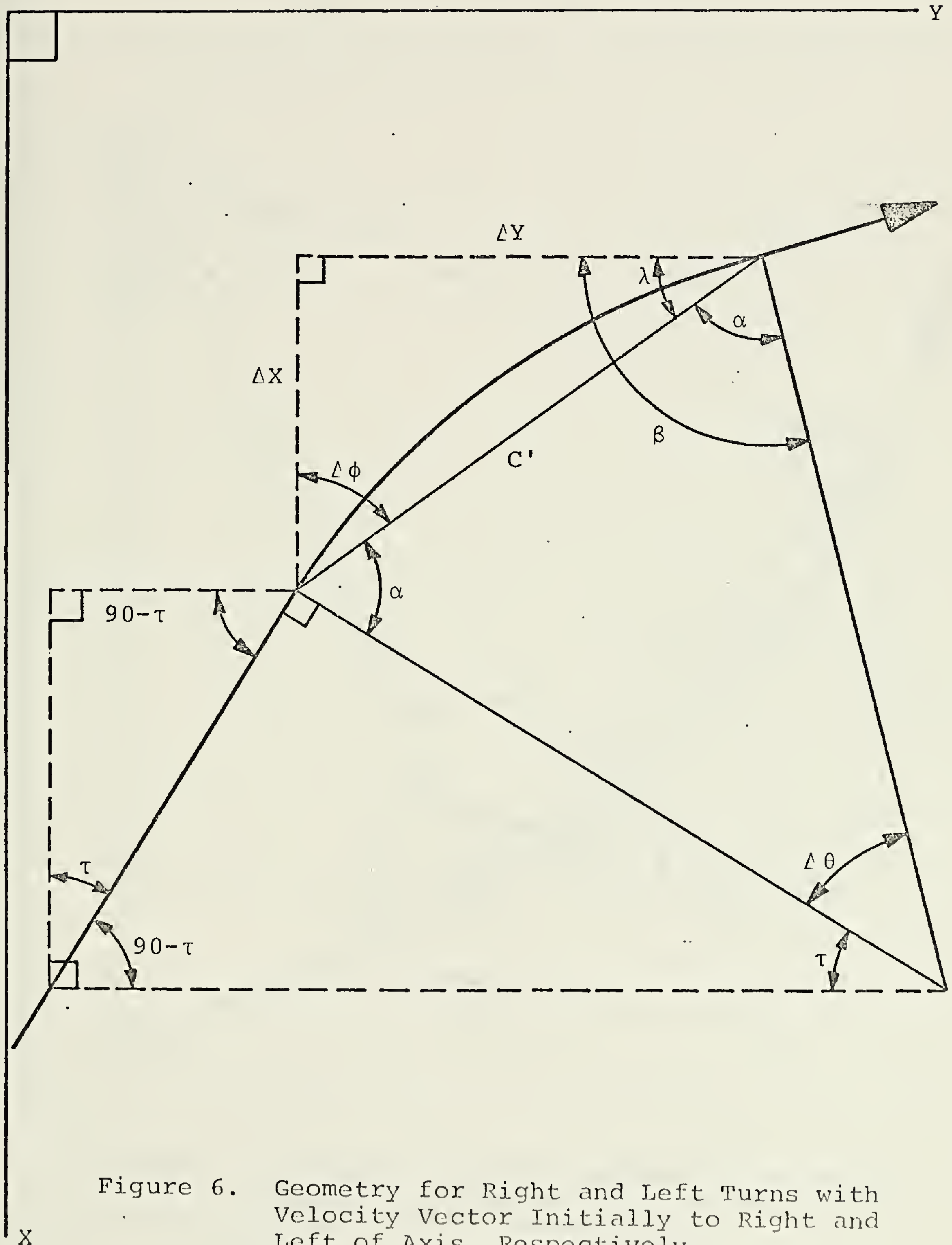


Figure 6. Geometry for Right and Left Turns with Velocity Vector Initially to Right and Left of Axis, Respectively.

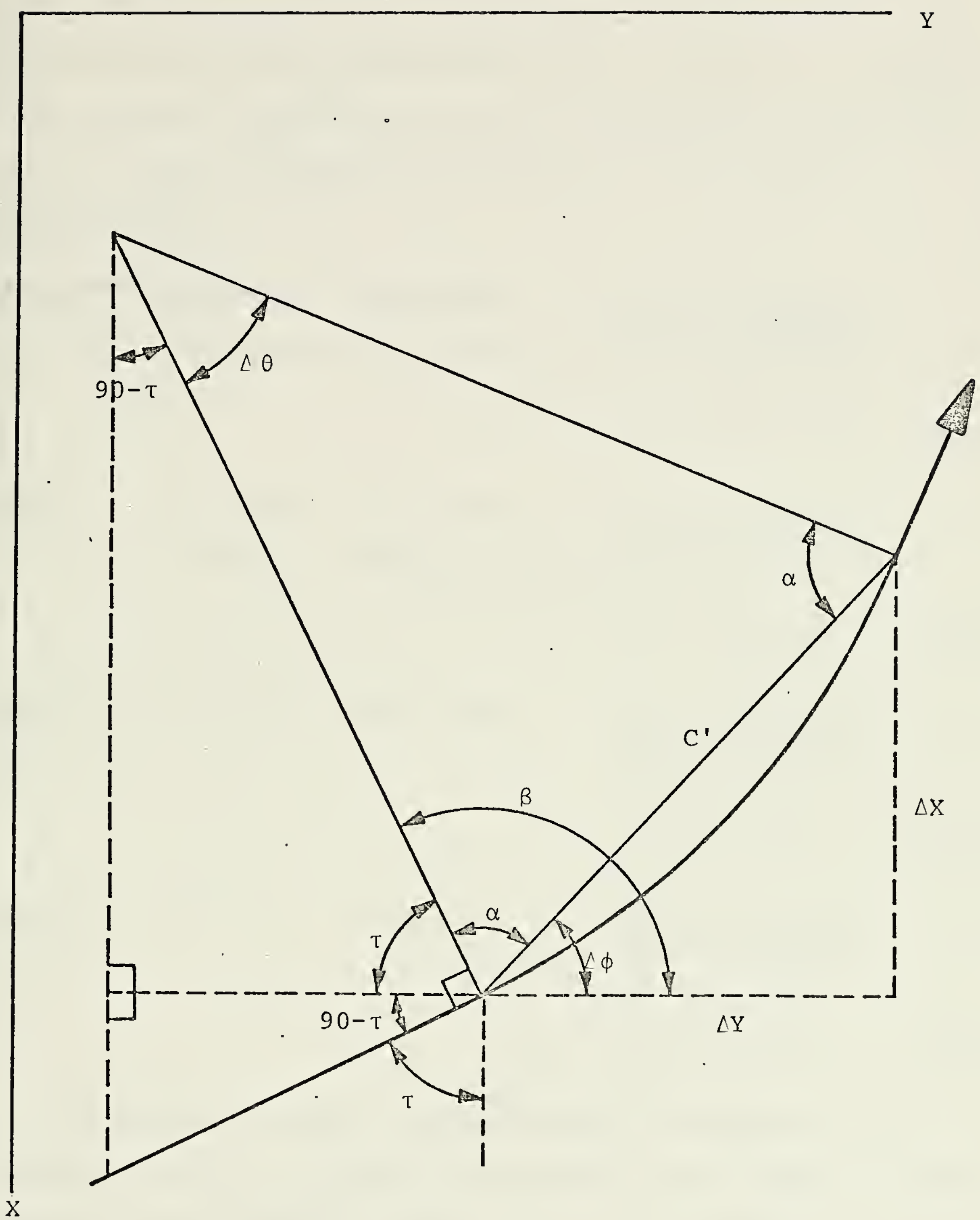


Figure 7. Geometry for Right and Left Turns with Velocity Vector Initially to Left and Right of Axis, Respectively.

Given that the quantities $\Delta\phi$ and C' are available (depending upon the choice of the variable parameters) the track points can be generated for the four cases. A summary for the four cases follows with the appropriate figures for a, b, c, and d included as Figures 8, 9, 10, and 11, respectively.

A.2.a τ to right - right turn

(Example: Points 114-137)

$$\Delta X = C' \cos \Delta\phi$$
$$\Delta Y = C' \sin \Delta\phi$$
$$X_n = X_o - \Delta X$$
$$Y_n = Y_o + \Delta Y \qquad (31)$$

A.2.b τ to right - left turn

(Example: Points 142-145)

$$\Delta X = C' \sin \Delta\phi$$
$$\Delta Y = C' \cos \Delta\phi$$
$$X_n = X_o - \Delta X$$
$$Y_n = Y_o + \Delta Y \qquad (32)$$

A.2.c τ to left - right turn

$$\Delta X = C' \sin \Delta\phi$$
$$\Delta Y = C' \cos \Delta\phi$$
$$X_n = X_o - \Delta X$$
$$Y_n = Y_o - \Delta Y \qquad (33)$$

A.2.d τ to left - left turn

$$\Delta X = C' \cos \Delta\phi$$
$$\Delta Y = C' \sin \Delta\phi$$
$$X_n = X_o - \Delta X$$
$$Y_n = Y_o - \Delta Y \qquad (34)$$

It should be noted that in using this approach it is assumed that when a turn is initiated at some angle, τ, with respect to an axis the quantity $(\tau + \theta)$ is less than or equal to 90 degrees. Otherwise, one must assure that the quantities ΔX and ΔY are added to or subtracted from the initial coordinates correctly. That is, beyond 90 degrees the X or Y

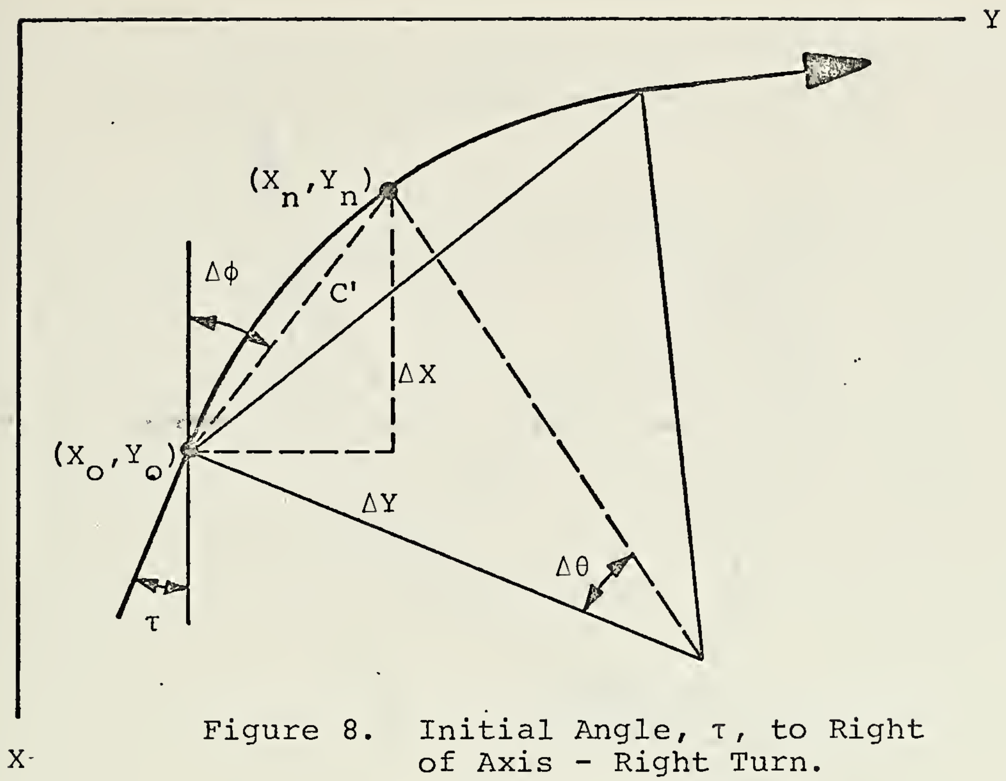


Figure 8. Initial Angle, τ, to Right of Axis - Right Turn.

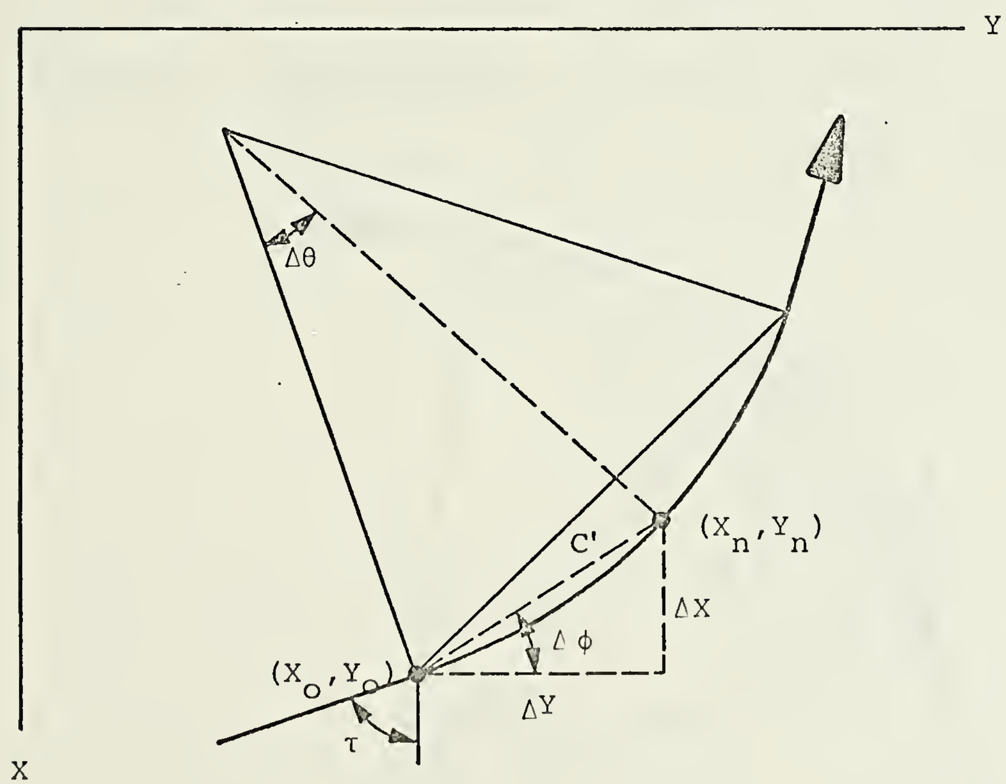


Figure 9. Initial Angle, τ, to Right of Axis - Left Turn.

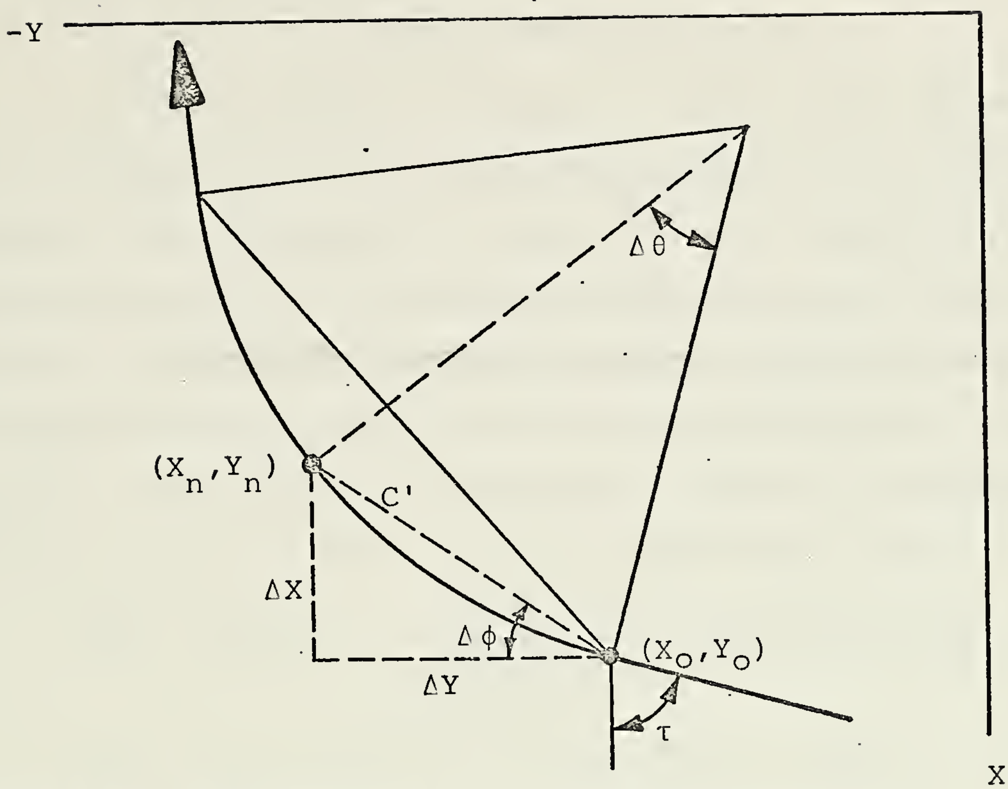


Figure 10. Initial Angle, τ, to Left of Axis - Right Turn.

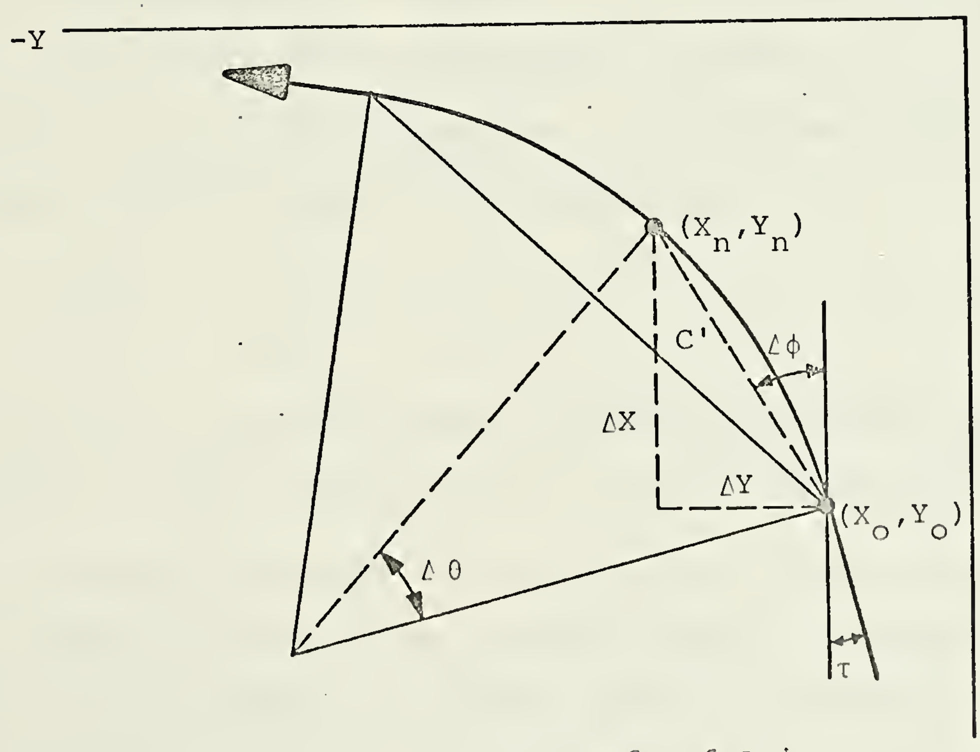


Figure 11. Initial Angle, τ, to Left of Axis - Left Turn.

coordinates may begin to change in directions opposite to that discussed in the scenario described above.

A.3 Initial conditions include a starting point anywhere in the coordinate system with constant-acceleration motion in a straight line. (Example: Points 34 - 113) Specified parameters may include initial and final velocity, time of execution, acceleration (assumed constant), angle of segment with respect to some plane, and final translation from starting point. The geometric arrangement is shown in Figure 12, and the pertinent kinematic relationships follow below

$$\text{Final Velocity} = \frac{2 \cdot S}{\left(\begin{array}{c}\text{Time of} \\ \text{Execution}\end{array}\right)} - \text{Initial Velocity}, \quad (35)$$

$$\text{Final Vel.} = \text{Init. Vel.} + \text{Acceleration} \cdot \text{Time}, \quad (36)$$

$$\text{Z Translation} = S \cdot \sin \alpha, \quad (37)$$

$$\text{X Translation} = S \cdot \cos \alpha, \quad (38)$$

$$S = \tfrac{1}{2}\,(\text{Init. Vel.} + \text{Final Vel.}) \cdot \text{Time}\ . \quad (39)$$

Using these equations to define a track segment, the incremental displacements can be defined as:

$$\Delta X = \Delta S \cdot \cos \alpha, \quad (40)$$

$$\Delta Z = \Delta S \cdot \sin \alpha\ . \quad (41)$$

Then each point along the track is defined by applying these incremental changes relative to the previous track point.

A.4 The last situation to be discussed is that applying to the evasive retreat of the aircraft from the seaborne target. In this scenario a gradual climb is initiated after passing over the ship, and the climb angle of β degrees

requires an additional perspective into the aircraft's motion. Using the techniques discussed previously in subsection A.2 the aircraft's track was defined in the XY plane of the coordinate system. If, however, the incremental Z motion is such as to give a β climb angle the apparent effect in the plane of the aircraft is to cause an increasing radially accelerating turn while maintaining the same constant-acceleration turn in the XY plane. It can be seen in Figure 13 that in the plane of the aircraft the radius of curvature decreases from R_1 to R_2; therefore, the radial acceleration increases as it is inversely related to the radius in a constant velocity model by $R = Vel^2/Acceleration$. This apparent complication is justifiable in the context of actual operation in that an aircraft might, in fact, execute a turn with an increasing amount of radial acceleration. This causes no problems in the simulation as long as the angle, τ, between the appropriate axis and the aircraft's velocity vector is continuously updated in order to correctly initiate the succeeding maneuver.

Two plots of the aircraft motion throughout the entire maneuver are included as Figures 14 and 15. Figure 14 depicts the XZ plane, and Figure 15 shows a top view of the XY plane.

B. TRACK OPTIONS

The groundwork for the track development having been discussed, next follows a description of the options available in using this or any other track in the MCSP. The discussion

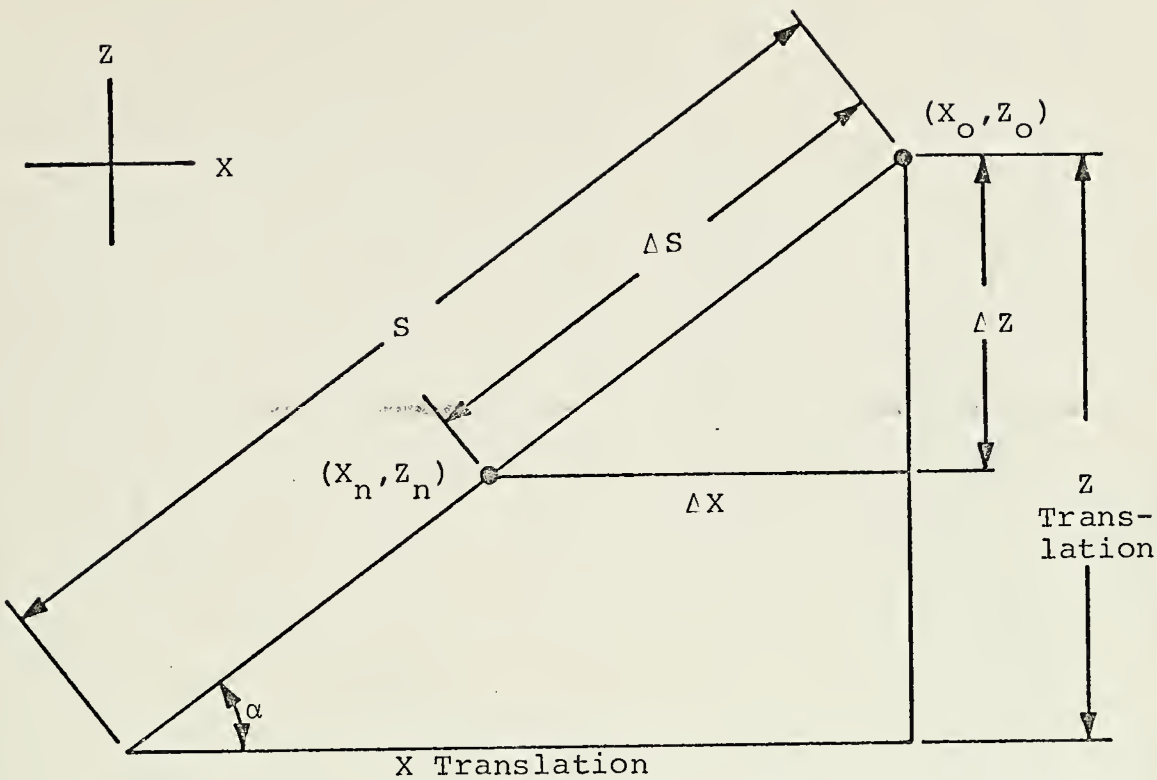


Figure 12. Constant Acceleration Motion in a Straight Line.

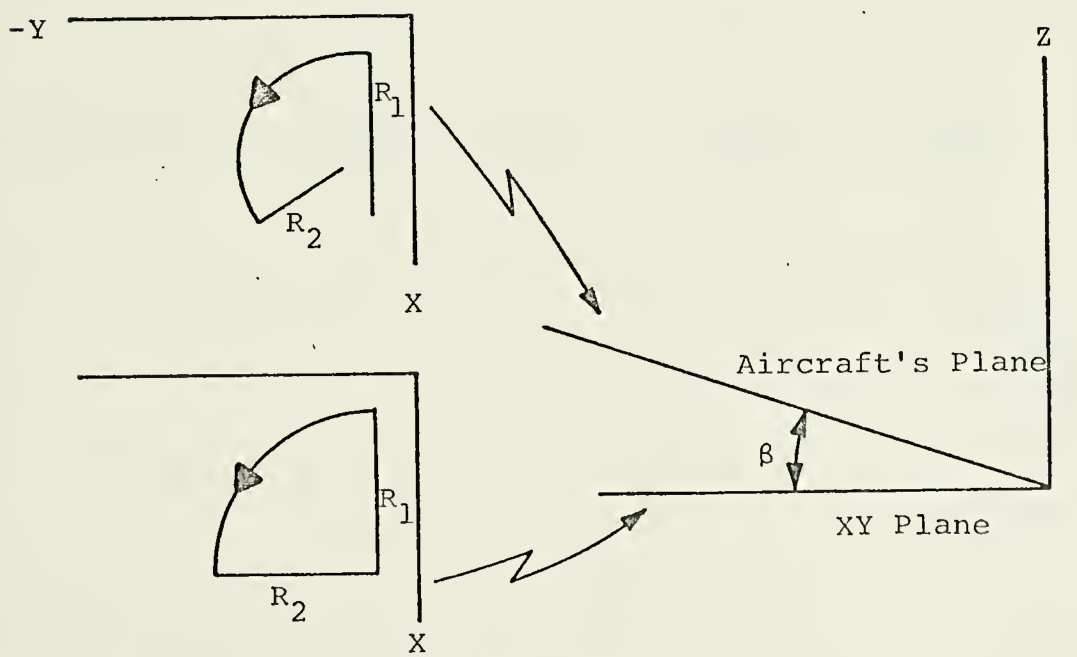
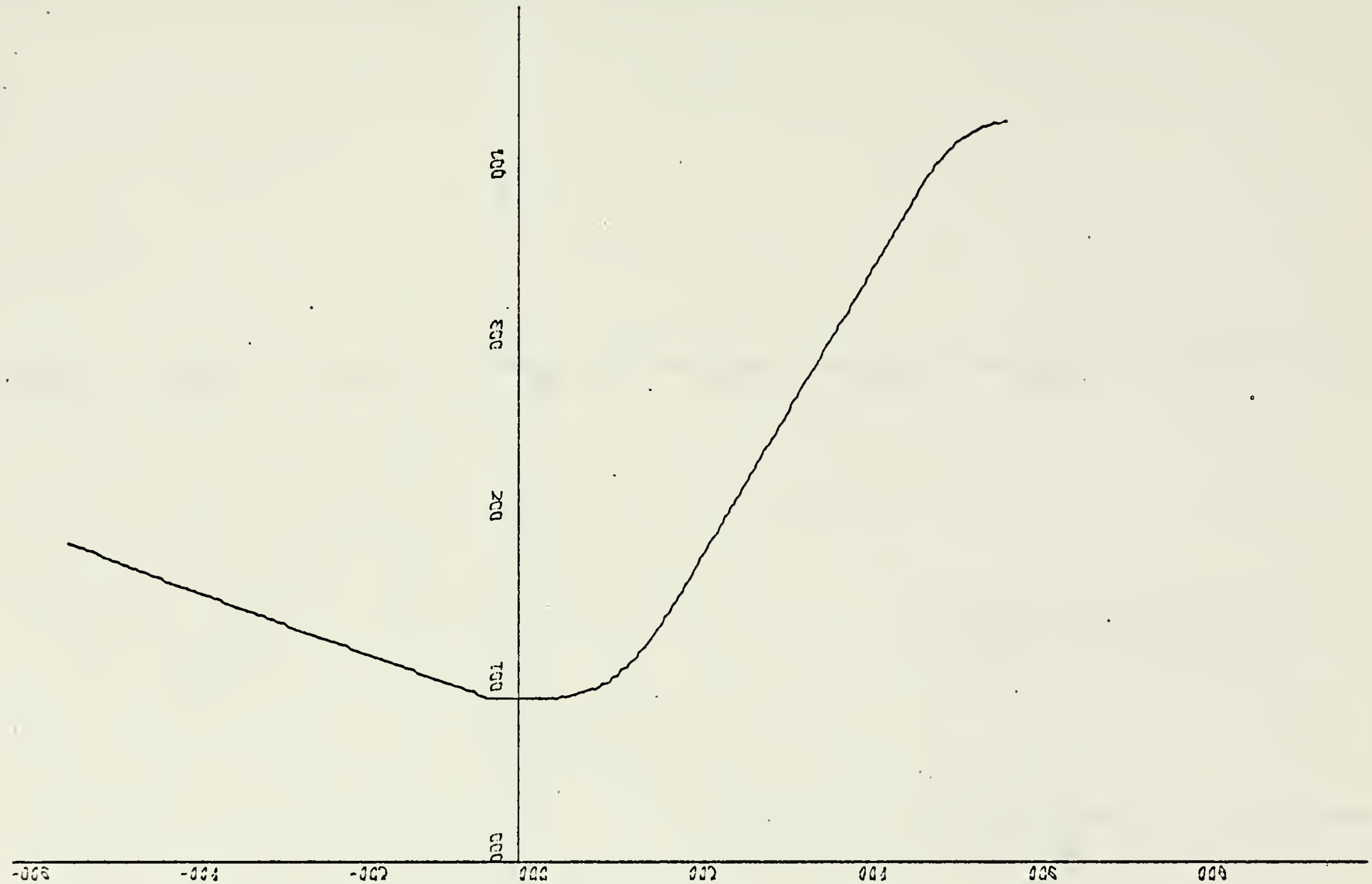


Figure 13. Constant Radial Acceleration Turn with Climb Angle of β.



X scale: 2000 yards/in
Z scale: 1000 yards/in

Figure 14. Aircraft Motion in XZ Plane.

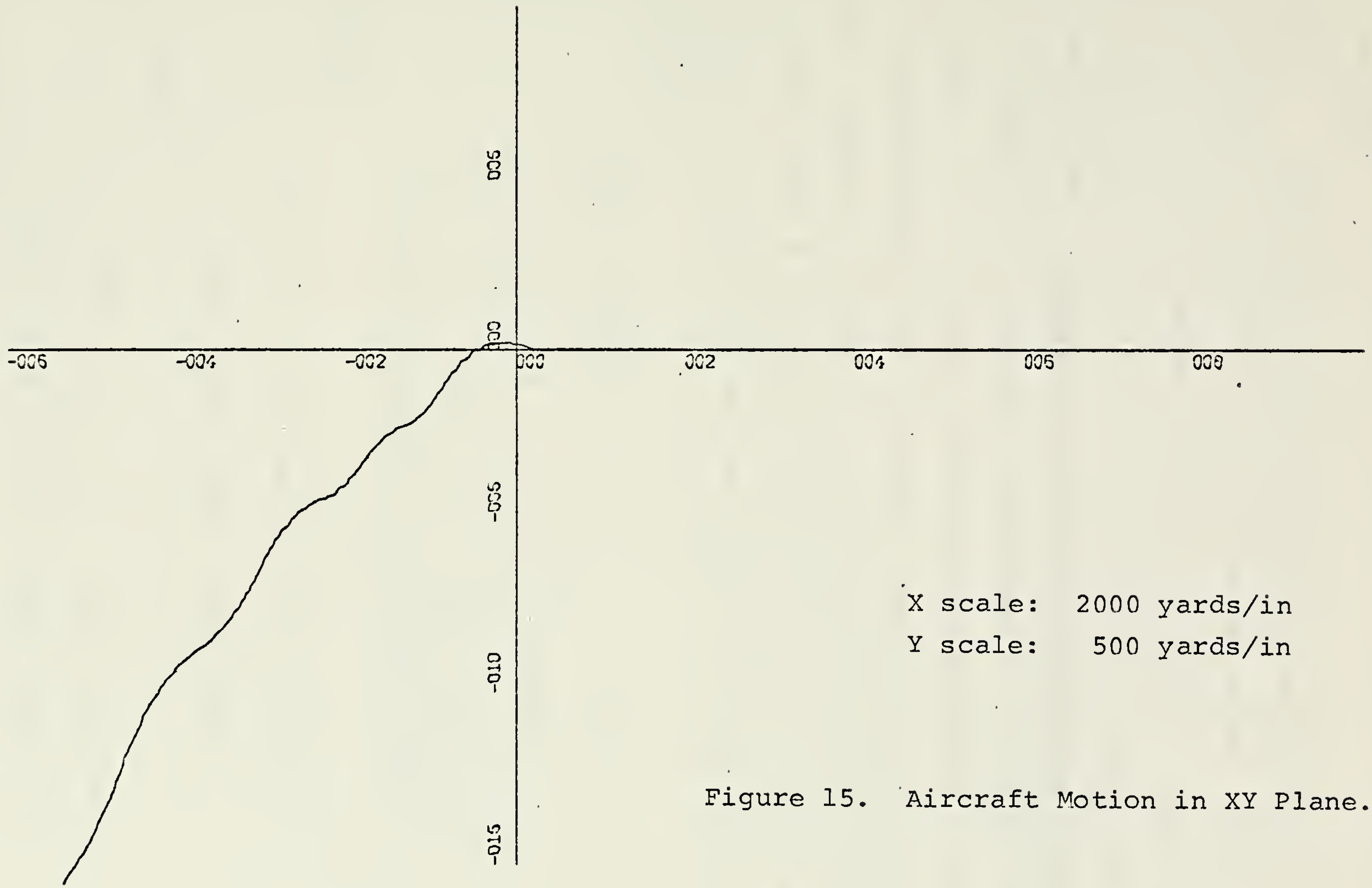


Figure 15. Aircraft Motion in XY Plane.

of implementing the desired options follows in a later section.

B.1 The first option available is the selection of direction of approach of the airborne target relative to the coordinate system centered at the ship. This rotational capability is achieved by inputing the original track in XYZ coordinates, converting into RBE, and adding a desired radial increment to the bearing coordinate of every point shown in Figure 16. If track velocities are also input to the program (an option that is covered later), it can be seen that the component in the Z direction is not affected while those in the X and Y directions are. Following is a derivation to handle this problem, and the vector presentation is shown in Figure 17.

$$|VEL'| = |VEL| \tag{42}$$

$$\dot{X} = -VEL \cdot \cos \beta \qquad \dot{Y} = VEL \cdot \sin \beta \tag{43}$$

$$\dot{X}' = -VEL' \cdot \cos (\beta-\Delta) = -VEL \cdot \cos (\beta-\Delta) \tag{44}$$

$$\dot{Y}' = VEL' \cdot \sin (\beta-\Delta) = VEL \cdot \sin (\beta-\Delta) \tag{45}$$

Using

$$\cos (\beta-\Delta) = \cos \beta \cos \Delta + \sin \beta \sin \Delta \tag{46}$$

$$\sin (\beta-\Delta) = \sin \beta \cos \Delta - \cos \beta \sin \Delta \tag{47}$$

We have

$$\dot{X}' = -VEL \cdot \cos \beta \cos \Delta - VEL \cdot \sin \beta \sin \Delta \tag{48}$$

$$\dot{Y}' = VEL \cdot \sin \beta \cos \Delta - VEL \cdot \cos \beta \sin \Delta \tag{49}$$

And finally,

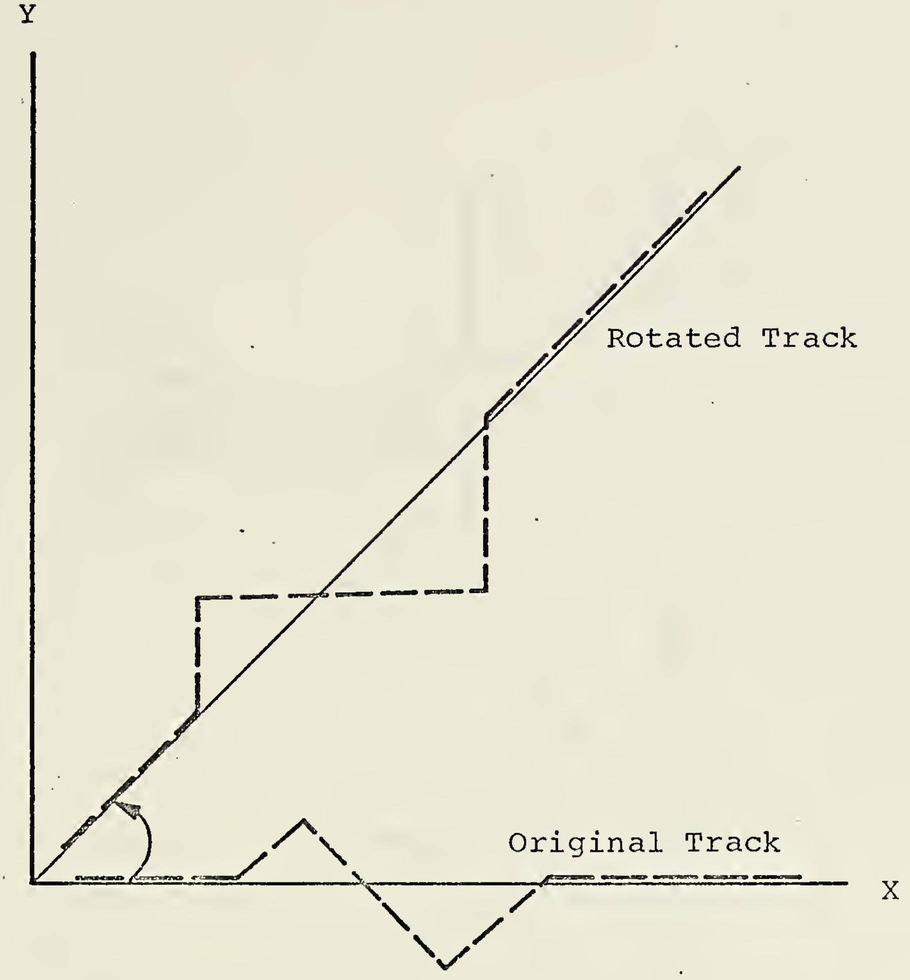


Figure 16. Rotation of Track.

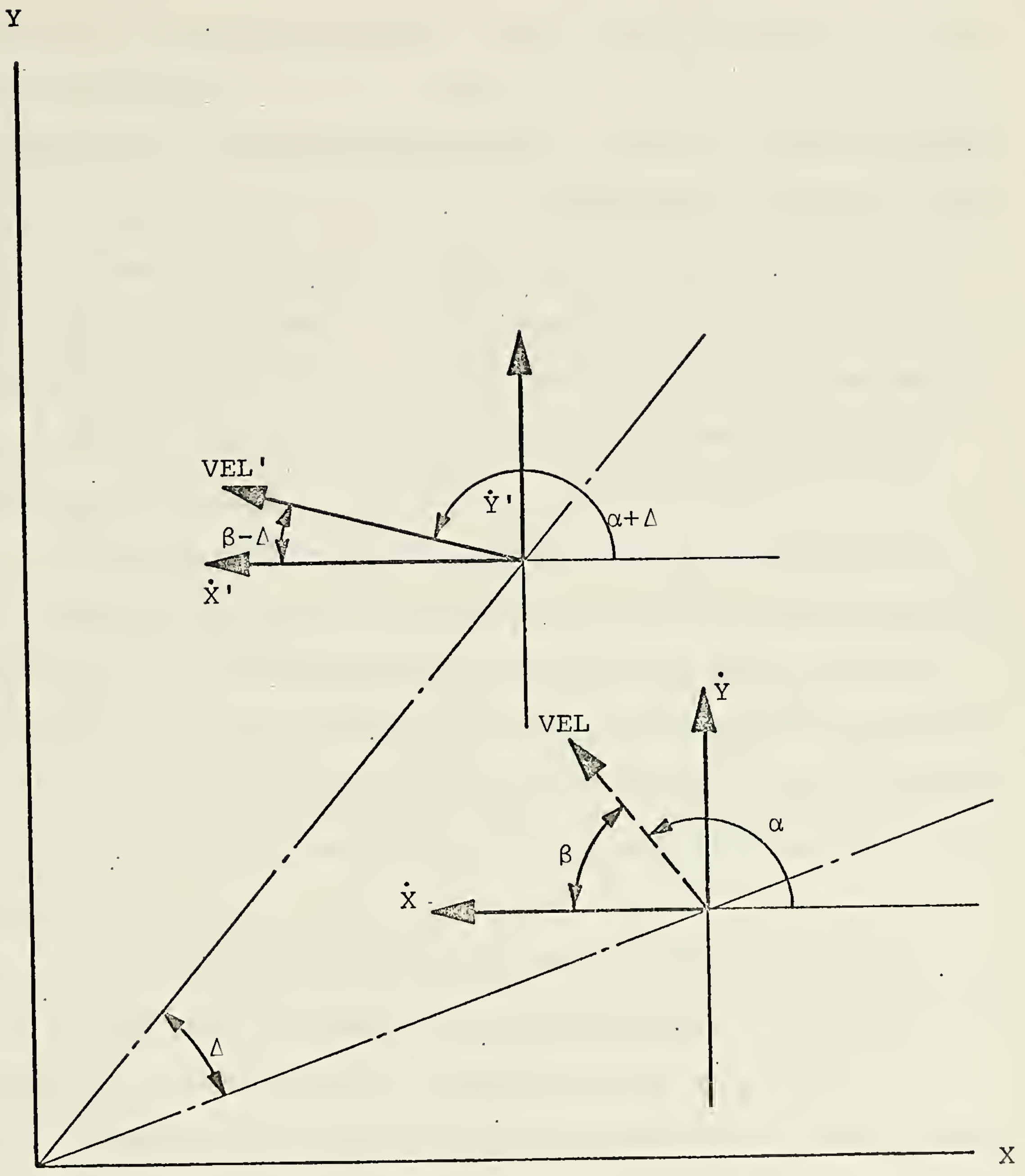


Figure 17. Velocity Vectors for Track Rotation.

$$\dot{X}' = \dot{X} \cos \Delta - \dot{Y} \sin \Delta \tag{50}$$

$$\dot{Y}' = \dot{Y} \cos \Delta + \dot{X} \sin \Delta \tag{51}$$

where $\dot{X}$ and $\dot{Y}$ are the original track velocities and Δ is the angle of rotation.

The MCSP is arranged to provide a choice of the following angles of approach: 0°, 45°, 90°, and 202.5°. If other than these choices is desired the MCSP can be easily altered by referring to the segment of the program several statements after label 13 in the listing of the MCSP included as Appendix C. The variable DEL can be set to the desired radial increment in radians.

B.2 The second option available is the translation of all of the track points by desired amounts for each coordinate axis. The implementation is discussed later, but the method simply involves adding the desired increments to all of the original track points' XYZ coordinates. The velocity components at each track point are not affected by the translation.

B.3 A third option is the choice of adding measurement noise to the track points. If requested, the noise is added to the RBE coordinates and conversion back into noisy XYZ occurs. This noise, as now exists in the MCSP is mean zero, and the user selects the desired standard deviations in range, bearing, and elevation (units are yards, radians, radians).

In conclusion, the procedure for implementation of a track and possible options requires that the XYZ data points be generated offline and read into the MCSP where various

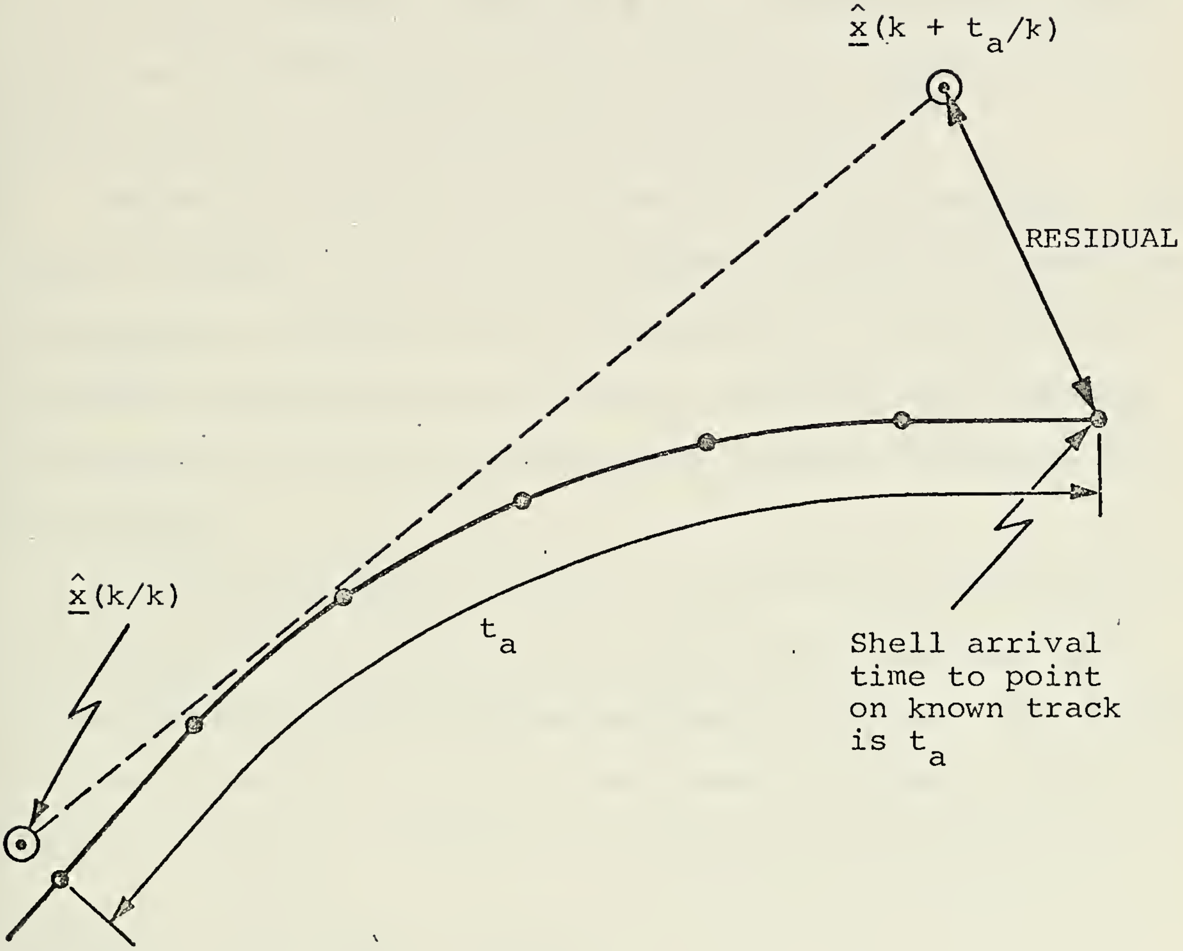
input parameters control the rotation, translation, and noise options applied.

C. SHELL-AT-TARGET ACCURACY

A feature which is an option in the MCSP provides approximate shell-at-target miss distances for additional filter evaluation. This feature is based on the firing table for a 5-inch 54-caliber gun and provides the approximate miss distances, or residuals, in X, Y, and Z directions and, also, the absolute miss distance with time of shell flight. Sections of the MCSP that are specifically responsible for these calculations are Subroutines ACCUR and STUDY.

The principles involved in obtaining the residuals will now be described. The first step required for implementation of the routine requires that the input XYZ track data be provided to program GUNFIRE, included as Appendix D, for offline calculation of the shell flight times to each point using interpolation in the 5-inch 54 firing table, included also in Appendix D. Program GUNFIRE is set up to provide these times for data points out to a range of 7500 yards and having an elevation of from 15 to 90 degrees above the horizon.

The known information includes the XYZ data points, the data point spacing, the shell times to each data point, and the $\underline{\phi}$, or state transition matrix, and the objective, as depicted in Figure 18, is to step back in time from a track point for which the shell arrival time, t_a, is known to a filter estimated point calculated an amount t_a earlier.



● - True track points

⊙ - Filter outputs

Figure 18. Shell-at-target Accuracy Diagram.

Knowing this estimated point $\underline{\hat{x}}(k/k)$ and the $\underline{\phi}$ matrix the filter's prediction of target location an amount t_a later is included in the vector defined by $\underline{\phi}^{(\frac{t_a}{T})}\,\underline{\hat{x}}(k/k)$ where T is the time between samples. The residual is then defined to be equal to the difference between the position states in the vector $\underline{\hat{x}}(k+t_a/k)$ and the actual track point.

Two major problems arise, however, in this approach. The first is that if t_a is not an integer multiple of track time increments, a filter estimate of states is not available. Secondly, implementation of this method generally requires raising the $\underline{\phi}$ matrix to a fractional power--a situation to be avoided.

A reasonable approximation to the desired results above is to step back in time to the first filter estimate prior to the desired time of filter estimation vector. This amounts to some fraction, α, of the time between samples, and the required vector of estimates is given, approximately, by

$\underline{\hat{x}}(k+\alpha/k) = \underline{\phi}^{(\frac{\alpha}{T})}\,\underline{\hat{x}}(k/k)$. Hence,

$$\underline{\hat{x}}(k+\alpha+t_a/k+\alpha) = \underline{\phi}^{(\frac{t_a}{T})}\,\underline{\hat{x}}(k+\alpha/k+\alpha) \tag{52}$$

$$\simeq \underline{\phi}^{(\frac{t_a}{T})}\,\underline{\phi}^{(\frac{\alpha}{T})}\,\hat{x}(k/k) \tag{53}$$

$$\simeq \phi^{(\frac{t_a+\alpha}{T})}\,\hat{x}(k/k) \tag{54}$$

but $\frac{t_a+\alpha}{T}$ is an integer, thus solving the problem of raising $\underline{\phi}$ to a fractional power.

EXAMPLE: Given that the time for a shell to reach data point 100 is 9.7 time periods (the same as assuming a 1-Hz sampling rate) finds the filter's predicted position.

SOLUTION: Because $\hat{\underline{x}}(90.3/90.3)$ is not available, approximate it is $\underline{\phi}^{0.3}\hat{\underline{x}}(90/90)$. Then the filter's prediction vector at time 100 is given by

$$\hat{\underline{x}}(100/90) = \underline{\phi}^{0.3}\ \underline{\phi}^{9.7}\ \hat{\underline{x}}(90/90)$$
$$= \underline{\phi}^{10}\ \hat{\underline{x}}(90/90)\ . \qquad (55)$$

It should be noted that when the shell flight times are generated by GUNFIRE these are applicable only to the original track or a track rotated by the MCSP. If translation of a track is to be performed by the MCSP the shell-at-target miss distances generated will be erroneous, and this option should not be used.

D. ONLINE CALCULATION OF COVARIANCE MATRIX OF POSITION MEASUREMENT NOISE

The covariance matrix of measurement noise, $\underline{R}$, is a measure of the noise associated with the measurements and is used in calculating the gains. $\underline{R}$ is defined as $E[\underline{n}(k)\ \underline{n}^T(k)]$ where $\underline{n}(k)$ is the additive noise vector at time k.

In problems where the measurements are linearly related to the filter states, $\underline{R}$ can be approximated as a constant matrix for the entire track, an option available in the MCSP.

However, when the observations presented to the filter are range, bearing, and elevation (with noise assumed mutually independent) and the state equations are in rectangular coordinates, the required coordinate conversion generates a coupled, non-independent relationship among the noise terms

in the filtering coordinate system. This section presents a derivation to define the resultant $\underline{R}$ matrix for the given coordinate conversion.

The radar measures range, r, bearing, θ, and elevation, ϕ, and these measurements are made available to the estimator at sample times with the parameter k as an index. The measurements also include independent noise terms $n_1(k)$, $n_2(k)$, and $n_3(k)$ that are assumed to be characterized by random, white distributions with mean zero and known variances. This implies that

$$E[n_i(k)]_{i=1,2,3} = 0 \qquad \text{for all } k \tag{56}$$

$$E[n_1(k)n_1(j)] \qquad \begin{cases} 0 & j \neq k \\ \sigma_R^2 & j = k \end{cases} \tag{57}$$

$$E[n_2(k)n_2(j)] \qquad \begin{cases} 0 & j \neq k \\ \sigma_B^2 & j = k \end{cases} \tag{58}$$

$$E[n_3(k)n_3(j)] \qquad \begin{cases} 0 & j \neq k \\ \sigma_E^2 & j = k \end{cases} \tag{59}$$

$$\left.\begin{array}{ll} E[n_1(k)n_2(j)] & 0 \\ E[n_1(k)n_3(j)] & 0 \\ E[n_2(k)n_3(j)] & 0 \end{array}\right\} \text{ for all } k,j \tag{60}$$

The terms σ_R^2, σ_B^2, and σ_E^2 must be known. The coordinate system is the same as shown in Figure 2.

The rectangular coordinates of the target are

$$X(k) = r(k) \cos \theta(k) \cos \phi(k) \tag{61}$$

$$Y(k) = r(k) \sin \theta(k) \cos \phi(k) \tag{62}$$

$$Z(k) = r(k) \sin \phi(k) \ . \tag{63}$$

However, the output of the radar is

$$S_1(k) = r(k) + n_1(k) \tag{64}$$

$$S_2(k) = \theta(k) + n_2(k) \tag{65}$$

$$S_3(k) = \phi(k) + n_3(k) \tag{66}$$

and what the estimator receives is given by the noisy XYZ values denoted by $Z_1(k)$, $Z_2(k)$, $Z_3(k)$ and defined as follows

$$Z_1(k) = S_1(k) \cos S_2(k) \cos S_3(k) \tag{67}$$

$$Z_2(k) = S_1(k) \sin S_2(k) \cos S_3(k) \tag{68}$$

$$Z_3(k) = S_1(k) \sin S_3(k) \ . \tag{69}$$

If equations (64) - (66) are substituted into equations (67)-(69), trigonometric identities used with subsequent expansion, and it is assumed that the bearing measurement noise, $n_2(k)$, and elevation measurement noise, $n_3(k)$, are small so that $\cos n_2(k) \simeq 1$, $\sin n_2(k) \simeq n_2(k)$, $\cos n_3(k) \simeq 1$, and $\sin n_3(k) \simeq n_3(k)$,[3] we have

$$Z_1(k) = X(k) + v_1(k) \tag{70}$$

$$Z_2(k) = Y(k) + v_2(k) \tag{71}$$

$$Z_3(k) = Z(k) + v_3(k) \tag{72}$$

where $v_1(k)$, $v_2(k)$, and $v_3(k)$ represent additive noise and are given by

---

[3]In most radar tracking systems the standard deviations in bearing and elevation are less than .005 radians, or .286 degrees. The 3σ point, which includes 99% of this is, therefore, approximately 0.86 degrees. For comparison,

$\sin 0.86° = .0153$ vs. .015 radians

$\cos 0.86° = .9999$ vs. 1.

$$
\begin{aligned}
v_1(k) = & - r(k)\ n_3(k)\ \cos\ \theta(k)\ \sin\ \phi(k) \\
& - r(k)\ n_2(k)\ \sin\ \theta(k)\ \cos\ \phi(k) \\
& + r(k)\ n_2(k)\ n_3(k)\ \sin\ \theta(k)\ \sin\ \phi(k) \\
& + n_1(k)\ \cos\ \theta(k)\ \cos\ \phi(k) \\
& - n_1(k)\ n_3(k)\ \cos\ \theta(k)\ \sin\ \phi(k) \\
& - n_1(k)\ n_2(k)\ \sin\ \theta(k)\ \cos\ \phi(k) \\
& + n_1(k)\ n_2(k)\ n_3(k)\ \sin\ \theta(k)\ \sin\ \phi(k)
\end{aligned} \tag{73}
$$

$$
\begin{aligned}
v_2(k) = & - r(k)\ n_3(k)\ \sin\ \theta(k)\ \sin\ \phi(k) \\
& + r(k)\ n_2(k)\ \cos\ \theta(k)\ \cos\ \phi(k) \\
& - r(k)\ n_2(k)\ n_3(k)\ \cos\ \theta(k)\ \sin\ \phi(k) \\
& + n_1(k)\ \sin\ \theta(k)\ \cos\ \phi(k) \\
& - n_1(k)\ n_3(k)\ \sin\ \theta(k)\ \sin\ \phi(k) \\
& + n_1(k)\ n_2(k)\ \cos\ \theta(k)\ \cos\ \phi(k) \\
& - n_1(k)\ n_2(k)\ n_3(k)\ \cos\ \theta(k)\ \sin\ \phi(k)
\end{aligned} \tag{74}
$$

$$
\begin{aligned}
v_3(k) = & \ r(k)\ n_3(k)\ \cos\ \phi(k) + n_1(k)\ \sin\ \phi(k) \\
& + n_1(k)\ n_3(k)\ \cos\ \phi(k)
\end{aligned} \tag{75}
$$

Using the assumed mutual independence of the noise processes and the state variables which implies $E[n_i\ n_j] = E[n_i] \cdot E[n_j]$, $i \neq j$, gives

$E[v_1(k)] = E[v_2(k)] = E[v_3(k)] = 0$, for all k, because

$E[n_1(k)] = E[n_2(k)] = E[n_3(k)] = 0.$

Then, if it is assumed that $r^2(k) \gg \sigma_R^2$, the following terms of the $\underline{R}$ matrix can be defined. In every case

$$E[v_m(k)\ v_n(j)] = 0 \text{ for } j \neq k \text{ and } m,n = 1,2,3 \tag{76}$$

$$\begin{aligned} E[v_1(k)\ v_1(j)] = {} & r^2(k)\ \sigma_E^2 \cos^2\theta(k) \sin^2\phi(k) \\ & + r^2(k)\ \sigma_B^2 \sin^2\theta(k) \cos^2\phi(k) \\ & + r^2(k)\ \sigma_B^2\ \sigma_E^2 \sin^2\theta(k) \sin^2\phi(k) \\ & + \sigma_R^2 \cos^2\theta(k) \cos^2\phi(k) \end{aligned} \tag{77}$$

$$\begin{aligned} E[v_1(k)\ v_2(j)] = {} & E[v_2(k)\ v_1(j)] = \\ & r^2(k) \cos\theta(k) \sin\theta(k) \sin^2\phi(k) \\ & \cdot [\sigma_E^2 - \sigma_B^2\ \sigma_E^2] \\ & + \sin\theta(k) \cos\theta(k) \cos^2\phi(k) [-r^2(k)\sigma_B^2 + \sigma_R^2] \end{aligned} \tag{78}$$

$$\begin{aligned} E[v_1(k)\ v_3(j)] = {} & E[v_3(k)\ v_1(j)] = \\ & \cos\phi(k) \sin\phi(k) \cos\theta(k) [\sigma_R^2 - r^2(k)\sigma_E^2] \end{aligned} \tag{79}$$

$$\begin{aligned} E[v_2(k)\ v_2(j) = {} & r^2(k)\ \sigma_E^2 \sin^2\theta(k) \sin^2\phi(k) \\ & + r^2(k)\ \sigma_B^2 \cos^2\theta(k) \cos^2\phi(k) \\ & + r^2(k)\ \sigma_B^2\ \sigma_E^2 \cos^2\theta(k) \sin^2\phi(k) \\ & + \sigma_R^2 \sin^2\theta(k) \cos^2\phi(k) \end{aligned} \tag{80}$$

$$\begin{aligned} E[v_2(k)\ v_3(j)] = {} & E[v_3(k)\ v_2(j)] = \\ & \sin\theta(k) \sin\phi(k) \cos\phi(k) [\sigma_R^2 - r^2(k)\sigma_E^2] \end{aligned} \tag{81}$$

$$E[v_3(k)\ v_3(j)] = r^2(k)\ \sigma_E^2 \cos^2 \phi(k) + \sigma_R^2 \sin^2 \phi(k) \tag{82}$$

A matrix summary of $\underline{R}$ for the three-dimensional coordinate system is

$$R = \begin{bmatrix} E[v_1(k)\ v_1(j)] & E[v_1(k)\ v_2(j)] & E[v_1(k)\ v_3(k)] \\ E[v_2(k)\ v_1(j)] & E[v_2(k)\ v_2(j)] & E[v_2(k)\ v_3(j)] \\ E[V_3(k)\ v_1(j)] & E[V_3(k)\ v_3(j)] & E[v_3(k)\ v_3(j)] \end{bmatrix} \tag{83}$$

This covariance matrix is included in the MCSP by the subroutine RNOISE and functions in the following manner. The current filter estimate is used to predict ahead one time increment to define the expected position states. These predicted XYZ values are taken by routine RNOISE, converted to RBE, and, along with the standard deviations, σ_R, σ_B, and σ_E input as data, which, when squared, give the necessary variances,are used to define the covariance matrix of measurement noise required for calculating the gains for the next filter estimate. That is, $\underline{\hat{x}}(k + 1/k)$ is used to obtain $\underline{R}$ for computing the gains needed to determine $\underline{\hat{x}}(k + 1/k + 1)$.

Again, this approach is an option which can be used with the Monte-Carlo simulation.

E. COVARIANCE MATRIX OF STATE EXCITATION

A filter designed to exactly follow a constant-velocity target will lag an accelerating target unless some technique

is incorporated to allow the filter to respond to the changes with increased gains to more heavily weight the latest observations. Increasing these gains is equivalent to increasing the bandwidth of the filter thus insuring that it is able to respond to deviations from the expected target performance.

The compromise made, however, is that additional measurement noise is able to enter into filter estimation and prediction through the increased bandwidth thus degrading the ability of the filter to reject measurement noise. There is, therefore, a balance which must be met between too small a covariance matrix of state excitation, $\underline{Q}$, and a resultant lagging filter and too large a $\underline{Q}$ with subsequent erratic filter behavior.

The MCSP can accommodate the use of a $\underline{Q}$ matrix in two ways--by using a constant input matrix of the anticipated covariance of state excitation in the calculation of all gains or by generating, online, $\underline{Q}$ matrices through a procedure that is adaptive to the residuals detected between predicted and estimated filter states. The alternative to these methods, as the MCSP currently exists, is to use no $\underline{Q}$ and input zeroes in its place.

The first of these methods requires some prior analysis of the target motion before the Monte-Carlo simulation is exercised, and with a modeled track, such as that designed for simulation with the MCSP, the variances can be evaluated using the track data.

The most direct method for obtaining terms to define a constant $\underline{Q}$ matrix is to take successive differences of the position data to approximate velocities, accelerations, and acceleration rates. Having these arrays of accelerations and acceleration rates in each dimension of the coordinate system, the mean and variance of each array can be calculated with the usual statistical procedures. The resultant quantities provide the necessary variances for determining $\underline{Q}$ for either constant-velocity or constant-acceleration models if the means produced are, in fact, close to zero (the assumed mean of random forcing). The implementation of this approach is included at the end of the track generating program in Appendix B, and the output of that program is also there showing the variances generated by the above technique.

While these variances are defined from actual track data it must be remembered that the resulting constant $\underline{Q}$ matrix will inevitably be sub-optimal since the accelerations or acceleration rates do not, in most cases, occur uniformly over the track; for some points $\underline{Q}$ will be "too small," thus not compensating for target motion and for other points $\underline{Q}$ will be "too large" allowing for measurement noise degradation of filter performance.

The use of a well-chosen constant $\underline{Q}$ matrix does, however, offer two distinct advantages. The first is that, if well chosen, it cannot help but improve the performance of a filter not dynamically adequate to follow a maneuvering target and, second, it offers simplicity and subsequent reduced

computation time and core area requirements in comparison to an adaptive technique.

The $\underline{Q}$ matrix, when implemented as a constant matrix, is defined by

$$\underline{Q} = \underline{\Gamma}\ E[\underline{w}\ \underline{w}^T]\ \underline{\Gamma}^T \tag{84}$$

where $E[w\ w^T]$ is the covariance matrix of state excitation, and $\underline{\Gamma}$ is necessary to relate changes in velocity or acceleration for the constant-velocity or constant-acceleration filters, respectively, through successive integrations, to the appropriate states, For the XYZ coordinate system, when the random forcing is assumed to be de-coupled,

$$E[\underline{w}\ \underline{w}^T] = \begin{bmatrix} \sigma_X^2 & 0 & 0 \\ 0 & \sigma_Y^2 & 0 \\ 0 & 0 & \sigma_Z^2 \end{bmatrix}$$

The matrix $\underline{\Gamma}$ for a de-coupled constant-velocity filter in three dimensions is given by

$$\underline{\Gamma}_V = \begin{bmatrix} \frac{T^2}{2} & 0 & 0 \\ T & 0 & 0 \\ 0 & \frac{T^2}{2} & 0 \\ 0 & T & 0 \\ 0 & 0 & \frac{T^2}{2} \\ 0 & 0 & T \end{bmatrix}$$

where T is the time between measurements. The constant-acceleration $\underline{\Gamma}$ matrix for a de-coupled target motion model is

$$\underline{\Gamma}_A = \begin{bmatrix} T^3/6 & 0 & 0 \\ T^2/2 & 0 & 0 \\ T & 0 & 0 \\ 0 & T^3/6 & 0 \\ 0 & T^2/2 & 0 \\ 0 & T & 0 \\ 0 & 0 & T^3/6 \\ 0 & 0 & T^2/2 \\ 0 & 0 & T \end{bmatrix}.$$

In the MCSP the matrices $\underline{\Gamma}$ and $E[\underline{w}\ \underline{w}^T]$ are input separately, and the necessary matrix products are formed to define the $\underline{Q}$ matrix.

The second of the two methods available is an adaptive technique described in (4). The Q matrix, at any point along the track, is defined to be

$$\underline{Q}(k) = K_1 [\hat{\underline{x}}(k/k) - \hat{\underline{x}}(k/k-1)][\hat{\underline{x}}(k/k) - \hat{\underline{x}}(k/k-1)]^T - K_2 \cdot \underline{Q}(k-1) \qquad (88)$$

where $\hat{\underline{x}}(k/k)$ is the filter's state estimation vector at time k and $\hat{\underline{x}}(k/k-1)$ is the state prediction vector determined at time k-1. It is this residual between the predicted states at time k using knowledge through time k-1 and the estimated

states at time k which offers a measure of the state excitation acting to cause the discrepancy. The terms K_1 and K_2 are weighting constants used to limit the degree to which $\underline{Q}$ can change from one point to the next. By increasing K_1, $\underline{Q}$ becomes much more responsive to residuals and, also, noisy estimates while an increase in K_2 results in a type of damping on the $\underline{Q}$ matrix.

This is, to say the least, a seat-of-the-pants approach, but, given a very complex modeling scenario, it offers many points in its favor. First, it is very simple and easy to implement with small additional computation time and storage required. Secondly, it does, in fact, vary the magnitude of $\underline{Q}$ quickly, from point to point, rather than waiting for some threshold to be passed before alteration. Also, in a steady-state situation where the residual goes to zero, so does $\underline{Q}$, thus maintaining minimum bandwidth as desired.

It should be noted that the nature of gain calculations requires that the $\underline{Q}$ matrix generated at point k is used to determine the gains for calculating the filter estimates at time k + 1, but given a reasonably high sampling rate the loss in performance should not be significant.

This adaptive method is easily requested, as will be shown in the section on using the MCSP, and the logic for its implementation appears at the end of subroutine UPDATE.

F. INITIALIZATION

The initialization of a Kalman filter can greatly affect the speed with which filter estimates and predictions improve.

Many techniques are available, and three options exist in the MCSP for user selection.

The first is a default option which is used if the other two methods are not requested. The filter initializing vector, $\hat{\underline{x}}(0/-1)$, has its position states defined as the first observation, $\underline{z}$, and the remaining states are initialized to zero. In conjunction with this choice it is advisable to initialize the prediction covariance matrix with very large values to assure large gains during the transient period.

The second possibility is to read in the initial conditions, an option that will be described in the next section. This is fine if reasonable initial conditions are known, but care must be taken not to detract from filter performance by choosing erroneous values offering worse initial conditions than the default case. Again, it pays to choose a large prediction covariance matrix if the gains are not precalculated.

The third option is to have the MCSP generate "synthetic" estimates by solving difference equations, online. With this method it takes two measurements to generate a velocity estimate and a third to generate an acceleration estimate if needed. After either two or three measurements, depending on whether a constant-velocity or constant-acceleration model is used, an initializing estimation covariance matrix is generated as follows in the MCSP:

Constant-Velocity Model

$$\underline{P}(1/1) = \begin{bmatrix} 1 & 1/T & 0 & 0 & 0 & 0 \\ 1/T & 2/T^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1/T & 0 & 0 \\ 0 & 0 & 1/T & 2/T^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1/T \\ 0 & 0 & 0 & 0 & 1/T & 2/T^2 \end{bmatrix} \qquad (89)$$

Constant-Acceleration Model

$$\underline{P}(2/2) = \begin{bmatrix} 1 & 1/T & 1/T^2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1/T & 2/T^2 & 3/T^3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1/T^2 & 3/T^3 & 6/T^4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1/T & 1/T^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/T & 2/T^2 & 3/T^3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/T^2 & 3/T^3 & 6/T^4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1/T & 1/T^2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1/T & 2/T^2 & 3/T^3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1/T^2 & 3/T^3 & 6/T^4 \end{bmatrix} \qquad (90)$$

The appropriate estimation covariance matrix is then used to generate a prediction covariance matrix for calculating the next gain matrix. With this option it is not necessary to input an initial prediction covariance matrix. The logic for its implementation can be found in subroutine UPDATE of the MCSP included as Appendix C.

IV. DESCRIPTION OF MONTE-CARLO SIMULATION PROGRAM (MCSP)

A. APPLICATION

The Monte-Carlo Simulation Program is designed for use in comparing 6th and 9th-order filter evaluations of airborne tracking methods and incorporates user controlled selection of the options mentioned in the previous sections and several other features that are discussed in this section. A listing of the program in FORTRAN IV appears in Appendix C. Comments appear liberally throughout the listing to aid in following the flow of the program as well as for pertinent information related to variable definitions and input instructions. The following sub-sections elaborate on the procedures to follow in using the MCSP and describe the output provided.

B. INPUTS

The input cards are described in sequence as required by the MCSP with description of the different arrangements dictated by the options requested. Unless otherwise specified the data should be double precision, D, format and right justified in the fields designated because calculations in the MCSP are done in double precision to minimize truncation errors.

CARD 1

COL(S)	VARIABLE (ARRAY) NAME	FORMAT	DESCRIPTION
1 - 10	LOOP	I 10	Integer specifying the number of Monte-Carlo runs desired
11 - 20	KVEL	I 10	An integer "1" specifies that true velocity data for the track is available and is to be input

COL(S)	VARIABLE (ARRAY) NAME	FORMAT	DESCRIPTION
31 - 44	LOOK	14 I 1	A 14-element integer array for outputing plots on the high speed printer. A "1" in the specified column produces a plot as follows: 31 - Mean miss distance when shell arrives (from shell-at-target calculations). Default option automatically prohibits this output if the gunfire option is not requested. 32 - Mean filter estimation error in X position calculation 33 - Same for Y coordinate 34 - Same for Z coordinate 35 - Variance of X position estimates 36 - Same for Y 37 - Same for Z 38 - Mean filter estimation error in X velocity calculation 39 - Same for Y 40 - Same for Z 41 - Variance of X velocity estimates 42 - Same for Y 43 - Same for Z 44 - Estimated target tangential velocity in ft/sec
51 - 70	TIME	D 20.0	Time between measurements in secs.

CARD 2

COL(S)	VARIABLE (ARRAY) NAME	FORMAT	DESCRIPTION
1 - 5	N	I 5	Integer specifying the number of states, i.e. order of filter such as 6 or 9
6 - 10	NN	I 5	An integer specifying the number of track points or measurements

COL(S)	VARIABLE (ARRAY) NAME	FORMAT	DESCRIPTION
11 - 15	OPT	I 5	An integer "1" causes the program to generate synthetic measurements for filter initialization
16 - 20	BEGIN	I 5	An integer "1" signifies that the filter initialization vector $\hat{\underline{x}}(0/-1)$ is to be input data
21 - 25	IR	I 5	An integer "0" causes the program to generate, online, R matrices as discussed in III.D. An integer "1" signifies that a constant R matrix is input data.
26 - 30	ADAPTQ	I 5	An integer "1" incorporates the adaptive Q matrix calculations
31 - 35	IG	I 5	An integer "1" causes the MCSP to punch out the gain matrices for each point on the track with one row of a matrix per card and having two coded integers in the first 10 columns of each card. The first specifies the track point and the second gives the row of the gain matrix for that point on the track. The gains are punched in a D format in columns 11-30, 31-50, and 51-70 of each card. NOTE: It is necessary to change the second control card of the source deck from // EXEC FORTCLG to // EXEC FORTCLGP when using this option.
36 - 40	IGAIN	I 5	This integer parameter controls the handling of gains in the MCSP. An integer "0" causes gains to be generated online for each MC run. An integer "1" generates gains on the first run with a noiseless track and uses these gains for all subsequent noisy tracks. An integer "2" causes gains to be read in by rows, 3 number per card. The data should be in D format in 20 column fields beginning in column 11.
41 - 80	LABEL	10 A 4	A 10-element alphanumeric array for run identification that appears at beginning of output

CARD 3

COL(S)	VARIABLE (ARRAY) NAME	FORMAT	DESCRIPTION
1 - 15	RR	D 15.0	Standard deviation of noise added to range data in yards. In MCSP this is added to .001· Range to provide a realistic figure
16-30	RB	D 15.0	Standard deviation of noise added to bearing data in radians
31-45	RE	D 15.0	Standard deviation of noise added to elevation data in radians
46 - 60	K1	D 15.0	Weighting constant for adaptive Q option which multiplies previous Q
61 - 75	K2	D 15.0	Weighting constant for adaptive Q option which multiplies new residual matrix

CARD 4

COL(S)	VARIABLE (ARRAY) NAME	FORMAT	DESCRIPTION
1 - 5	IDIR	I 5	An integer which specifies the rotation of the target track desired: 0 - no rotation 1 - 45° counterclockwise 2 - 90° counterclockwise 3 - 200.5° counterclockwise
6 - 10	INDEX	I 5	An integer controlling the addition of measurement noise to the track 0 - no noise 1 - add noise
11 - 15	IGUN	I 5	An integer "1" causes MCSP to calculate and output shell-at-target residuals
16 - 35	TRANSX	D 20.0	Desired translation of all X-axis data points in yards
36 - 55	TRANSY	D 20.0	Same for Y axis
56 - 75	TRANSZ	D 20.0	Same for Z axis

Card 5 begins the input section for matrices, and, therefore, the description given is for a typical card of the matrix with an indication of the number of cards that should be in the matrix. This section also depends upon the options that have been chosen above, and the possible combinations are discussed below.

1 - If gains are to be read in (IGAIN = 2) they are done so here under the format specified under the IGAIN = 2 description. There should be NN · N cards. (Next go to #1 below to read $\underline{\phi}$ matrix.)

The following optional matrices apply when gains are calculated by the MCSP.

2 - If the $\underline{R}$ matrix is to be computed by the MCSP and synthetic measurements for initialization are not requested (OPT $\neq$ 1) the following sequence applies.

(a) Read initial prediction covariance matrix, $\underline{P}(0/-1)$, an N X N matrix. Data is read with an 8D10.5 format, thus there are N cards for 6th-order and 2·N cards for 9th-order with the 9th element of each matrix row on a card by itself in the first 10 columns following the card with the first 8 elements punched.

(b) Read initial matrix of random forcing covariances, $E[\underline{w}\ \underline{w}^T]$, a (3 X 3) matrix with the same input format as (a).

(c) Read the (3 X N) $\underline{\Gamma}$ matrix with the same format as (a).

3 - If $\underline{R}$ is to be a constant matrix, the instructions in 2 above apply with the addition of reading the (3 X 3) $\underline{R}$ matrix between (a) and (b). The format is the same as 2 (a) above.

4 - If gains are calculated online and synthetic measurements are desired for filter initialization (OPT = 1) it is not necessary to provide a prediction covariance matrix. Therefore, in this case either the matrix of expected forcing covariances or a covariance matrix of measurement noise should appear first in this section of optional matrices.

These optional matrices having been arranged correctly, the next two matrices follow for all four possibilities.

1 - The N X N plant dynamics matrix, $\underline{\phi}$, is read. Again, the format follows that described in case 2 (a) above.

2 - The 3 X N (for a three-dimensional problem) measurement matrix, $\underline{H}$, is read next with the same format as for $\underline{\phi}$. There should be either six or 12 cards depending on the order of the filter.

The next major group of cards pertains to the track data. The format for all track data is to read the XYZ components for a given point with a 3D20.0 specification for fields beginning in column nine. If true velocities are available and to be read they should be punched according to the same format, and the velocity data card for a given track point should follow immediately after the corresponding position data card and prior to the next point's position data card. There should be either NN or 2 · NN cards in this segment depending on whether velocity data is available or not.

Following the track data, if IGUN = 1 implying that the shell-at-target calculations are desired, should appear the shell flight times punched one per card in the first 20 columns. There should be NN of these cards, and the format statement controlling their input is D20.0.

The last two cards possibly needed are for the initial conditions for the state vector $\hat{\underline{x}}(0/-1)$, called for when BEGIN = 1. The same format is followed as for the track data with the position data on the first card and velocity data on the second. In the case of a 9th-order filter the acceleration initial conditions are automatically set to zero.

This concludes the discussion of input data organization, and a complete sample data set is included in an example in Appendix E.

C. OUTPUT

Most of the output provided by the MCSP is self-explanatory, but a summary is included here with some additional insight into what the output actually means and a few peculiarities that the user should be aware of. A complete sample output is included with the example in Appendix E.

The first general output segment includes specification of some of the input parameters as well as the initial condition matrices. The matrices printed depend on what is relevant to the run requested. For example, if gains are read in, no prediction covariance matrix or random forcing matrix is provided. Also, if the adaptive $\underline{Q}$ method is used the actual initial state excitation matrix is printed instead of the output for constant-$\underline{Q}$ filter runs which includes the variances of random forcing that are provided as input data.

The next output segment, if requested, is the performance table showing the shell-at-target residuals. Because it takes several seconds for a shell to reach the target after filtering has been initiated, the output shows no residuals until this flight time has been satisfied. In the case of the track designed for the MCSP with no track translation, this amounts to 36 time increments. Also printed is a figure of merit for the given filtering scheme which is the average shell-at-target miss distance for those track points where it is applicable.

Following the gunfire figure of merit is a filter performance factor representing the rms error in filter position estimates averaged over the entire track. If true velocity data is available, a similar performance factor is printed which represents the rms error in filter velocity estimates averaged over the entire track.

The succeeding output table displays actual filter estimation accuracy by showing the mean and variance of the error between the filter estimate and the true track data. If velocity data is not available, the output for the velocity statistics presents the mean and variance of the estimates themselves.

The last and optional set of output is the plots requested on the first data card. These are produced by the routine PLOTP with one plot per page and headings already provided for the three-dimensional XYZ coordinate system.

V. SUMMARY OF REPRESENTATIVE SIMULATION RUNS

A. DESCRIPTION OF RUNS AND NUMERICAL RESULTS

This section presents the results of the simulations using some of the various combinations of options available in the MCSP. Rather than include the extensive outputs of the simulations only relative figures of merit for the shell-at-target accuracy calculations and filter performance factors are listed in Table II to provide some insight into the effects of possible combinations of selected options. In addition, results are included that were obtained using the MK-86 filter simulation developed in [5].

Descriptions of the various runs follow that include mention of the unique options used for each run. Unless otherwise specified the controls and initial conditions used in all runs include:

(a) 100 Monte-Carlo runs,

(b) $\underline{\phi}$, $\underline{\Gamma}$ and $\underline{H}$ are the same as described previously for 6th and 9th-order filters,

(c) $\underline{P}(k/k-1)$ is "large" (1.D05 for all diagonal terms),

(d) Covariances of random forcing input matrix = 0,

(e) Standard deviations of Gaussian noise added to track are $\sigma_R = 5 + .001 \times \text{Range}$, $\sigma_B = .002$ radians, $\sigma_E = .002$ radians,

(f) No rotation or translation of track data,

(g) Initialization of filter by letting the first observation define the position states in $\hat{\underline{x}}(0/-1)$.

Reference made to "Constant Q" in the descriptions signifies that the covariances of random forcing used for defining the constant $\underline{Q}$ matrix are those calculated by successive differences in the track generating program. It should be kept in mind that this process is based on the analysis of a noiseless track, and the use of the covariances thus obtained are inherently small for the noisy-track simulations with subsequent estimator error.

Run Number	Order of Filter	Special Features
1	6	1 Monte-Carlo run No additive noise on track
2	6	IGAIN = 0 (gains calculated every run)
3	6	IGAIN = 1 (gains calculated on first run with noiseless track)
4	6	IGAIN = 1 Constant $\underline{Q}$ (values listed in Appendix B)
5	6	Same as 4 OPT = 1 (synthetic initialization) $\underline{P}(k/k-1) = 0$
6	6	Same as 4 BEGIN = 1 ($\hat{\underline{x}}(0/-1)$ is set to 0)
7	6	IGAIN = 0 Adaptive $\underline{Q}$ with $K_1 = .1$, $K_2 = .9$
8	6	IGAIN = 0 Adaptive $\underline{Q}$ with $K_1 = .5$, $K_2 = .5$
9	6	IGAIN = 0 Adaptive $\underline{Q}$ with $K_1 = .9$, $K_2 = .1$
10	6	IGAIN = 0 Adaptive $\underline{Q}$ with $K_1 = .1$, $K_2 = .9$ IDIR = 1 (45° rotation of track)

Run Number	Order of Filter	Special Features
11	6	IGAIN = 0 Adaptive $\underline{Q}$ with K_1 = .5, K_2 = .5 IDIR = 1
12	6	IGAIN = 0 Adaptive $\underline{Q}$ with K_1 = .9, K_2 = .1 IDIR = 1
13	9	1 Monte-Carlo run No additive noise on track
14	9	IGAIN = 1
15	9	IGAIN = 1 Constant $\underline{Q}$
16	9	IGAIN = 0 Adaptive $\underline{Q}$ with K_1 = .1, K_2 = .9
17	9	IGAIN = 0 Adaptive $\underline{Q}$ with K_1 = .5, K_2 = .5
18	9	IGAIN = 0 Adaptive $\underline{Q}$ with K_1 = .9, K_2 = .1
19	9	Same as 18 Only position and velocity estimates used for shell-at-target accuracy calculations
20	6-9	Model of MK-86 estimator No additive noise on track 1 Monte-Carlo run

The second column of Table II lists shell-at-target figures of merit normalized to the results obtained with the MK-86 estimator. This is done to indicate relative performance against what must be considered a severe test of the filters studied. Due to compatability problems between the MK-86 estimator and the MCSP the results for run 20 are based on a noiseless track with one Monte-Carlo simulation. This produces optimal statistics since noise on the track degrades filter performance, and the comparison between the MK-86

estimator and the other filters in the study is slanted in favor of the former.

Additional insight into the potential success of the fire control system estimators in generating accurate firing orders is reflected in Table III where summaries of the numbers of hits within several radial miss distances of the target are tabulated for the better filters. Again, these figures would undoubtedly improve given a less maneuvering target, but it is worthwhile to know the possible consequences of a threat as reflected in the modeled track. The numbers shown are out of a total of 197 shells fired in the simulation.

B. EVALUATION OF RESULTS

An analysis of the numerical results presented in Table II reveals several items of interest. In this section the influence of such factors as the method of determining and using gains, initialization, type of Q matrix used, and direction of approach are empirically compared, and possible explanations for the trends shown are discussed.

The motivation for runs 1-3 and 13-14 was to check the reliability of the Monte-Carlo statistics and to compare the various techniques for determining and using gains. With noise of mean zero added to the track in runs 2, 3, and 14 the filter estimate errors should have been consistent with those generated in runs 1 and 13 (single run, no additive noise simulations). In addition, runs 2 and 3 were executed

TABLE II

ESTIMATOR PERFORMANCE TABLE

Run Number	Normalized RMS Shell-at-Target Miss Distance	RMS Mean Filter Position Estimation Error - yds	RMS Mean Filter Velocity Estimation Error - yds/sec
1	5.49	463.83	79.02
2	5.48	463.67	78.99
3	5.48	463.80	78.99
4	0.93	5.19	11.22
5	0.94	5.29	11.42
6	0.93	5.22	11.22
7	2.40	8.48	58.32
8	2.06	5.88	47.92
9	0.96	2.06	16.34
10	2.40	8.91	61.18
11	2.06	6.12	50.82
12	0.96	2.12	17.46
13	3.44	190.22	50.13
14	3.46	190.32	50.26
15	1.74	1.79	6.63
16	1.62	7.13	32.74
17	1.46	4.35	26.09
18	1.19	1.59	10.25
19	0.79	1.59	10.25
20	1.00	6.30	10.43

TABLE III

NUMBER OF HITS WITHIN SEVERAL MISS DISTANCES FOR SELECTED FILTERS

Run	Description	15	20	25	30
4	6th order Constant <u>Q</u>	3	6	10	16
9	6th order Adaptive <u>Q</u> $K_1 = .9, \bar{K}_2 = .1$	0	0	0	0
15	9th order Constant <u>Q</u>	35	41	44	52
18	9th order Adaptive <u>Q</u> $k_1 = .9, \bar{K}_2 = .1$	18	30	40	43
19	Same as 18 Accel. terms not used	0	6	13	20
20	MK-86 Estimator	0	3	6	12

to test the reliability of using gains calculated on the first run having a noiseless track against the results of a simulation having gains calculated on every run. As can be seen in Table II both tests showed desired results--the statistics for 100 runs with noisy tracks gave the expected mean errors, and the gain calculations on the first run having a noiseless track provided reliable results if used for all other runs having noisy tracks. It should be noted that calculating and using gains in this manner is only possible when $\underline{Q}$ is a constant matrix. Attempts to use IGAIN = 1 with adaptive $\underline{Q}$ simulations resulted in major discrepancies because the noise on the track is a major factor influencing the magnitude of the $\underline{Q}$ matrix from point to point, and the noiseless track did not provide similar residuals.

Runs 4-6 were designed to evaluate the various initialization methods, and the results show only minor variation.

A most interesting trend in the application of the adaptive $\underline{Q}$ option is revealed by runs 7-12 and 16-18. It is apparent that for the track used it is beneficial to provide some damping in the form of a high K_1/K_2 ratio. A plausible explanation is that additive noise can cause undesirable residual variation and subsequent erratic $\underline{Q}$ matrices if allowed to too heavily weight the updated $\underline{Q}$ matrix. Selecting the correct ratio shows obvious improvement in filter performance and provides better performance factors in Table II than those for the constant Q runs with 6th and

9th order filters. However, the numbers of hits within the several radii shown in Table III do not show the same pattern, and this suggests that if the requirement to have a few shells within a small radius is more important than having a lot of shells within a larger radius, then a less responsive $\underline{Q}$ matrix calculating technique is needed. Because in most airborne filtering situations the random forcing on the target is unknown, the successful implementation of an adaptive technique seems to be the only way to deal with all possible situations.

The rotation of the track in runs 10-12 was provided to check that option and confirm that the trends shown are not functions of the direction of approach of the target.

A final area of comparison is between the 6th and 9th order filters with similar options used. As expected the 9th order filter provided smaller mean filter estimation errors because of its ability to exactly follow a constantly-accelerating target. Specific runs for comparison are 4 and 15 for the constant $\underline{Q}$ runs and 9 and 18 for the best of the adaptive $\underline{Q}$ runs. As shown in Table II the two methods for determining $\underline{Q}$ provide similar results for the 6th-order filter with the constant $\underline{Q}$ method producing slightly smaller filter estimation errors. In the case of the 9th-order filter the adaptive $\underline{Q}$ technique provides superior position estimates, but the constant $\underline{Q}$ method is superior in generating velocity estimates.

A most noteworthy incompatibility is the contradiction shown for shell-at-target miss distances between 6th and 9th-order estimators. From Table II it is suggested that the 6th-order filters offer superior performance in minimizing the miss distances, but the actual superiority belongs to the 9th-order filters as evidenced by Table III. The discrepancy can be explained by the fact that the 6th-order filters consistently provide miss distances that are in the range from 50 to 150 yards but fail to reduce this substantially whereas the 9th-order filters do provide these minimum miss distances but are substantially worse at ranges greater than 3000 yards due to the effects of acceleration estimation errors. At great distances these acceleration estimation errors are important due to the larger flight times of the shells.

Run 19 was included as a special study to determine the shell-at-target accuracy for the filter in run 18 when only position and velocity estimates were used for prediction. Some internal changes were required in the MCSP to obtain these results that are intended to reinforce the comments made above.

The results for run 20 present the performance of the existing system against the scenario of the MCSP track. This is a combination 6th-9th order filter using only position and velocity estimates to generate predicted target location for determining gunfire orders.

VI. CONCLUSIONS AND RECOMMENDATIONS

Judging from the results presented in the previous section the techniques and derivations presented in this thesis have been shown to be applicable to airborne tracking problems with possible results superior to those obtained using present estimators. The use of a Monte-Carlo simulator has been beneficial in insuring that the results obtained are statistically legitimate for the scenario studied. Also, the capacity to vary the implementation of the many options available has made it possible to determine those factors that offer the greatest potential for improved estimator performance.

While the results presented in this thesis are encouraging, the scope of the study must be reiterated in order to avoid drawing too broad conclusions regarding the applicability to aircraft tracking problems. The track developed and used was selected because of its common use today in typical aircraft tactics, but much additional analysis would be necessary to categorically accept the relative capacities of the several filters presented herein. It is the adherence to such an analytical approach that can insure that the filters acquired for modern gunfire control systems are capable of handling the threats they are expected to experience.

The results have shown that given a complex target approach, basically simple filters are capable of surprising results if reinforced by techniques designed specifically for the model scenario. The online generation of the covariance matrix of measurement noise and adaptive covariance matrices of random forcing are examples that contribute to minimizing filter estimation errors. These features are, of course, dependent upon the availability of computation capacity, but with the advent of smaller, faster, and less expensive computers the restrictions imposed are becoming more liberal all the time.

This thesis was intended to correlate some of the important concepts pertinent to airborne tracking filters into an easily used simulation program for current and future study. Some areas that appear to offer additional potential for filter improvement and simulation accuracy include continued study in the field of adaptive covariance matrices of random forcing, a feature seen to be most powerful in this thesis, further work in the area of computation time and storage requirements for implementing various filtering approaches, and refinement of the shell-at-target accuracy calculations. Furthermore, implementation of simulation-generated techniques into actual models for testing is mandatory to provide feedback for refining the simulators, thus saving investment in inferior systems due to acceptance of incorrect simulation results.

APPENDIX A

PROGRAM LISTING FOR MONTE-CARLO RUN REQUIREMENT EVALUATION

The program written for analyzing the existing generators of normally distributed random numbers is included in this appendix. It provides statistics for empirical mean and standard deviation within 5, 10, 15, 20, and 25 percent of the desired standard deviation for ensembles taken a variable number of times.

In the program the following variables can be altered to obtain additional statistics describing expected results in Monte-Carlo simulations

ITER - The number of samples in each ensemble

N - The number of ensembles determined by data cards using an I5 format in the first five columns.

The program is written to terminate when N = 5000, but this can be revised by referring to the statement identified by NORM0094.

```
C
C      THIS PROGRAM ANALYZES THE AVAILABLE RANDCM NUMBER GENERATORS FOR
C        NORMALLY DISTRIBUTED RANDOM NUMBERS
C
      REAL MEAN,NUM1,NUM2,NUM3,NUM4,NUM5,NUM6,NUM7,NUM8,NUM9,NUM10
      DIMENSION MEAN(10000),STDDEV(10000)
C
C      ITER IS THE NUMBER OF TIMES THE REQUESTED NUMBER OF SAMPLES ARE
C        TAKEN TO ASSURE STEADY-STATE CONDITIONS
C
   10 ITER = 1000
C
C      ITER IS NUMBER OF ITERATIONS OF 'N' SAMPLES EACH
C
      READ(5,25) N
   25 FORMAT(I5)
C
C     XMEAN IS THE REQUESTED MEAN
C      XSTD IS THE REQUESTED STANDARD DEVIATION
C
      XMEAN = 0.0
      XSTD = 2.0
      WRITE(6,100) N,ITER,XMEAN,XSTD
  100 FORMAT(1H1,10X,'EXPERIMENTAL NORMAL DISTRIBUTION',/,
     1 11X,'PARAMETERS FOR ',I3,' SAMPLES TAKEN'     ,I5,' TIMES',/,11X,
     2 'WITH GIVEN MEAN=',F4.0,' STD. DEV.=',F4.0,///)
C
C      THE FOLLOWING CARD IS NOT PRESENT WHEN 'GAUSS' IS USED
C
      CALL OVFLOW
      KERNEL = 2568317
      DO 1000 I=1,ITER
      MEAN(I) = 0
      STDDEV(I) = 0
      DO 2000 J=1,N
C
C      WHEN GAUSS IS USED THE CALLING STATEMENT IS 'CALL GAUSS (KERNEL,
C        XSTD,XMEAN,TEMP)', AND THE FOLLOWING TWO CARDS ARE ELIMINATED
C
      CALL NORMAL(KERNEL,TEMP,1)
      TEMP = TEMP*XSTD
      TERM1 = FLOAT(J-1)/FLOAT(J)
      TERM2 = 1.0/FLOAT(J)
      MEAN(I) = MEAN(I)*TERM1+TERM2*TEMP
 2000 STDDEV(I) = STDDEV(I)+(TEMP-MEAN(I))**2
 1000 STDDEV(I) = SQRT(STDDEV(I)/(N-1))
   50 NUM1=0
      NUM2=0
```

```
      NUM3=0                                                            NORM0048
      NUM4=0                                                            NORM0049
      NUM5=0                                                            NORM0050
      NUM6=0                                                            NORM0051
      NUM7=0                                                            NORM0052
      NUM8=0                                                            NORM0053
      NUM9=0                                                            NORM0054
      NUM10=0                                                           NORM0055
      DO 300 I=1,ITER                                                   NORM0056
C     FIND RISK FOR 5,10,15,20.25 PERCENT OF STD DEV RANGE OF MEAN      NORM0057
      DIFF=ABS(MEAN(I)-XMEAN)                                           NORM0058
      IF(DIFF.LE.(.05*XSTD)) NUM1=NUM1+1                                NORM0059
      IF(DIFF.LE.(.1*XSTD)) NUM2 = NUM2+1                               NORM0060
      IF(DIFF.LE.(.15*XSTD)) NUM3=NUM3+1                                NORM0061
      IF(DIFF.LE.(.2*XSTD)) NUM4 =NUM4+1                                NORM0062
      IF(DIFF.LE.(.25*XSTD)) NUM5=NUM5+1                                NORM0063
C      FIND PRECISIONS FOR STANDARD DEVIATION ERRORS OF                 NORM0064
C       5,10,15,20,25 PERCENT                                           NORM0065
      DIFF=ABS(STDDEV(I)-XSTD)                                          NORM0066
      IF(DIFF.LE.(.05*XSTD)) NUM6=NUM6+1                                NORM0067
      IF(DIFF.LE.(.10*XSTD)) NUM7=NUM7+1                                NORM0068
      IF(DIFF.LE.(.15*XSTD)) NUM8=NUM8+1                                NORM0069
      IF(DIFF.LE.(.20*XSTD)) NUM9=NUM9+1                                NORM0070
      IF(DIFF.LE.(.25*XSTD)) NUM10=NUM10+1                              NORM0071
  300 CONTINUE                                                          NORM0072
      FITER=ITER                                                        NORM0073
      R5=NUM1/FITER*100.                                                NORM0074
      R10=NUM2/FITER*100.                                               NORM0075
      R15=NUM3/FITER*100.                                               NORM0076
      R20=NUM4/FITER*100.                                               NORM0077
      R25=NUM5/FITER*100.                                               NORM0078
      S5=NUM6/FITER*100.                                                NORM0079
      S10=NUM7/FITER*100.                                               NORM0080
      S15=NUM8/FITER*100.                                               NORM0081
      S20=NUM9/FITER*100.                                               NORM0082
      S25=NUM10/FITER*100.                                              NORM0083
      WRITE(6,400) R5,S5,R10,S10,R15,S15,R20,S20,R25,S25                NORM0084
  400 FORMAT(1H0,5X,'PERCENT',10X,'PERCENT OF MEANS',11X,               NORM0085
     1 'PERCENT OF SD',/,6X,'CRITERIA',11X,'WITHIN PERCENT',12X,        NORM0086
     2 'WITHIN PERCENT',/,26X,'CRITERIA OF',14X,'CRITERIA OF',/,        NORM0087
     3 27X,'DESIRED SD',16X,'DESIRED SD',/,6X,60('-'),//,               NORM0088
     4 10X,'5',17X,F7.3,19X,F7.3,/,                                     NORM0089
     5 9X,'10',17X,F7.3,19X,F7.3,/,                                     NORM0090
     6 9X,'15',17X,F7.3,19X,F7.3,/,                                     NORM0091
     7 9X,'20',17X,F7.3,19X,F7.3,/,                                     NORM0092
     8 9X,'25',17X,F7.3,19X,F7.3)                                       NORM0093
      IF(N.NE.5000) GO TO 10                                            NORM0094
      STOP                                                              NORM0095
```

APPENDIX B

TRACK GENERATING PROGRAM

The following program generates the track data used in the simulations conducted for comparison in this thesis. Both position and velocity data are produced for each of the 233 points on the track, and, in addition, the means and variances of accelerations and acceleration rates are calculated and averaged over the entire track to provide data used in the "Constant Q" simulations.

The results of the latter calculations follow, and it should be remembered that the validity of using the variances in defining a constant Q matrix is dependent on having enough sample points to insure accelerations and acceleration rates with the assumed mean of random forcing (zero).

Mean acceleration in X direction:	-1.675	yd/sec^2
" Y " :	-1.419	"
" Z " :	0.772	"
Variance of acceleration in X direction:	45.739	$(yd/sec^2)^2$
" Y " :	296.869	"
" Z " :	224.467	"
Mean acceleration rate in X direction:	-0.131	yd/sec^3
" Y " :	-0.351	"
" Z " :	0.207	"
Variance of acceleration rate in X direction:	106.984	$(yd/sec^3)^2$
Variance of acceleration rate in Y direction:	957.846	"
Variance of acceleration rate in Z direction:	4249.27	"

```
C
C     THIS PROGRAM GENERATES THE TRACK AND VELOCITY DATA AS WELL AS
C       GENERATING THE VARIANCES USED FOR FINDING A CONSTANT,
C       OPTIMAL, Q MATRIX
C
      IMPLICIT REAL*8 (A-H,L-Z)
      DIMENSION X(233),Y(233),Z(233)
     1 ,XVEL(233),YVEL(233),ZVEL(233)
     2 ,VSTOR(233,3),ASTOR(233,3),ARSTOR(233,3)
      DEGRAD = 57.2957795131D0
      X(1) = 17250.D0
      Y(1)= 0.D0
      Z(1)= 13000.D0
      XVEL(1) = -417.5D0
      YVEL(1) = 0.D0
      ZVEL(1) = 0.D0
      THETA = 80.D0/DEGRAD
      R= 6680.D0/THETA
      ANGLE = 0.D0
      DO 10 I=2,33
      ANGLE = ANGLE+1.25D0/DEGRAD
      ZTEMP = R*DCOS(ANGLE)
      XTEMP = R*DSIN(ANGLE)
      XVEL(I) = -417.5D0*DCOS(ANGLE)
      YVEL(I) = 0.D0
      ZVEL(I) = -417.5D0*DSIN(ANGLE)
      X(I) = X(1)-XTEMP
      Y(I) = 0.D0
   10 Z(I) = Z(1)-(R-ZTEMP)
C
C     THIS IS END OF INITIAL DIVE PHASE - BEGIN ACCELERATING DIVE
C
      SIN40 = DSIN(40.D0/DEGRAD)
      COS40 = DCOS(40.D0/DEGRAD)
      TIME = .25D0
      DIST = 7500.D0/SIN40
      VEL = (2.D0*DIST)/20.D0-417.5D0
      DVEL = (VEL-417.5D0)/80.D0
      OLDVEL = 417.5D0
      DO 20 I=34,113
      X(I) = X(I-1)-(OLDVEL*TIME+DVEL*TIME/2.D0)*COS40
      Y(I) = Y(I-1)
      Z(I) = Z(I-1)-(OLDVEL*TIME+DVEL*TIME/2.D0)*SIN40
      OLDVEL = OLDVEL+DVEL
      XVEL(I) = -OLDVEL*DCOS(40.D0/DEGRAD)
      YVEL(I) = 0.D0
   20 ZVEL(I) = -OLDVEL*DSIN(40.D0/DEGRAD)
C
```

```
C       THIS IS END OF ACCELERATING DIVE - BEGIN PULLOUT                  TRAC0048
C                                                                         TRAC0049
      R = 12.D0*VEL/THETA                                                 TRAC0050
      ANGLE = 0.D0                                                        TRAC0051
      ADEL = 40.D0/24.D0/DEGRAD                                           TRAC0052
      DO 30 I=114,137                                                     TRAC0053
      ANGLE = ANGLE+ADEL                                                  TRAC0054
      CPRIME = 2.D0*R*DSIN(ANGLE/2.D0)                                    TRAC0055
      PHI = 40.D0/DEGRAD - ANGLE/2.D0                                     TRAC0056
      XVEL(I) = -VEL*DCOS(40.D0/DEGRAD-ANGLE)                             TRAC0057
      YVEL(I) = 0.D0                                                      TRAC0058
      ZVEL(I) = -VEL*DSIN(40.D0/DEGRAD-ANGLE)                             TRAC0059
      X(I) = X(113)-CPRIME*DCOS(PHI)                                      TRAC0060
      Z(I) = Z(113)-CPRIME*DSIN(PHI)                                      TRAC0061
   30 Y(I) = Y(I-1)                                                       TRAC0062
C                                                                         TRAC0063
C       THIS IS END OF PULLOUT - BEGIN EVASIVE MANEUVER TO RIGHT OF 5 DEGTRAC0064
C                                                                         TRAC0065
      THETA = 5.D0/DEGRAD                                                 TRAC0066
      R = VEL /THETA                                                      TRAC0067
      ANGLE = 0.D0                                                        TRAC0068
      DO 40 I=138,141                                                     TRAC0069
      ANGLE = ANGLE+1.25D0/DEGRAD                                         TRAC0070
      XVEL(I) = -VEL*DCOS(ANGLE)                                          TRAC0071
      YVEL(I) = VEL*DSIN(ANGLE)                                           TRAC0072
      ZVEL(I) = 0.D0                                                      TRAC0073
      X(I) = X(137)-R*DSIN(ANGLE)                                         TRAC0074
      Y(I) = Y(137)+R-R*DCOS(ANGLE)                                       TRAC0075
   40 Z(I) = Z(137)                                                       TRAC0076
C                                                                         TRAC0077
C       THIS IS END OF STARBOARD MANEUVER - REPEAT TO PORT                TRAC0078
C                                                                         TRAC0079
      ANGLE = 0.D0                                                        TRAC0080
      DO 50 I=142,145                                                     TRAC0081
      ANGLE = ANGLE+1.25D0/DEGRAD                                         TRAC0082
      CPRIME = 2.D0*R*DSIN(ANGLE/2.D0)                                    TRAC0083
      PHI = 85.D0/DEGRAD+ANGLE/2.D0                                       TRAC0084
      XVEL(I) = -VEL*DCOS(5.D0/DEGRAD-ANGLE)                              TRAC0085
      YVEL(I) = VEL*DSIN(5.D0/DEGRAD-ANGLE)                               TRAC0086
      ZVEL(I) = 0.D0                                                      TRAC0087
      X(I) = X(141)-CPRIME*DSIN(PHI)                                      TRAC0088
      Y(I) = Y(141)+CPRIME*DCOS(PHI)                                      TRAC0089
   50 Z(I) = Z(141)                                                       TRAC0090
C                                                                         TRAC0091
C       THIS IS END OF PORT MANEUVER - EXECUTE ANOTHER SIMILAR PORT MAN.  TRAC0092
C                                                                         TRAC0093
      ANGLE = 0.D0                                                        TRAC0094
      DO 60 I=146,149                                                     TRAC0095
```

```
      ANGLE = ANGLE+1.25D0/DEGRAD                                      TRAC0096
      XVEL(I) = -VEL*DCOS(ANGLE)                                       TRAC0097
      YVEL(I) = -VEL*DSIN(ANGLE)                                       TRAC0098
      ZVEL(I) = 0.D0                                                   TRAC0099
      X(I) = X(145)-R*DSIN(ANGLE)                                      TRAC0100
      Y(I) = Y(145)-(R-R*DCOS(ANGLE))                                  TRAC0101
   60 Z(I) = Z(145)                                                    TRAC0102
C                                                                      TRAC0103
C      THIS IS END OF PORT MANEUVER - PLANE NOW HAS PASSED OVER SHIP - TRAC0104
C      EXECUTE 3.5 G PORT TURN FOR 2 SECS AND BEGIN 10 DEG CLIMB       TRAC0105
C                                                                      TRAC0106
      SIN10 = DSIN(10.D0/DEGRAD)                                       TRAC0107
      COS10 = DCOS(10.D0/DEGRAD)                                       TRAC0108
      ANGLE = 0.D0                                                     TRAC0109
      GRAV = 32.2D0                                                    TRAC0110
      TAU = 5.D0/DEGRAD                                                TRAC0111
      R = VEL**2/(3.5D0*GRAV)                                          TRAC0112
      S = VEL *2.D0                                                    TRAC0113
      THETA = S/R                                                      TRAC0114
      DELTH = THETA/8.D0                                               TRAC0115
      DO 70 I=150,157                                                  TRAC0116
      ANGLE = ANGLE+DELTH                                              TRAC0117
      C = 2.D0*R*DSIN(ANGLE/2.D0)                                      TRAC0118
      PHI = TAU+ANGLE/2.D0                                             TRAC0119
      XVEL(I) = -VEL*COS10*DCOS(5.D0/DEGRAD+ANGLE)                     TRAC0120
      YVEL(I) = -VEL*COS10*DSIN(5.D0/DEGRAD+ANGLE)                     TRAC0121
      ZVEL(I) = VEL*SIN10                                              TRAC0122
      X(I) = X(149)-C*DCOS(PHI)                                        TRAC0123
      Y(I) = Y(149)-C*DSIN(PHI)                                        TRAC0124
   70 Z(I) = Z(149)+C*SIN10                                            TRAC0125
      TAU = TAU+THETA                                                  TRAC0126
C                                                                      TRAC0127
C      END OF TURN - BEGIN 3.5 G STARBOARD TURN FOR 2 SECS.            TRAC0128
C                                                                      TRAC0129
      ANGLE = 0.D0                                                     TRAC0130
      DO 80 I=158,165                                                  TRAC0131
      ANGLE = ANGLE+DELTH                                              TRAC0132
      C = 2.D0*R*DSIN(ANGLE/2.D0)                                      TRAC0133
      PHI = 90.D0/DEGRAD-TAU+ANGLE/2.D0                                TRAC0134
      XVEL(I) = -VEL*COS10*DCOS(TAU-ANGLE)                             TRAC0135
      YVEL(I) = -VEL*COS10*DSIN(TAU-ANGLE)                             TRAC0136
      ZVEL(I) = VEL*SIN10                                              TRAC0137
      X(I) = X(157)-C*DSIN(PHI)                                        TRAC0138
      Y(I) = Y(157)-C*DCOS(PHI)                                        TRAC0139
   80 Z(I) = Z(157)+C*SIN10                                            TRAC0140
      TAU = TAU-THETA                                                  TRAC0141
C                                                                      TRAC0142
C      END OF TURN - BEGIN 3.0 G PORT TURN FOR 2 SECS.                 TRAC0143
```

```
C                                                                    TRAC0144
      ANGLE = 0.D0                                                   TRAC0145
      R = VEL**2/(3.D0*GRAV)                                         TRAC0146
      THETA = S/R                                                    TRAC0147
      DELTH = THETA/8.D0                                             TRAC0148
      DO 90 I=166,173                                                TRAC0149
      ANGLE = ANGLE+DELTH                                            TRAC0150
      C = 2.D0*R*DSIN(ANGLE/2.D0)                                    TRAC0151
      PHI = TAU+ANGLE/2.D0                                           TRAC0152
      XVEL(I) = -VEL*COS10*DCOS(TAU+ANGLE)                           TRAC0153
      YVEL(I) = -VEL*COS10*DSIN(TAU+ANGLE)                           TRAC0154
      ZVEL(I) = VEL*SIN10                                            TRAC0155
      X(I) = X(165)-C*DCOS(PHI)                                      TRAC0156
      Y(I) = Y(165)-C*DSIN(PHI)                                      TRAC0157
   90 Z(I) = Z(165)+C*SIN10                                          TRAC0158
      TAU = TAU+THETA                                                TRAC0159
C                                                                    TRAC0160
C      END OF TURN - BEGIN 3.0 G STARBOARD TURN FOR 2 SECS.          TRAC0161
C                                                                    TRAC0162
      ANGLE = 0.D0                                                   TRAC0163
      DO 100 I=174,181                                               TRAC0164
      ANGLE = ANGLE+DELTH                                            TRAC0165
      C = 2.D0*R*DSIN(ANGLE/2.D0)                                    TRAC0166
      PHI = 90.D0/DEGRAD-TAU+ANGLE/2.D0                              TRAC0167
      XVEL(I) = -VEL*COS10*DCOS(TAU-ANGLE)                           TRAC0168
      YVEL(I) = -VEL*COS10*DSIN(TAU-ANGLE)                           TRAC0169
      ZVEL(I) = VEL*SIN10                                            TRAC0170
      X(I) = X(173)-C*DSIN(PHI)                                      TRAC0171
      Y(I) = Y(173)-C*DCOS(PHI)                                      TRAC0172
  100 Z(I) = Z(173)+C*SIN10                                          TRAC0173
      TAU = TAU-THETA                                                TRAC0174
C                                                                    TRAC0175
C      END OF TURN - BEGIN 3.0 G PORT TURN FOR 3 SECS.               TRAC0176
C                                                                    TRAC0177
      ANGLE = 0.D0                                                   TRAC0178
      S = VEL*3.D0                                                   TRAC0179
      THETA = S/R                                                    TRAC0180
      DELTH = THETA/12.D0                                            TRAC0181
      DO 110 I=182,193                                               TRAC0182
      ANGLE = ANGLE+DELTH                                            TRAC0183
      C = 2.D0*R*DSIN(ANGLE/2.D0)                                    TRAC0184
      PHI = TAU+ANGLE/2.D0                                           TRAC0185
      XVEL(I) = -VEL*COS10*DCOS(TAU+ANGLE)                           TRAC0186
      YVEL(I) = -VEL*COS10*DSIN(TAU+ANGLE)                           TRAC0187
      ZVEL(I) = VEL*SIN10                                            TRAC0188
      X(I) = X(181)-C*DCOS(PHI)                                      TRAC0189
      Y(I) = Y(181)-C*DSIN(PHI)                                      TRAC0190
  110 Z(I) = Z(181)+C*SIN10                                          TRAC0191
```

```
      TAU = TAU+THETA                                                   TRAC0192
C                                                                       TRAC0193
C      END OF TURN - BEGIN 2.5 G STARBOARD TURN FOR 3 SECS.            TRAC0194
C                                                                       TRAC0195
      ANGLE = 0.D0                                                      TRAC0196
      R = VEL**2/(2.5D0*GRAV)                                           TRAC0197
      THETA = S/R                                                       TRAC0198
      DELTH = THETA/12.D0                                               TRAC0199
      DO 120 I=194,205                                                  TRAC0200
      ANGLE = ANGLE+DELTH                                               TRAC0201
      C = 2.D0*R*DSIN(ANGLE/2.D0)                                       TRAC0202
      PHI = 90.D0/DEGRAD - TAU+ANGLE/2.D0                               TRAC0203
      XVEL(I) = -VEL*COS10*DCOS(TAU-ANGLE)                              TRAC0204
      YVEL(I) = -VEL*COS10*DSIN(TAU-ANGLE)                              TRAC0205
      ZVEL(I) = VEL*SIN10                                               TRAC0206
      X(I) = X(193)-C*DSIN(PHI)                                         TRAC0207
      Y(I) = Y(193)-C*DCOS(PHI)                                         TRAC0208
  120 Z(I) = Z(193) + C*SIN10                                           TRAC0209
      TAU = TAU-THETA                                                   TRAC0210
C                                                                       TRAC0211
C      END OF TURN - BEGIN 2.5 G PORT TURN FOR 4 SECS.                 TRAC0212
C                                                                       TRAC0213
      ANGLE = 0.D0                                                      TRAC0214
      S = VEL*4.D0                                                      TRAC0215
      THETA = S/R                                                       TRAC0216
      DELTH = THETA/16.D0                                               TRAC0217
      DO 130 I=206,221                                                  TRAC0218
      ANGLE = ANGLE+DELTH                                               TRAC0219
      C = 2.D0*R*DSIN(ANGLE/2.D0)                                       TRAC0220
      PHI = TAU+ANGLE/2.D0                                              TRAC0221
      XVEL(I) = -VEL*COS10*DCOS(TAU+ANGLE)                              TRAC0222
      YVEL(I) = -VEL*COS10*DSIN(TAU+ANGLE)                              TRAC0223
      ZVEL(I) = VEL*SIN10                                               TRAC0224
      X(I) = X(205)-C*DCOS(PHI)                                         TRAC0225
      Y(I) = Y(205)-C*DSIN(PHI)                                         TRAC0226
  130 Z(I) = Z(205)+C*SIN10                                             TRAC0227
      TAU = TAU+THETA                                                   TRAC0228
C                                                                       TRAC0229
C      END OF TURN - BEGIN 2.0 G STARBOARD TURN FOR 3 SECS.            TRAC0230
C                                                                       TRAC0231
      ANGLE = 0.D0                                                      TRAC0232
      S = VEL*3.D0                                                      TRAC0233
      R = VEL**2/(2.D0*GRAV)                                            TRAC0234
      THETA = S/R                                                       TRAC0235
      DELTH = THETA/12.D0                                               TRAC0236
      DO 140 I=222,233                                                  TRAC0237
      ANGLE = ANGLE+DELTH                                               TRAC0238
      C = 2.D0*R*DSIN(ANGLE/2.D0)                                       TRAC0239
```

```
      PHI = 90.D0/DEGRAD-TAU+ANGLE/2.D0                                 TRAC0240
      XVEL(I) = -VEL*COS10*DCOS(TAU-ANGLE)                              TRAC0241
      YVEL(I) = -VEL*COS10*DSIN(TAU-ANGLE)                              TRAC0242
      ZVEL(I) = VEL*SIN10                                               TRAC0243
      X(I) = X(221)-C*DSIN(PHI)                                         TRAC0244
      Y(I) = Y(221)-C*DCOS(PHI)                                         TRAC0245
  140 Z(I) = Z(221)+C*SIN10                                             TRAC0246
      DO 150 I=1,233                                                    TRAC0247
C                                                                       TRAC0248
C      CHANGE POSITION AND VELOCITY DATA TO YARDS AND YARDS/SEC         TRAC0249
C        INSTEAD OF FEET AND FEET/SEC                                   TRAC0250
C                                                                       TRAC0251
      X(I) = X(I)/3.D0                                                  TRAC0252
      Y(I) = Y(I)/3.D0                                                  TRAC0253
      Z(I) = Z(I)/3.D0                                                  TRAC0254
      VEL = DSQRT(XVEL(I)**2+YVEL(I)**2+ZVEL(I)**2)                     TRAC0255
      XVEL(I) = XVEL(I)/3.D0                                            TRAC0256
      YVEL(I) = YVEL(I)/3.D0                                            TRAC0257
      ZVEL(I) = ZVEL(I)/3.D0                                            TRAC0258
C                                                                       TRAC0259
C      THE POSITION/VELOCITY VALUES ARE PUNCHED ON CARD PAIRS WITH      TRAC0260
C        THE 3 POSITION VALUES ON THE FIRST CARD AND THE 3 VELOCITY     TRAC0261
C        VALUES ON THE SECOND                                           TRAC0262
C                                                                       TRAC0263
      WRITE(7,4778) I,X(I),Y(I),Z(I),I,XVEL(I),YVEL(I),ZVEL(I)          TRAC0264
  150 WRITE(6,4777) I,X(I),Y(I),Z(I),I,XVEL(I),YVEL(I),ZVEL(I),VEL      TRAC0265
 4778 FORMAT(5X,I3,3D20.10,/,5X,I3,3D20.10)                             TRAC0266
 4777 FORMAT(5X,I3,3D20.10,/,5X,I3,3D20.10,10X,D20.6)                   TRAC0267
C                                                                       TRAC0268
C      THE REMAINDER OF THE PROGRAM GENERATES VARIANCES OF ACCELERATION TRAC0269
C        AND ACCELERATION RATES BY TAKING SUCCESSIVE DIFFERENCES AND    TRAC0270
C        ANALYZING OVER THE ENTIRE TRACK                                TRAC0271
C                                                                       TRAC0272
      DO 160 I=1,232                                                    TRAC0273
      VSTOR(I,1) =      (X(I+1)-X(I))/.25D0                             TRAC0274
      VSTOR(I,2) =      (Y(I+1)-Y(I))/.25D0                             TRAC0275
  160 VSTOR(I,3) =      (Z(I+1)-Z(I))/.25D0                             TRAC0276
      DO 170 I=1,231                                                    TRAC0277
      ASTOR(I,1) =      (VSTOR(I+1,1)-VSTOR(I,1))/.25D0                 TRAC0278
      ASTOR(I,2) =      (VSTOR(I+1,2)-VSTOR(I,2))/.25D0                 TRAC0279
  170 ASTOR(I,3) =      (VSTOR(I+1,3)-VSTOR(I,3))/.25D0                 TRAC0280
      DO 180 I=1,230                                                    TRAC0281
      ARSTOR(I,1) =      (ASTOR(I+1,1)-ASTOR(I,1))/.25D0                TRAC0282
      ARSTOR(I,2) =      (ASTOR(I+1,2)-ASTOR(I,2))/.25D0                TRAC0283
  180 ARSTOR(I,3) =      (ASTOR(I+1,3)-ASTOR(I,3))/.25D0                TRAC0284
      AMEANX=0.D0                                                       TRAC0285
      AMEANY = 0.D0                                                     TRAC0286
      AMEANZ = 0.D0                                                     TRAC0287
```

```
      AVARX = 0.D0                                                      TRAC0288
      AVARY = 0.D0                                                      TRAC0289
      AVARZ = 0.D0                                                      TRAC0290
      DO 190 I=1,231                                                    TRAC0291
      AMEANX = AMEANX+ASTOR(I,1)                                        TRAC0292
      AMEANY = AMEANY+ASTOR(I,2)                                        TRAC0293
      AMEANZ = AMEANZ+ASTOR(I,3)                                        TRAC0294
      AVARX = AVARX+ASTOR(I,1)**2                                       TRAC0295
      AVARY = AVARY+ASTOR(I,2)**2                                       TRAC0296
  190 AVARZ = AVARZ+ASTOR(I,3)**2                                       TRAC0297
      AMEANX = AMEANX/231.D0                                            TRAC0298
      AMEANY = AMEANY/231.D0                                            TRAC0299
      AMEANZ = AMEANZ/231.D0                                            TRAC0300
      AVARX = AVARX/231.D0-AMEANX**2                                    TRAC0301
      AVARY = AVARY/231.D0-AMEANY**2                                    TRAC0302
      AVARZ = AVARZ/231.D0-AMEANZ**2                                    TRAC0303
      WRITE(6,2719) AMEANX,AMEANY,AMEANZ,AVARX,AVARY,AVARZ              TRAC0304
 2719 FORMAT(1H1,9X,'ACC MEANX X=',D20.6,/,10X,'ACC MEAN Y=',D20.6,/,   TRAC0305
     1 10X,'ACC MEAN Z=',D20.6,/,10X,'VAR X=',D20.6,/,10X,'VAR Y=',D20.6TRAC0306
     3 ,/,10X,'VAR Z=',D20.6,///)                                       TRAC0307
      AMEANX = 0.D0                                                     TRAC0308
      AMEANY = 0.D0                                                     TRAC0309
      AMEANZ = 0.D0                                                     TRAC0310
      AVARX = 0.D0                                                      TRAC0311
      AVARY = 0.D0                                                      TRAC0312
      AVARZ = 0.D0                                                      TRAC0313
      DO 200 I=1,230                                                    TRAC0314
      AMEANX = AMEANX+ARSTOR(I,1)                                       TRAC0315
      AMEANY = AMEANY+ARSTOR(I,2)                                       TRAC0316
      AMEANZ = AMEANZ+ARSTOR(I,3)                                       TRAC0317
      AVARX = AVARX+ARSTOR(I,1)**2                                      TRAC0318
      AVARY = AVARY+ARSTOR(I,2)**2                                      TRAC0319
  200 AVARZ = AVARZ+ARSTOR(I,3)**2                                      TRAC0320
      AMEANX = AMEANX/230.D0                                            TRAC0321
      AMEANY = AMEANY/230.D0                                            TRAC0322
      AMEANZ = AMEANZ/230.D0                                            TRAC0323
      AVARX = AVARX/230.D0-AMEANX**2                                    TRAC0324
      AVARY = AVARY/230.D0-AMEANY**2                                    TRAC0325
      AVARZ = AVARZ/230.D0-AMEANZ**2                                    TRAC0326
      WRITE(6,2720) AMEANX,AMEANY,AMEANZ,AVARX,AVARY,AVARZ              TRAC0327
 2720 FORMAT (1H0,9X,'ACC RATE MEAN X=',D20.6,/,10X,'ACC RATE MEAN Y=', TRAC0328
     1 D20.6,/,10X,'ACC RATE MEAN Z=',D20.6,/,10X,'VAR X=',D20.6,/,10X, TRAC0329
     2 'VAR Y=',D20.6,/,10X,'VAR Z=',D20.6)                             TRAC0330
      STOP                                                              TRAC0331
      END                                                               TRAC0332
```

APPENDIX C

LISTING OF MONTE-CARLO SIMULATION PROGRAM (MSCP)

A complete listing of the Monte-Carlo Simulation Program is included herein. For user convenience the statement numbers are in ascending order allowing for easy location of designated statements, and the cards are labeled in the right margin for reference and ordering if required.

Variable and array descriptions appear at the beginning of the program with instructions for the input parameters immediately following each READ statement. Format statements controlling headings on the output plots are included in the program statements MCSP0600-MSCP0694.

```
C
C     THIS IS THE MONTE-CARLO SIMULATICN PROGRAM (MCSP)
C        AND CAN ANALYZE UP TO A 233 PCINT TRACK AS MANY AS
C        99999 TIMES TO OBTAIN TRUE MCNTE-CARLO STATISTICS
C        OF FILTER PERFORMANCE
C
      IMPLICIT REAL*8(Q,H,R,G,P,A,B,C,D,S,T)
      REAL*8 II,X,Y,Z,XTRUE,YTRUE,ZTRUE,EST,RTRUE,BTRUE,ETRUE,RT,BT,ET,X
     1T,YT,ZT,K1,K2
      REAL*4 SEEABS,SEEORD
      INTEGER*4 OPT,OPTNO,BEGIN,ADAPTQ
      DIMENSION PRED(9,9), LABEL(10), STATES(9,9), RADAR(9,9), XTRUE(233
     1), YTRUE(233), ZTRUE(233), QP(9,9), PKKM1P(9,9), RP(9,9), A(9,9),
     2B(9,9), GSTOR(9,3,233), TVELX(233), TVELY(233), TVELZ(233), RTRUE(
     3233), BTRUE(233), ETRUE(233), SEEABS(233), SEEORD(233), LOCK(14),
     4START(9),GAMMA(9,9),GAMMAT(9,9),HCLD(9,9)
      COMMON II(9,9),Q(9,9),H(9,9),R(9,9),G(9,9),PHI(9,9),PKK(9,9),PKKM1
     1(9,9),RMEAN(233,6),RSD(6),RESID(233,4),AHEAD(233),EST(9,233),PHIPR
     2M(9,9,40),PKSTOR(9,233)
C
C     DESCRIPTION OF ARRAYS FOLLOWS:
C        PRED - FILTER PREDICTION X(K+1/K)
C        LABEL - USERS DESCRIPTION OF RUN
C        STATES - FILTER ESTIMATE X(K/K)
C        RADAR - MEASUREMENTS
C        XTRUE - NOISELESS INPUT 'X' VALUES
C        YTRUE - NOISELESS INPUT 'Y' VALUES
C        ZTRUE - NOISELESS INPUT 'Z' VALUES
C        QP - COVARIANCE MATRIX OF STATE EXCITATICN
C        PKKM1P - PREDICTION COVARIANCE MATRIX
C        RP - COVARIANCE CF MEASUREMENT ERROR MATRIX
C        A - TEMPORARY MATRIX USED IN CALCULATING POWERS OF PHI MATRIX
C        B - TEMPORARY MATRIX USED IN CALCULATING POWERS OF PHI MATRIX
C        GSTOR - OPTIMUM GAIN MATRICES STORAGE ARRAY
C        TVELX - TRUE, NOISELESS 'X' VELOCITY
C        TVELY - TRUE, NOISELESS 'Y' VELOCITY
C        TVELZ - TRUE, NOISELESS 'Z' VELOCITY
C        RTRUE - TRUE RANGE CALCULATED FROM TRUE X,Y,Z
C        BTRUE - TRUE BEARING CALCULATED FROM TRUE X,Y,Z
C        ETRUE - TRUE ELEVATION CALCULATED FROM TRUE X,Y,Z
C
      CALL OVFLOW
      KERNL1=3790437
      KERNL2=30117221
      KERNL3=1993183
      DEGRAD=57.295779513100
C
C        II - IDENTITY MATRIX
```

```
C         Q - COVARIANCE MATRIX OF STATE EXCITATION (VARIABLE)          MCSP0048
C         GAMMA - MATRIX TO RELATE NON-DETERMINISTIC                     MCSP0049
C           FORCING TO STATES                                            MCSP0050
C         H - MEASUREMENT MATRIX                                         MCSP0051
C         R - COVARIANCE OF MEASUREMENT ERROR MATRIX (VARIABLE)          MCSP0052
C         G - OPTIMUM GAIN MATRIX (VARIABLE)                             MCSP0053
C         PHI - PHI MATRIX, I.E. PLANT DYNAMICS                          MCSP0054
C         PKK - ESTIMATION COVARIANCE MATRIX, P(K/K)                     MCSP0055
C         PKKM1 - PREDICTION COVARIANCE MATRIX, P(K/K-1) (VARIABLE)      MCSP0056
C         RMEAN - MEAN OF ESTIMATES, LATER MEAN OF ERROR                 MCSP0057
C         RSD - SUM OF POSITION RESIDUALS IN EACH DIRECTION              MCSP0058
C         RESID - POSITION RESIDUALS IN EACH DIRECTION PLUS TOTAL MISS   MCSP0059
C         AHEAD - TIMES FOR SHELL TO GET TO GIVEN POINT ON TRACK         MCSP0060
C         EST - STORAGE ARRAY FOR FILTER ESTIMATES                       MCSP0061
C         PHIPRM - STORAGE FOR POWERS OF PHI MATRIX UP TO 40             MCSP0062
C         PKSTOR - STORAGE FOR FILTER VARIANCES                          MCSP0063
C                                                                        MCSP0064
      DO 4 I=1,233                                                       MCSP0065
      DO 1 K=1,9                                                         MCSP0066
      PKSTOR(K,I)=0.D0                                                   MCSP0067
    1 EST(K,I)=0.D0                                                      MCSP0068
      DO 2 J=1,6                                                         MCSP0069
    2 RMEAN(I,J)=0.D0                                                    MCSP0070
      DO 3 L=1,4                                                         MCSP0071
    3 RESID(I,L)=0.D0                                                    MCSP0072
    4 CONTINUE                                                           MCSP0073
      DO 5 I=1,6                                                         MCSP0074
    5 RSD(I)=0.D0                                                        MCSP0075
C                                                                        MCSP0076
      READ (5,6) LOOP,KVEL,(LOOK(I),I=1,14),TIME                         MCSP0077
    6 FORMAT (2I10,10X,14I1,6X,D20.0)                                    MCSP0078
C                                                                        MCSP0079
C       LOOP IS THE NUMBER OF MONTE CARLO RUNS                           MCSP0080
C      KVEL = 1 WILL SET UP THE PROGRAM FOR READING IN TRUE VELOCITIES   MCSP0081
C         IN ADDITION TO TRACK POSITIONS                                 MCSP0082
C       ARRAY 'LOOK' DETERMINES PLOTS TO BE PRINTED                      MCSP0083
C         IF INPUT ARRAY ELEMENT IS NONZERO, THE FOLLOWING PLOTS         MCSP0084
C         WILL BE PRINTED                                                MCSP0085
C         LOOK(1):  MEAN MISS DISTANCE WHEN SHELL ARRIVES                MCSP0086
C         LOOK(2):  MEAN ERROR IN X POSITION                             MCSP0087
C         LOOK(3):  MEAN ERROR IN Y POSITION                             MCSP0088
C         LOOK(4):  MEAN ERROR IN Z POSITION                             MCSP0089
C         LOOK(5):  X POSITION VARIANCE                                  MCSP0090
C         LOOK(6):  Y POSITION VARIANCE                                  MCSP0091
C         LOOK(7):  Z POSITION VARIANCE                                  MCSP0092
C         LOOK(8):  MEAN VELOCITY ERROR IN X DIRECTION                   MCSP0093
C         LOOK(9):  MEAN VELOCITY ERROR IN Y DIRECTION                   MCSP0094
C         LOOK(10):  MEAN VELOCITY ERROR IN Z DIRECTION                  MCSP0095
```

```
C        LOOK(11):  X VELOCITY VARIANCE                                 MCSP0096
C        LOOK(12):  Y VELOCITY VARIANCE                                 MCSP0097
C        LOOK(13):  Z VELOCITY VARIANCE                                 MCSP0098
C        LOOK(14):  TARGET SPEED IN FT/SEC                              MCSP0099
C       'TIME' IS THE TIME BETWEEN MEASUREMENTS                         MCSP0100
C                                                                       MCSP0101
      READ (5,7) N,NN,OPT,BEGIN,IR,ADAPTQ,IG,IGAIN,(LABEL(I),I=1,10)    MCSP0102
    7 FORMAT (8I5,10A4)                                                 MCSP0103
C                                                                       MCSP0104
C       N --------- NUMBER OF STATES OR ORDER OF FILTER                 MCSP0105
C       NN -------- NUMBER OF STEPS - I.E. MEASUREMENTS                 MCSP0106
C       OPT ------- A '1' HERE WILL GENERATE SYNTHETIC MEASUREMENTS     MCSP0107
C       BEGIN ----- A '1' MEANS THAT THE STATE INITIAL CONDITION        MCSP0108
C         MATRIX X(0/-1) IS TO BE READ IN AT THE END OF THE DATA        MCSP0109
C       IR -------- A '0' WILL CAUSE THE PROGRAM TO GENERATE ONLINE     MCSP0110
C         R MATRICES.                                                   MCSP0111
C                   A '1' WILL READ IN THE CONSTANT R MATRIX            MCSP0112
C       ADAPTQ ---- A '1' WILL CAUSE THE Q MATRIX TO BE CHANGED         MCSP0113
C         ONLINE DEPENDING ON THE RESIDUAL BETWEEN PREDICTED AND        MCSP0114
C         ESTIMATED STATES.                                             MCSP0115
C       IG -------- A '1' MEANS PUNCH OUT GAINS - NOTE:  MUST ALSO CHANGEMCSP0116
C         2ND CONTROL CARD - // EXEC FORTCLG ... TO // EXEC FORTCLGP ... MCSP0117
C       IGAIN ----- A '0' WILL CAUSE GAINS TO BE GENERATED ONLINE FOR   MCSP0118
C         EACH MONTE CARLO RUN                                          MCSP0119
C                   A '1' WILL GENERATE GAINS ON THE FIRST RUN WITH A   MCSP0120
C         NOISELESS TRACK AND USE FOR ALL OTHER RUNS                    MCSP0121
C                   A '2' WILL CAUSE GAINS TO BE READ IN ACCORDING TO   MCSP0122
C         SUBROUTINE 'MREAD1'                                           MCSP0123
C                                                                       MCSP0124
      READ (5,8) RR,RB,RE,K1,K2                                         MCSP0125
    8 FORMAT (5D15.0)                                                   MCSP0126
C                                                                       MCSP0127
C       RR = RANGE STANDARD DEVIATION                                   MCSP0128
C       RB = BEARING STANDARD DEVIATION                                 MCSP0129
C       RE = ELEVATION STANDARD DEVIATION                               MCSP0130
C       K1 AND K2 ARE WEIGHTING CONSTANTS WHICH DETERMINE THE EXTENT TO MCSP0131
C         WHICH THE EFFECT OF THE LATEST RESIDUAL WILL ENTER INTO THE   MCSP0132
C         Q MATRIX. I.E. K1 X OLD Q + K2 X NEW Q                        MCSP0133
C                                                                       MCSP0134
      READ (5,9) IDIR,INDEX,IGUN,TRANSX,TRANSY,TRANSZ                   MCSP0135
    9 FORMAT (3I5,3D20.0)                                               MCSP0136
C                                                                       MCSP0137
C       INDEX = 0 : NO NOISE ADDED ----- INDEX = 1 : NOISE ADDED        MCSP0138
C       IDIR DETERMINES DIRECTION OF APPROACH                           MCSP0139
C         0 DEGS. IS RIGHT BEAM AND ANGLES INCREASE COUNTERCLOCKWISE    MCSP0140
C       IDIR = 0   ANGLE = 0 DEGS.                                      MCSP0141
C       IDIR = 1   ANGLE = 45 DEGS.                                     MCSP0142
C       IDIR = 2   ANGLE = 90 DEGS.                                     MCSP0143
```

```
C        IDIR = 3   ANGLE = 202.5 DEGS.
C        IGUN=1 WILL CALCULATE SHELL AT TARGET RESICUALS
C        TRANSX = TRANSLATION OF ALL X DATA POINTS
C        TRANSY = TRANSLATION OF ALL Y DATA POINTS
C        TRANSZ = TRANSLATION OF ALL Z DATA POINTS
C
      M=N/3
      IF (OPT.EQ.1.AND.N.EQ.9) OPTNO=3
      IF (OPT.EQ.1.AND.N.EQ.6) OPTNO=2
      NUM=N/M
      N2=N/3+1
      N3=N*2/3+1
      N4=N2+1
      N5=N3+1
      DO 10 I=1,9
      START(I)=0.D0
      DO 10 J=1,9
      RADAR(I,J)=0.D0
      STATES(I,J)=0.D0
      PRED(I,J)=0.D0
      QP(I,J)=0.D0
      PHI(I,J)=0.D0
      PKK(I,J)=0.D0
      PKKM1P(I,J)=0.D0
      G(I,J)=0.D0
      H(I,J)=0.D0
      HOLD(I,J) = 0.D0
      GAMMA(I,J) = 0.D0
      GAMMAT(I,J) = 0.D0
   10 RP(I,J)=0.D0
      IF (IGAIN.EQ.0.OR.IGAIN.EQ.1) GO TO 12
      DO 11 I=1,NN
C
C        IF GAINS ARE TO BE READ, THEY ARE DONE SO HERE - THERE WILL BE
C          NN QUANTITY OF (N X NUM) MATRICES
C
      CALL MREAD1 (G,N,NUM)
      DO 11 J=1,N
      DO 11 K=1,NUM
   11 GSTOR(J,K,I)=G(J,K)
      GO TO 13
C
C        INITIAL CONDITIONS ON COVARIANCE OF ESTIMATION ERROR READ HERE
C          THIS WILL BE A (N X N) MATRIX
C
   12 IF(OPT.NE.1) CALL MREAD(PKKM1P,N,N)
C
C        IF THE 'R' MATRIX IS NOT TIME VARYING, READ IT IN HERE
```

```
C         THIS WILL BE A (NUM X NUM) MATRIX                                      MCSP0192
C                                                                                MCSP0193
      IF(IR.NE.0) CALL MREAD(RP,NUM,NUM)                                         MCSP0194
C                                                                                MCSP0195
C      READ INITIAL CONDITION VARIANCES                                          MCSP0196
C         OF RANDOM FORCING MATRIX                                               MCSP0197
C         THIS WILL BE A (NUM X NUM) MATRIX                                      MCSP0198
C                                                                                MCSP0199
      CALL MREAD (QP,NUM,NUM)                                                    MCSP0200
C                                                                                MCSP0201
C      READ GAMMA MATRIX - THIS WILL BE A (N X NUM) MATRIX                       MCSP0202
C                                                                                MCSP0203
      CALL MREAD(GAMMA,N,NUM)                                                    MCSP0204
C                                                                                MCSP0205
C      READ PHI MATRIX HERE - THIS WILL BE A (N X N) MATRIX                      MCSP0206
C                                                                                MCSP0207
   13 CALL MREAD (PHI,N,N)                                                       MCSP0208
C                                                                                MCSP0209
C      READ MEASUREMENT MATRIX HERE - THIS WILL BE A (NUM X N) MATRIX            MCSP0210
C                                                                                MCSP0211
      CALL MREAD (H,NUM,N)                                                       MCSP0212
      IF (IDIR.EQ.0) DEL=0.D0                                                    MCSP0213
      IF (IDIR.EQ.1) DEL=45.D0/DEGRAD                                            MCSP0214
      IF (IDIR.EQ.2) DEL=90.D0/DEGRAD                                            MCSP0215
      IF (IDIR.EQ.3) DEL=202.5D0/DEGRAD                                          MCSP0216
      SINDEL=DSIN(DEL)                                                           MCSP0217
      COSDEL=DCOS(DEL)                                                           MCSP0218
      DO 15 I=1,NN                                                               MCSP0219
C                                                                                MCSP0220
C      THE FOLLOWING SECTION READS THE TRUE (X,Y,Z) DATA AND                     MCSP0221
C         CALCULATES TRUE (R,B,E)                                                MCSP0222
C                                                                                MCSP0223
      READ (5,14) XTRUE(I),YTRUE(I),ZTRUE(I)                                     MCSP0224
   14 FORMAT (8X,3D20.0)                                                         MCSP0225
      XTRUE(I)=XTRUE(I)+TRANSX                                                   MCSP0226
      YTRUE(I)=YTRUE(I)+TRANSY                                                   MCSP0227
      ZTRUE(I)=ZTRUE(I)+TRANSZ                                                   MCSP0228
      XT=XTRUE(I)                                                                MCSP0229
      YT=YTRUE(I)                                                                MCSP0230
      ZT=ZTRUE(I)                                                                MCSP0231
      RT=DSQRT(XT**2+YT**2+ZT**2)                                                MCSP0232
      ET=DATAN2(ZT,DSQRT(XT**2+YT**2))                                           MCSP0233
      BT = 0.D0                                                                  SUPP233A
      IF (YT.GE.0.D0.AND.XT.NE.0.D0) BT=DATAN2(YT,XT)+DEL                        MCSP0234
      IF(YT.GT.0.D0.AND.XT.EQ.0.D0) BT = PI/2.D0 + DEL                           MCSP0235
      IF(YT.EQ.0.D0.AND.XT.LT.0.D0) BT = 180.D0/DEGRAD + DEL                     MCSP0236
      IF (YT.LT.0.D0.AND.XT.LT.0.D0) BT=DATAN2(DABS(YT),DABS(XT))+DEL+18MCSP0237
     10.D0/DEGRAD                                                                MCSP0238
```

```
      IF(YT.LT.0.D0.AND.XT.EQ.0.D0) BT = 3.D0*PI/2.D0 + DEL             MCSP0239
      IF (YT.LT.0.D0.AND.XT.GT.0.D0) BT=-DATAN2(DABS(YT),XT)+DEL+360.D0/MCSP0240
     1DEGRAD                                                            MCSP0241
      RTRUE(I)=RT                                                       MCSP0242
      BTRUE(I)=BT                                                       MCSP0243
      ETRUE(I)=ET                                                       MCSP0244
      XTRUE(I)=RT*DCOS(BT)*DCOS(ET)                                     MCSP0245
      YTRUE(I)=RT*DSIN(BT)*DCOS(ET)                                     MCSP0246
      ZTRUE(I)=RT*DSIN(ET)                                              MCSP0247
C     WRITE(6,4444) I,XTRUE(I),YTRUE(I),ZTRUE(I),RT,BT,ET               MCSP0248
      TVELX(I)=0.D0                                                     MCSP0249
      TVELY(I)=0.D0                                                     MCSP0250
      TVELZ(I)=0.D0                                                     MCSP0251
C                                                                       MCSP0252
C      TRUE VELOCITIES ARE READ IN IF AVAILABLE                         MCSP0253
C                                                                       MCSP0254
      IF (KVEL.EQ.1) READ (5,14) TVELX(I),TVELY(I),TVELZ(I)             MCSP0255
      STOREV = TVELX(I)                                                 SUPP0255
      TVELX(I)= STOREV *COSDEL-TVELY(I)*SINDEL                          MCSP0256
      TVELY(I)=TVELY(I)*COSDEL+ STOREV *SINDEL                          MCSP0257
      AHEAD(I)=0.D0                                                     MCSP0258
   15 CONTINUE                                                          MCSP0259
C                                                                       MCSP0260
C      TIMES FOR SHELL TO REACH TRACK ARE READ IN IF REQUESTED          MCSP0261
C                                                                       MCSP0262
      IF (IGUN.NE.1) GO TO 18                                           MCSP0263
      DO 16 I=1,NN                                                      MCSP0264
   16 READ (5,17) AHEAD(I)                                              MCSP0265
   17 FORMAT (D20.0)                                                    MCSP0266
C                                                                       MCSP0267
C      DETERMINE THE FILTER INITIALIZATION TERMS                        MCSP0268
C                                                                       MCSP0269
   18 IF (BEGIN.NE.1) GO TO 20                                          MCSP0270
      READ (5,14) START(1),START(N2),START(N3),START(2),START(N4),START(MCSP0271
     1N5)                                                               MCSP0272
      DO 19 I=1,9                                                       MCSP0273
   19 PRED(I,1)=START(I)                                                MCSP0274
      GO TO 21                                                          MCSP0275
   20 CALL NOISE (RTRUE(1),BTRUE(1),ETRUE(1),RR,RE,RE,KERNL1,KERNL2,KERNMCSP0276
     1L3,X,Y,Z,INDEX)                                                   MCSP0277
      KERNL1=3790437                                                    MCSP0278
      KERNL2=30117221                                                   MCSP0279
      KERNL3=1993183                                                    MCSP0280
      START(1)=X                                                        MCSP0281
      START(N2)=Y                                                       MCSP0282
      START(N3)=Z                                                       MCSP0283
      PRED(1,1)=X                                                       MCSP0284
      PRED(N2,1)=Y                                                      MCSP0285
```

```
      PRED(N3,1)=Z                                                      MCSP0286
C                                                                       MCSP0287
C      GENERATE NXN IDENTITY MATRIX                                     MCSP0288
C                                                                       MCSP0289
   21 DO 22 I=1,N                                                       MCSP0290
      DO 22 J=1,N                                                       MCSP0291
      II(I,J)=0.D0                                                      MCSP0292
      IF (I.EQ.J) II(I,J)=1.D0                                          MCSP0293
   22 CONTINUE                                                          MCSP0294
C                                                                       MCSP0295
C      THE FOLLOWING SECTION GENERATES HIGHER POWERS OF THE PHI MATRIX  MCSP0296
C        FOR SHELL AT TARGET ACCURACY CALCULATIONS                      MCSP0297
C                                                                       MCSP0298
      IF (IGUN.NE.1) GO TO 26                                           MCSP0299
      DO 23 I=1,N                                                       MCSP0300
      DO 23 J=1,N                                                       MCSP0301
   23 PHIPRM(I,J,1)=PHI(I,J)                                            MCSP0302
      NTIME=39                                                          MCSP0303
      DO 25 I=1,NTIME                                                   MCSP0304
      DO 24 J=1,N                                                       MCSP0305
      DO 24 K=1,N                                                       MCSP0306
   24 A(J,K)=PHIPRM(J,K,I)                                              MCSP0307
      CALL PRCD (A,PHI,N,N,N,B)                                         MCSP0308
      DO 25 J=1,N                                                       MCSP0309
      DO 25 K=1,N                                                       MCSP0310
   25 PHIPRM(J,K,I+1)=B(J,K)                                            MCSP0311
   26 WRITE (6,27) (LABEL(I),I=1,10),N                                  MCSP0312
   27 FORMAT (1H1,10A4,//,10X,'ORDER OF FILTER=',I5,//)                 MCSP0313
      WRITE (6,28) LOOP,IDIR,TRANSX,TRANSY,TRANSZ,INDEX                 MCSP0314
   28 FORMAT (1H ,9X,'NUMBER OF MONTE CARLO RUNS=',I4,//,               MCSP0315
     1  10X,'CODE FOR DIRECTION OF APPROACH=',I4,//,10X,                MCSP0316
     2  'TRANSLATION OF X DATA POINTS=',D20.6,/,25X,                    MCSP0317
     3  'Y DATA POINTS=',D20.6,/,25X,'Z DATA POINTS=',D20.6,//,10X,     MCSP0318
     4  'NOISE CODE=',I3,////)                                          MCSP0319
      WRITE (6,29)                                                      MCSP0320
   29 FORMAT (1H0,'PHI MATRIX',//)                                      MCSP0321
      CALL MWRITE (PHI,N,N)                                             MCSP0322
      WRITE (6,30)                                                      MCSP0323
   30 FORMAT (1H0,//,' H OR MEASUREMENT MATRIX',//)                     MCSP0324
      CALL MWRITE (H,NUM,N)                                             MCSP0325
      IF (IGAIN.EQ.2) GO TO 39                                          MCSP0326
      WRITE (6,121)                                                     MCSP0327
      CALL MWRITE (GAMMA,N,NUM)                                         MCSP0328
      WRITE (6,31)                                                      MCSP0329
   31 FORMAT (1H0,//,' P(K/K-1) OR PREDICTION COVARIANCE MATRIX',//)    MCSP0330
      CALL MWRITE (PKKM1P,N,N)                                          MCSP0331
      IF (ADAPTQ.EQ.1) GO TO 33                                         MCSP0332
      WRITE (6,32)                                                      MCSP0333
```

```
   32 FORMAT (1H0,//,' COVARIANCES OF RANDOM FORCING FOR CALCULATION OF MCSP0334
     1Q MATRIX',//)                                                      MCSP0335
      GO TO 35                                                           MCSP0336
   33 WRITE (6,34) K1,K2                                                 MCSP0337
   34 FORMAT (1H0,//,' INITIAL COVARIANCES OF RANDOM FORCING MATRIX OF 'MCSP0338
     1,/,' STATE EXCITATION FOR ADAPTIVE Q FILTER',//,' WEIGHTING FACTORMCSP0339
     2S FOR Q MATRIX UPDATE ARE - K1=',D20.6,/,45X,'K2=',D20.6,//)       MCSP0340
   35 CALL MWRITE (QP,NUM,NUM)                                           MCSP0341
C                                                                        MCSP0342
C       DEFINE Q MATRIX AS GAMMA X (COVARIANCES OF RANDOM FORCING,       MCSP0343
C         INPUT AS QP) X GAMMA TRANSPOSE                                 MCSP0344
C                                                                        MCSP0345
      CALL TRANS (GAMMA,N,NUM,GAMMAT)                                    MCSP0346
      CALL PROD (GAMMA,QP,N,NUM,NUM,HOLD)                                MCSP0347
      CALL PROD (HOLD,GAMMAT,N,NUM,N,QP)                                 MCSP0348
C                                                                        MCSP0349
C       QP IS NOW THE INITIAL CONDITION COVARIANCE MATRIX                MCSP0350
C         OF STATE EXCITATION                                            MCSP0351
C                                                                        MCSP0352
      IF (IR.EQ.0) GO TO 37                                              MCSP0353
      WRITE (6,36)                                                       MCSP0354
   36 FORMAT (1H0,//,' CONSTANT R OR COVARIANCE OF MEASUREMENT ERROR MATMCSP0355
     1RIX',//)                                                           MCSP0356
      CALL MWRITE (RP,NUM,NUM)                                           MCSP0357
      GO TO 41                                                           MCSP0358
   37 WRITE (6,38)                                                       MCSP0359
   38 FORMAT (1H0,//,' R OR COVARIANCE OF MEAS. ERROR MATRIX GENERATED BMCSP0360
     1Y PROGRAM',//)                                                     MCSP0361
      GO TO 41                                                           MCSP0362
   39 WRITE (6,40)                                                       MCSP0363
   40 FORMAT (1H0,//,' GAINS ARE READ IN',//)                            MCSP0364
   41 WRITE (6,42) RR,RB,RE                                              MCSP0365
   42 FORMAT (1H0,9X,'RANGE STAN DEV,RR=',1PD20.12,/,10X,                MCSP0366
     1 'BEARING STAN DEV,RB=',1PD20.12,/,10X,                            MCSP0367
     2 'ELEVATION STAN DEV,RE=',1PD20.12)                                MCSP0368
      IF (OPT.EQ.1) WRITE (6,43) OPTNO                                   MCSP0369
   43 FORMAT (1H0,//,' THE FIRST ',I2,' ESTIMATES OF POSITION WILL BE THMCSP0370
     1E OBSERVED POSITION',//)                                           MCSP0371
      WRITE (6,44)                                                       MCSP0372
   44 FORMAT (1H0,//,' STATE INITIAL CONDITIONS ARE',//)                 MCSP0373
      CALL MWRITE (PRED,N,1)                                             MCSP0374
      IF (IR.NE.0) GO TO 46                                              MCSP0375
      CALL RNOISE (PRED,RB,RR,RE,N)                                      MCSP0376
      DO 45 I=1,NUM                                                      MCSP0377
      DO 45 J=1,NUM                                                      MCSP0378
   45 RP(I,J)=R(I,J)                                                     MCSP0379
   46 CONTINUE                                                           MCSP0380
C                                                                        MCSP0381
```

```
C         THE FOLLOWING LOOP IS THE MONTE CARLO SIMULATION                  MCSP0382
C                                                                           MCSP0383
        DO 58 ITER=1,LOOP                                                   MCSP0384
        DO 47 I=1,N                                                         MCSP0385
        DO 47 J=1,N                                                         MCSP0386
        Q(I,J)=QP(I,J)                                                      MCSP0387
        PKKM1(I,J)=PKKM1P(I,J)                                              MCSP0388
   47   R(I,J)=RP(I,J)                                                      MCSP0389
        DO 48 I=1,N                                                         MCSP0390
   48   PRED(I,1)=START(I)                                                  MCSP0391
C                                                                           MCSP0392
C         THE FOLLOWING LOOP STEPS THROUGH THE TRACK POINTS                 MCSP0393
C                                                                           MCSP0394
        DO 57 L=1,NN                                                        MCSP0395
C                                                                           MCSP0396
C         IF DESIRED, NOISE IS ADDED TO THE TRACK POINTS                    MCSP0397
C                                                                           MCSP0398
        CALL NOISE (RTRUE(L),BTRUE(L),ETRUE(L),RR,RB,RE,KERNL1,             MCSP0399
      1 KERNL2,KERNL3,X,Y,Z,INDEX)                                          MCSP0400
        IFLAG = 0                                                           MCSP0401
C                                                                           MCSP0402
C         THE FOLLOWING CARD SETS THE OPTION FOR USING INITIAL OBSERVATIONSMCSP0403
C           TO GENERATE PREDICTIONS INSTEAD OF FILTER ESTIMATES             MCSP0404
C                                                                           MCSP0405
        IF (OPT.EQ.1.AND.L.LE.OPTNO) IFLAG=1                                MCSP0406
C                                                                           MCSP0407
C         THE FOLLOWING SECTION HANDLES THE VARIOUS GAIN OPTIONS            MCSP0408
C                                                                           MCSP0409
        IF (IGAIN.EQ.0) GO TO 52                                            MCSP0410
        IF (IGAIN.EQ.1.AND.ITER.NE.1) GO TO 50                              MCSP0411
        IF (IGAIN.EQ.2) GO TO 50                                            MCSP0412
        CALL GAIN (N,NUM,ITER,L,IFLAG,KVEL)                                 MCSP0413
        DO 49 IZ=1,N                                                        MCSP0414
        DO 49 IX=1,NUM                                                      MCSP0415
   49   GSTOR(IZ,IX,L)=G(IZ,IX)                                             MCSP0416
        GO TO 53                                                            MCSP0417
   50   DO 51 IZ=1,N                                                        MCSP0418
        DO 51 IX=1,NUM                                                      MCSP0419
   51   G(IZ,IX)=GSTOR(IZ,IX,L)                                             MCSP0420
        GO TO 53                                                            MCSP0421
   52   CALL GAIN (N,NUM,ITER,L,IFLAG,KVEL)                                 MCSP0422
C                                                                           MCSP0423
C         FOR MOST SITUATIONS THE RADAR MEASUREMENTS ARE MADE NOISY         MCSP0424
C                                                                           MCSP0425
   53   RADAR(1,1)=X                                                        MCSP0426
        RADAR(2,1)=Y                                                        MCSP0427
        RADAR(3,1)=Z                                                        MCSP0428
C                                                                           MCSP0429
```

```
C
C     HOWEVER, FOR 1 GAIN MONTE CARLO SIMULATION TRUE TRACK POINTS ARE
C       USED - THIS IS ONLY DONE ON THE FIRST MONTE CARLO RUN
C
      IF (IGAIN.NE.1.OR.ITER.NE.1) GO TO 54
      RADAR(1,1)=XTRUE(L)
      RADAR(2,1)=YTRUE(L)
      RADAR(3,1)=ZTRUE(L)
   54 CONTINUE
C
C     'UPDATE' WILL GENERATE ESTIMATED AND PREDICTED FILTER STATES
C
      CALL UPDATE (N,NUM,RADAR,STATES,PRED,IFLAG,L,TIME,IGAIN,ITER,K1,K2
     1,ADAPTQ)
      DO 55 I=1,9
C
C     THE FOLLOWING MATRIX SAVES THE ESTIMATED STATES FOR USE IN
C       CALCULATING MISS DISTANCES WHEN THE SHELL REACHES THE TARGET
C
   55 EST(I,L)=STATES(I,1)
C
C     THE FOLLOWING 'IF' CALLS FOR COVARIANCE OF MEASUREMENT NOISE ONLY
C       WHEN NECESSARY
C
      IF((IGAIN.EQ.0.OR.(IGAIN.EQ.1.AND.ITER.EQ.1)).AND.IR.EQ.0)
     1 CALL RNOISE(PRED,RB,RR,RE,N)
C
C     'ACCUR' IS CALLED TO CALCULATE SHELL AT TARGET MISS DISTANCE
C       WHEN SUFFICIENT TIME HAS PASSED FOR SHELL TO GET TO TARGET
C
      IF (IGUN.NE.1) GO TO 56
      TSPACE=1.D0/TIME
      IF (FLOAT(L).GT.(TSPACE*AHEAD(L)+1.D0)) CALL ACCUR (XTRUE(L),YTRUE
     1(L),ZTRUE(L),L,N,ITER)
C
C     'SCENAR' IS CALLED TO PROVIDE MONTE CARLO STATISTICS
C
   56 CALL SCENAR (XTRUE(L),YTRUE(L),ZTRUE(L),STATES,N,L,ITER,IGAIN,TVEL
     1X(L),TVELY(L),TVELZ(L),KVEL)
   57 CONTINUE
   58 CONTINUE
C
C     OUTPUT PERFORMANCE TABLE IF SHELL AT TARGET RESIDUALS CALCULATED
C
      IF (IGUN.NE.1) GO TO 60
      WRITE (6,59)
   59 FORMAT (1H1,'POINT',7X,'TRUE',10X,'FILTER',8X,'RESIDUALS',8X,
     1 'TIME',10X,'MEAN',/,' NUMBER',6X,'TRACK',10X,'PRED',25X,
     2 'OF FLT',9X,'MISS',/,13X,'POINTS', 8X,'POINTS',24X,'------',
```

```
     3  7X,'DISTANCE',/,55X,'TRACK TIME',/,55X,'WHEN FIRED',/,1X,          MCSP04
     4  88('-'),//)                                                      MCSP0479
C                                                                        MCSP0480
C       THE FOLLOWING SECTION DOES FINAL COMPUTATIONS ON STATISTICAL     MCSP0481
C         QUANTITIES AND OUTPUTS THEM                                    MCSP0482
C                                                                        MCSP0483
   60 DO 62 I=1,NN                                                       MCSP0484
C                                                                        MCSP0485
C       RMEAN'S ARE CHANGED TO MEAN OF ESTIMATE AT EACH POINT FROM SUM   MCSP0486
C         OF ESTIMATES FOR ALL MONTE CARLO RUNS                          MCSP0487
C                                                                        MCSP0488
      DO 61 J=1,6                                                        MCSP0489
   61 RMEAN(I,J)=RMEAN(I,J)/FLOAT(LOOP)                                  MCSP0490
C                                                                        MCSP0491
C       THE FOLLOWING 6 TERMS ARE USED TO CALCULATE FILTER PERFORMANCE   MCSP0492
C         FACTORS, AND REPRESENT SUMS OF MEAN ERRORS                     MCSP0493
C                                                                        MCSP0494
      RSD(1)=RSD(1)+DABS(RMEAN(I,1)-XTRUE(I))                            MCSP0495
      RSD(2)=RSD(2)+DABS(RMEAN(I,2)-YTRUE(I))                            MCSP0496
      RSD(3)=RSD(3)+DABS(RMEAN(I,3)-ZTRUE(I))                            MCSP0497
      RSD(4) = RSD(4) + DABS(RMEAN(I,4)-TVELX(I))                        MCSP0498
      RSD(5) = RSD(5) + DABS(RMEAN(I,5)-TVELY(I))                        MCSP0499
      RSD(6) = RSD(6) + DABS(RMEAN(I,6)-TVELZ(I))                        MCSP0500
   62 CONTINUE                                                           MCSP0501
      IF (IGUN.NE.1) GO TO 66                                            MCSP0502
      IFIRE = 0                                                          MCSP0503
      DO 63 I=1,NN                                                       MCSP0504
      RESID(I,1)=RESID(I,1)/FLOAT(LOOP)                                  MCSP0505
      RESID(I,2)=RESID(I,2)/FLOAT(LOOP)                                  MCSP0506
      RESID(I,3)=RESID(I,3)/FLOAT(LOOP)                                  MCSP0507
      RESID(I,4)=DSQRT(RESID(I,1)**2+RESID(I,2)**2+RESID(I,3)**2)        MCSP0508
C                                                                        MCSP0509
C       IF INSUFFICIENT TIME HAD PASSED TO CALCULATE MISS DISTANCE       MCSP0510
C         THEN DO NOT OUTPUT STATISTICS                                  MCSP0511
C                                                                        MCSP0512
      IF(RESID(I,4).EQ.0.D0) GO TO 63                                    MCSP0513
      IFIRE = IFIRE + 1                                                  MCSP0514
      CALL STUDY (I,XTRUE(I),YTRUE(I),ZTRUE(I),RESID(I,1),RESID(I,2),RESMCSP0515
     1ID(I,3),AHEAD(I),RESID(I,4))                                       MCSP0516
   63 CONTINUE                                                           MCSP0517
      GUNSUM = 0.D0                                                      MCSP0518
      DO 64 I=1,NN                                                       MCSP0519
   64 GUNSUM = GUNSUM + RESID(I,4)                                       MCSP0520
      GMISS = GUNSUM/FLOAT(IFIRE)                                        MCSP0521
      WRITE (6,65) GMISS                                                 MCSP0522
   65 FORMAT(1H0,////,10X,'AVERAGE SHELL AT TARGET MISS DISTANCE =',     MCSP0523
     1 D20.8,' YARDS')                                                   MCSP0524
   66 PERFX=RSD(1)/FLOAT(NN)                                             MCSP0525
```

```
      PERFY=RSD(2)/FLOAT(NN)
      PERFZ=RSD(3)/FLOAT(NN)
      PERFXV = RSD(4)/FLOAT(NN)
      PERFYV = RSD(5)/FLOAT(NN)
      PERFZV = RSD(6)/FLOAT(NN)
C
C      CALCULATE FILTER PERFORMANCE FACTOR
C
      PERF=DSQRT(PERFX**2+PERFY**2+PERFZ**2)
      WRITE (6,67) PERF
   67 FORMAT (1H0,////,10X,'PERFORMANCE FACTOR CF FILTER, AVERAGE FILTER
     1 POSITION ESTIMATE ERROR = ',D20.8,' YARDS')
      PERFV = DSQRT(PERFXV**2+PERFYV**2+PERFZV**2)
      IF(KVEL.EQ.1) WRITE(6,120) PERFV
      WRITE (6,68)
   68 FORMAT (1H1,'POINT NO',6X,'POSITICN',12X,'PCSITION',12X,'VELOCITY'
     1, 11X,'VELOCITY VARIANCES',
     2  /,15X,'ESTIMATE',12X,  'VARIANCES',12X,'/ERRORS',11X,
     3    'ARCUND ESTIMATE/',/,16X,'ERRORS',52X,'AROUND TRUTH',/,
     4    88('_'),//)
      DO 73 I=1,NN
      DO 69 J=1,9
   69 PKSTOR(J,I)=PKSTOR(J,I)/FLOAT(LCOP)
C
C      RMEAN'S NOW BECOME THE EXPECTEC VALUE OF ESTIMATICN ERROR
C
      RMEAN(I,1)=RMEAN(I,1)-XTRUE(I)
      RMEAN(I,2)=RMEAN(I,2)-YTRUE(I)
      RMEAN(I,3)=RMEAN(I,3)-ZTRUE(I)
C
C      PKSTOR'S BECOME VARIANCES DEFINED AS EXPECTED VALUE OF SQUARE
C        OF ESTIMATICN ERROR MINUS EXPECTED VALUE OF ESTIMATICN ERROR**2
C        E(X**2) - E(X)**2
C
      PKSTOR(1,I)=PKSTOR(1,I)-RMEAN(I,1)**2
      PKSTOR(N2,I)=PKSTOR(N2,I)-RMEAN(I,2)**2
      PKSTOR(N3,I)=PKSTOR(N3,I)-RMEAN(I,3)**2
      IF (KVEL.NE.1) GO TO 71
      RMEAN(I,4)=RMEAN(I,4)-TVELX(I)
      RMEAN(I,5)=RMEAN(I,5)-TVELY(I)
      RMEAN(I,6)=RMEAN(I,6)-TVELZ(I)
      PKSTOR(2,I)=PKSTOR(2,I)-RMEAN(I,4)**2
      PKSTOR(N4,I)=PKSTOR(N4,I)-RMEAN(I,5)**2
      PKSTOR(N5,I)=PKSTOR(N5,I)-RMEAN(I,6)**2
      WRITE (6,70) I,RMEAN(I,1),PKSTOR(1,I),RMEAN(I,4),PKSTOR(2,I),RMEAN
     1(I,2),PKSTOR(N2,I),RMEAN(I,5),PKSTCR(N4,I),RMEAN(I,3),PKSTOR(N3,I)
     2,RMEAN(I,6),PKSTOR(N5,I)
   70 FORMAT (1H0,I5,4X,'X:',D15.6,4X,'X:',D15.6,C15.6,9X,'X:',D15.6,/,
```

```
   1   10X,'Y:',D15.6,4X,'Y:',D15.6,D15.6,9X,'Y:',D15.6,/,            MCSP0574
   2   10X,'Z:',D15.6,4X,'Z:',D15.6,D15.6,9X,'Z:',D15.6,/)            MCSP0575
      GC TO 73                                                        MCSP0576
C                                                                     MCSP0577
C       IF TRUE VELOCITIES ARE NOT AVAILABLE, THEN CALCULATE MEAN AND MCSP0578
C         VARIANCE OF ESTIMATED VELOCITY, NOT ERROR OF ESTIMATED      MCSP0579
C         VELOCITY                                                    MCSP0580
C                                                                     MCSP0581
   71 AVEL=DSQRT(RMEAN(I,4)**2+RMEAN(I,5)**2+RMEAN(I,6)**2)           MCSP0582
      PKSTOR(2,I)=PKSTOR(2,I)-RMEAN(I,4)**2                           MCSP0583
      PKSTOR(N4,I)=PKSTOR(N4,I)-RMEAN(I,5)**2                         MCSP0584
      PKSTOR(N5,I)=PKSTOR(N5,I)-RMEAN(I,6)**2                         MCSP0585
      WRITE (6,72) I,RMEAN(I,1),PKSTOR(1,I),PKSTOR(2,I),RMEAN(I,2),PKSTOMCSP0586
     1R(N2,I),AVEL,PKSTOR(N4,I),RMEAN(I,3),PKSTOR(N3,I),PKSTOR(N5,I)  MCSP0587
   72 FORMAT (1H0,I5,4X,'X:',D15.6,4X,'X:',D15.6,24X,'X:',D15.6,/,    MCSP0588
     1   10X,'Y:',D15.6,4X,'Y:',D15.6,D15.6,9X,'Y:',D15.6,/,          MCSP0589
     2   10X,'Z:',D15.6,4X,'Z:',D15.6,24X,'Z:',D15.6,/)               MCSP0590
   73 CONTINUE                                                        MCSP0591
C                                                                     MCSP0592
C                                                                     MCSP0593
C       FOLLOWING IS THE SEGMENT FOR OUTPUTING GRAPHS                 MCSP0594
C                                                                     MCSP0595
C                                                                     MCSP0596
      DO 74 I=1,NN                                                    MCSP0597
   74 SEEABS(I)=I                                                     MCSP0598
      IF (LOOK(1).EQ.0.OR.IGUN.NE.1) GO TO 77                         MCSP0599
      WRITE (6,75)                                                    MCSP0600
   75 FORMAT (1H1,10X,'KETRON MK-86 THESIS',/,11X,                    MCSP0601
     1   'MEAN MISS DISTANCE WHEN SHELL ARRIVES IN YDS.',//)          MCSP0602
      DO 76 I=1,NN                                                    MCSP0603
   76 SEEORD(I)=RESID(I,4)                                            MCSP0604
      CALL PLOTP (SEEABS,SEEORD,NN,0)                                 MCSP0605
   77 IF (LOOK(2).EQ.0) GO TO 80                                      MCSP0606
      WRITE (6,78)                                                    MCSP0607
   78 FORMAT (1H1,10X,'KETRON MK-86 THESIS',/,11X,                    MCSP0608
     1   'MEAN ERROR IN X POSITION',//)                               MCSP0609
      DO 79 I=1,NN                                                    MCSP0610
   79 SEEORD(I)=RMEAN(I,1)                                            MCSP0611
      CALL PLOTP (SEEABS,SEEORD,NN,0)                                 MCSP0612
   80 IF (LOOK(3).EQ.0) GO TO 83                                      MCSP0613
      WRITE (6,81)                                                    MCSP0614
   81 FORMAT (1H1,10X,'KETRON MK-86 THESIS',/,11X,                    MCSP0615
     1   'MEAN ERROR IN Y POSITION',//)                               MCSP0616
      DO 82 I=1,NN                                                    MCSP0617
   82 SEEORD(I)=RMEAN(I,2)                                            MCSP0618
      CALL PLOTP (SEEABS,SEEORD,NN,0)                                 MCSP0619
   83 IF (LOOK(4).EQ.0) GO TO 86                                      MCSP0620
      WRITE (6,84)                                                    MCSP0621
```

```
 84 FORMAT (1H1,10X,'KETRON MK-86 THESIS',/,11X,                    MCSP0622
   1   'MEAN ERROR IN Z POSITION',//)                              MCSP0623
    DO 85 I=1,NN                                                   MCSP0624
 85 SEEORD(I)=RMEAN(I,3)                                           MCSP0625
    CALL PLOTP (SEEABS,SEEORD,NN,0)                                MCSP0626
 86 IF (LOOK(5).EQ.0) GO TO 89                                     MCSP0627
    WRITE (6,87)                                                   MCSP0628
 87 FORMAT (1H1,10X,'KETRON MK-86 THESIS',/,11X,                    MCSP0629
   1   'X POSITION VARIANCE',//)                                   MCSP0630
    DO 88 I=1,NN                                                   MCSP0631
 88 SEEORD(I)=PKSTOR(1,I)                                          MCSP0632
    CALL PLOTP (SEEABS,SEEORD,NN,0)                                MCSP0633
 89 IF (LOOK(6).EQ.0) GO TO 92                                     MCSP0634
    WRITE (6,90)                                                   MCSP0635
 90 FORMAT (1H1,10X,'KETRON MK-86 THESIS',/,11X,                    MCSP0636
   1   'Y POSITION VARIANCE',//)                                   MCSP0637
    DO 91 I=1,NN                                                   MCSP0638
 91 SEEORD(I)=PKSTOR(N2,I)                                         MCSP0639
    CALL PLOTP (SEEABS,SEEORD,NN,0)                                MCSP0640
 92 IF (LOOK(7).EQ.0) GO TO 95                                     MCSP0641
    WRITE (6,93)                                                   MCSP0642
 93 FORMAT (1H1,10X,'KETRON MK-86 THESIS',/,11X,                    MCSP0643
   1   'Z POSITION VARIANCE',//)                                   MCSP0644
    DO 94 I=1,NN                                                   MCSP0645
 94 SEEORD(I)=PKSTOR(N3,I)                                         MCSP0646
    CALL PLOTP (SEEABS,SEEORD,NN,0)                                MCSP0647
 95 IF (LOOK(8).EQ.0) GO TO 98                                     MCSP0648
    WRITE (6,96)                                                   MCSP0649
 96 FORMAT (1H1,10X,'KETRON MK-86 THESIS',/,11X,                    MCSP0650
   1   'MEAN VELOCITY ERROR IN X DIRECTION',//)                    MCSP0651
    DO 97 I=1,NN                                                   MCSP0652
 97 SEEORD(I)=RMEAN(I,4)                                           MCSP0653
    CALL PLOTP (SEEABS,SEEORD,NN,0)                                MCSP0654
 98 IF (LOOK(9).EQ.0) GO TO 101                                    MCSP0655
    WRITE (6,99)                                                   MCSP0656
 99 FORMAT (1H1,10X,'KETRON MK-86 THESIS',/,11X,                    MCSP0657
   1   'MEAN VELOCITY ERROR IN Y DIRECTION',//)                    MCSP0658
    DO 100 I=1,NN                                                  MCSP0659
100 SEEORD(I)=RMEAN(I,5)                                           MCSP0660
    CALL PLOTP (SEEABS,SEEORD,NN,0)                                MCSP0661
101 IF (LOOK(10).EQ.0) GO TO 104                                   MCSP0662
    WRITE (6,102)                                                  MCSP0663
102 FORMAT (1H1,10X,'KETRON MK-86 THESIS',/,11X,                    MCSP0664
   1   'MEAN VELOCITY ERROR IN Z DIRECTION',//)                    MCSP0665
    DO 103 I=1,NN                                                  MCSP0666
103 SEEORD(I)=RMEAN(I,6)                                           MCSP0667
    CALL PLOTP (SEEABS,SEEORD,NN,0)                                MCSP0668
104 IF (LOOK(11).EQ.0) GO TO 107                                   MCSP0669
```

```
      WRITE (6,105)                                                     MCSP0670
  105 FORMAT (1H1,10X,'KETRON MK-86 THESIS',/,11X,                      MCSP0671
     1  'X VELOCITY VARIANCE',//)                                       MCSP0672
      DO 106 I=1,NN                                                     MCSP0673
  106 SEEORD(I)=PKSTOR(2,I)                                             MCSP0674
      CALL PLOTP (SEEABS,SEEORD,NN,0)                                   MCSP0675
  107 IF (LOOK(12).EQ.0) GO TO 110                                      MCSP0676
      WRITE (6,108)                                                     MCSP0677
  108 FORMAT (1H1,10X,'KETRON MK-86 THESIS',/,11X,                      MCSP0678
     1  'Y VELOCITY VARIANCE',//)                                       MCSP0679
      DO 109 I=1,NN                                                     MCSP0680
  109 SEEORD(I)=PKSTOR(N4,I)                                            MCSP0681
      CALL PLOTP (SEEABS,SEEORD,NN,0)                                   MCSP0682
  110 IF (LOOK(13).EQ.0) GO TO 113                                      MCSP0683
      WRITE (6,111)                                                     MCSP0684
  111 FORMAT (1H1,10X,'KETRON MK-86 THESIS',/,11X,                      MCSP0685
     1  'Z VELOCITY VARIANCE',//)                                       MCSP0686
      DO 112 I=1,NN                                                     MCSP0687
  112 SEEORD(I)=PKSTOR(N5,I)                                            MCSP0688
      CALL PLOTP (SEEABS,SEEORD,NN,0)                                   MCSP0689
  113 IF (LOOK(14).EQ.0) GO TO 116                                      MCSP0690
      WRITE (6,114)                                                     MCSP0691
  114 FORMAT (1H1,10X,'KETRON MK-86 THESIS',/,11X,                      MCSP0692
     1  'TARGET SPEED IN FT/SEC',//)                                    MCSP0693
      DO 115 I=1,NN                                                     MCSP0694
  115 SEEORD(I)=DSQRT((RMEAN(I,4)+TVELX(I))**2+(RMEAN(I,5)+TVELY(I))**2+MCSP0695
     1(RMEAN(I,6)+TVELZ(I))**2)*3.D0                                    MCSP0696
      CALL PLOTP (SEEABS,SEEORD,NN,0)                                   MCSP0697
  116 CONTINUE                                                          MCSP0698
C                                                                       MCSP0699
C       THE FOLLOWING SEGMENT PUNCHES GAIN MATRICES IF REQUESTED        MCSP0700
C                                                                       MCSP0701
      IF (IG.NE.1) GO TO 119                                            MCSP0702
      DO 117 I=1,NN                                                     MCSP0703
      DO 117 J=1,N                                                      MCSP0704
  117 WRITE (7,118) I,J,(GSTOR(J,JJ,I),JJ=1,3)                          MCSP0705
  118 FORMAT (3X,2I4,3D20.8)                                            MCSP0706
  119 STOP                                                              MCSP0707
  120 FORMAT(1H0,//,40X,'AVERAGE FILTER VELOCITY ESTIMATE ERROR = ',    MCSP0708
     1 D20.8,' YARDS/SEC')                                              MCSP0709
  121 FORMAT(1H0,//,' GAMMA MATRIX',//)                                 MCSP0710
      END                                                               MCSP0711


      SUBROUTINE NOISE (RT,BT,ET,R1,RB,RE,KERNL1,KERNL2,KERNL3,X,Y,Z,   MCSP0712
```

```
     1 INDEX)                                                          MCSP0713
C      THIS SUBROUTINE TAKES THE INPUT R,B,E DATA, ADDS NOISE, AND THEN MCSP0714
C        CONVERTS INTO NOISY X,Y,Z.                                     MCSP0715
C                                                                       MCSP0716
      IMPLICIT REAL*8(A-H,L-Z)                                          MCSP0717
      REAL*4 RERR,BERR,EERR                                             MCSP0718
      RERROR = 0                                                        MCSP0719
      BERROR = 0                                                        MCSP0720
      EERROR = 0                                                        MCSP0721
      IF(INDEX.EQ.0) GO TO 1                                            MCSP0722
C                                                                       MCSP0723
C      SET RANGE STANDARD DEVIATION AT INPUT VALUE + .1 PER CENT RANGE  MCSP0724
C                                                                       MCSP0725
      RR=R1+.001D0*RT                                                   MCSP0726
      CALL NORMAL (KERNL1,RERR,1)                                       MCSP0727
      RERROR=RR*DBLE(RERR)                                              MCSP0728
      CALL NORMAL (KERNL2,BERR,1)                                       MCSP0729
      BERROR=RB*DBLE(BERR)                                              MCSP0730
      CALL NORMAL (KERNL3,EERR,1)                                       MCSP0731
      EERROR=RE*DBLE(EERR)                                              MCSP0732
    1 R=RT+RERROR                                                       MCSP0733
      B=BT+BERROR                                                       MCSP0734
      E=ET+EERROR                                                       MCSP0735
C                                                                       MCSP0736
C      CONVERT INTO NOISY X,Y,Z DATA                                    MCSP0737
C                                                                       MCSP0738
      X=R*DCOS(B)*DCOS(E)                                               MCSP0739
      Y=R*DSIN(B)*DCOS(E)                                               MCSP0740
      Z=R*DSIN(E)                                                       MCSP0741
      RETURN                                                            MCSP0742
      END                                                               MCSP0743


      SUBROUTINE UPDATE (N,NUM,RADAR,STATES,PRED,IFLAG,L,TIME,IGAIN,ITERMCSP0744
     1,K1,K2,ADAPTQ)                                                    MCSP0745
C                                                                       MCSP0746
C      THIS ROUTINE COMPUTES THE FILTER ESTIMATE X(K/K)                 MCSP0747
C        AND FILTER PREDICTION X(K+1/K)                                 MCSP0748
C        IT ALSO GENERATES THE SYNTHETIC ESTIMATES IF REQUESTED, AND    MCSP0749
C        THE ADAPTIVE Q MATRIX IS UPDATED IF DESIRED                    MCSP0750
C                                                                       MCSP0751
      IMPLICIT REAL*8(P,Q,H,G,R,T,S)                                    MCSP0752
      REAL*8 TEMP1,TEMP2,TEMP3,II,AHEAD,EST,K1,K2                       MCSP0753
      INTEGER*4 ADAPTQ                                                  MCSP0754
      DIMENSION RADAR(9,9), STATES(9,9), PRED(9,9), TEMP1(9,9), TEMP2(9,MCSP0755
```

```
     19), TEMP3(9,9), QOLD(9,9), QRESID(9,9), SPDIFF(9,9), SPTR(9,9)
      COMMON II(9,9),Q(9,9),H(9,9),R(9,9),G(9,9),PHI(9,9),PKK(9,9),PKKM1
     1(9,9),RMEAN(233,6),RSD(6),RESID(233,4),AHEAD(233),EST(9,233),PHIPR
     2M(9,9,40),PKSTOR(9,233)
      DO 1 I=1,9
      DO 1 J=1,9
      TEMP1(I,J)=0.D0
      TEMP2(I,J)=0.D0
    1 TEMP3(I,J)=0.D0
      IF(IFLAG.EQ.1) GO TO 2
C
C      X(K/K) = X(K/K-1) + G(K)(Z(K)-HX(K/K-1))
C
      CALL PROD (H,PRED,NUM,N,1,TEMP1)
      CALL SUB (RADAR,TEMP1,NUM,1,TEMP3)
      CALL PROD (G,TEMP3,N,NUM,1,TEMP2)
      CALL ADD (PRED,TEMP2,N,1,STATES)
C
C      IFLAG = 1 SIGNIFIES THAT THE ESTIMATED POSITIONS ARE TO BE THE
C        MEASURED POSITIONS - THIS OPTION IS ONLY AVAILABLE FOR
C        THE CASES WHERE GAINS ARE CALCULATED BY THE PROGRAM
C
    2 IF(IFLAG.NE.1.OR.IGAIN.EQ.2) GO TO 5
      CALL TRANS (PHI,N,N,TEMP2)
C
C      THE FOLLOWING SEGMENT GENERATES SYNTHETIC FILTER ESTIMATES AND
C        COVARIANCES OF ESTIMATION ERROR IF REQUESTED BY 'OPT' = 1
C
      N2=N/3+1
      N3=N*2/3+1
      N4=N2+1
      N5=N3+1
      STATES(1,1)=RADAR(1,1)
      STATES(N2,1)=RADAR(2,1)
      STATES(N3,1)=RADAR(3,1)
      IF (L.EQ.1) GO TO 5
      STATES(2,1)=(RADAR(1,1)-EST(1,L-1))/TIME
      STATES(N4,1)=(RADAR(2,1)-EST(N2,L-1))/TIME
      STATES(N5,1)=(RADAR(3,1)-EST(N3,L-1))/TIME
      PKK(1,1)=1.D0
      PKK(N2,N2)=1.D0
      PKK(N3,N3)=1.D0
      PKK(1,2)=1.D0/TIME
      PKK(N2,N4)=PKK(1,2)
      PKK(N3,N5)=PKK(1,2)
      PKK(2,1)=1.D0/TIME
      PKK(N4,N2)=PKK(2,1)
      PKK(N5,N3)=PKK(2,1)
```

```
      PKK(2,2)=2.DO/TIME**2                                             MCSP0804
      PKK(N4,N4)=PKK(2,2)                                               MCSP0805
      PKK(N5,N5)=PKK(2,2)                                               MCSP0806
C                                                                       MCSP0807
C      IF FILTER IS 2ND ORDER OR 3 SAMPLES HAVE NOT BEEN TAKEN TO       MCSP08C8
C         CALCULATE FINAL 3RD ORDER ESTIMATES AND COVARIANCE TERMS,     MCSP0809
C         SKIP THE FOLLOWING SECTION                                    MCSP0810
C                                                                       MCSP0811
      IF (N.EQ.6) GO TO 3                                               MCSP0812
      IF (N.EQ.9.AND.L.LT.3) GO TO 5                                    MCSP0813
      N6=N4+1                                                           MCSP0814
      N7=N5+1                                                           MCSP0815
      STATES(3,1)=(RADAR(1,1)-2.DO*EST(1,L-1)+EST(1,L-2))/TIME**2       MCSP0816
      STATES(N6,1)=(RADAR(2,1)-2.DO*EST(N2,L-1)+EST(N2,L-2))/TIME**2    MCSP0817
      STATES(N7,1)=(RADAR(3,1)-2.DO*EST(N3,L-1)+EST(N3,L-2))/TIME**2    MCSP0818
      PKK(1,3)=1.DO/TIME**2                                             MCSP0819
      PKK(N2,N6)=PKK(1,3)                                               MCSP0820
      PKK(N3,N7)=PKK(1,3)                                               MCSP0821
      PKK(2,3)=3.DO/TIME**3                                             MCSP0822
      PKK(N4,N6)=PKK(2,3)                                               MCSP0823
      PKK(N5,N7)=PKK(2,3)                                               MCSP0824
      PKK(3,1)=1.DO/TIME**2                                             MCSP0825
      PKK(N6,N2)=PKK(3,1)                                               MCSP0826
      PKK(N7,N3)=PKK(3,1)                                               MCSP0827
      PKK(3,2)=3.DO/TIME**3                                             MCSP0828
      PKK(N6,N4)=PKK(3,2)                                               MCSP0829
      PKK(N7,N5)=PKK(3,2)                                               MCSP0830
      PKK(3,3)=6.DO/TIME**4                                             MCSP0831
      PKK(N6,N6)=PKK(3,3)                                               MCSP0832
      PKK(N7,N7)=PKK(3,3)                                               MCSP0833
    3 DC 4 I=1,N                                                        MCSP0834
      DC 4 J=1,N                                                        MCSP0835
    4 PKKM1(I,J)=0.DO                                                   MCSP0836
      CALL PROD (PHI,PKK,N,N,N,TEMP1)                                   MCSP0837
      CALL PROD (TEMP1,TEMP2,N,N,N,TEMP3)                               MCSP0838
      CALL ADD(TEMP3,Q,N,N,PKKM1)                                       MCSP0839
    5 CONTINUE                                                          MCSP0840
C                                                                       MCSP0841
C      THE FOLLOWING SEGMENT IS USED TO UPDATE THE ADAPTIVE Q MATRIX    MCSP0842
C                                                                       MCSP0843
      IF (ADAPTQ.NE.1.OR.IGAIN.EQ.2.OR.(IGAIN.EQ.1.AND.ITER.NE.1)) GO TOMCSP0844
     1 8                                                                MCSP0845
      DC 6 I=1,N                                                        MCSP0846
      DO 6 J=1,N                                                        MCSP0847
    6 SPDIFF(I,J)=STATES(I,J)-PRED(I,J)                                 MCSP0848
      CALL TRANS (SPDIFF,N,N,SPTR)                                      MCSP0849
      CALL PROD (SPDIFF,SPTR,N,N,N,QRESID)                              MCSP0850
      DC 7 I=1,N                                                        MCSP0851
```

```
      DO 7 J=1,N                                                        MCSP0852
      QOLD(I,J)=Q(I,J)                                                  MCSP0853
      Q(I,J) = K1*QOLD(I,J) + K2*QRESID(I,J)                            MCSP0854
    7 CONTINUE                                                          MCSP0855
C                                                                       MCSP0856
C       X(K+1/K) = PHI * X(K/K)                                         MCSP0857
C                                                                       MCSP0858
    8 CALL PROD (PHI,STATES,N,N,1,PRED)                                 MCSP0859
      RETURN                                                            MCSP0860
      END                                                               MCSP0861



      SUBROUTINE SCENAR (X,Y,Z,STATES,N,L,ITER,IGAIN,TVELX,TVELY,TVELZ,KMCSP0862
     1VEL)                                                              MCSP0863
C                                                                       MCSP0864
C       THIS SUBROUTINE COMPUTES THE MEANS OF THE FILTER ESTIMATES      MCSP0865
C         FOR 'ITER' SAMPLES FOR USE IN CALCULATING THE FILTER          MCSP0866
C         PERFORMANCE FACTOR AND ALSO THE VARIANCES WHEN GAINS ARE READ MCSP0867
C                                                                       MCSP0868
      IMPLICIT REAL*8(P,Q,H,G,R,T,S)                                    MCSP0869
      REAL*8 X,Y,Z,XCAL,YCAL,ZCAL,TERM1,TERM2,II,AHEAD,EST              MCSP0870
      DIMENSION STATES(9,9)                                             MCSP0871
      COMMON II(9,9),Q(9,9),H(9,9),R(9,9),G(9,9),PHI(9,9),PKK(9,9),PKKM1MCSP0872
     1(9,9),RMEAN(233,6),RSD(6),RESID(233,4),AHEAD(233),EST(9,233),PHIPRMCSP0873
     2M(9,9,40),PKSTOR(9,233)                                           MCSP0874
      N2=N/3+1                                                          MCSP0875
      N3=N*2/3+1                                                        MCSP0876
      N4=N2+1                                                           MCSP0877
      N5=N3+1                                                           MCSP0878
      XCAL=STATES(1,1)                                                  MCSP0879
      YCAL=STATES(N2,1)                                                 MCSP0880
      ZCAL=STATES(N3,1)                                                 MCSP0881
C                                                                       MCSP0882
C       RMEAN'S ARE SUM OF FILTER ESTIMATES IN THIS ROUTINE - AFTER FULL MCSP0883
C         MONTE CARLO RUN HAS BEEN CONCLUDED THE MEANS ARE CALCULATED BY MCSP0884
C         DIVIDING THESE SUMS BY THE NUMBER OF MONTE CARLO RUNS IN MAIN MCSP0885
C                                                                       MCSP0886
      RMEAN(L,1)=RMEAN(L,1)+XCAL                                        MCSP0887
      RMEAN(L,2)=RMEAN(L,2)+YCAL                                        MCSP0888
      RMEAN(L,3)=RMEAN(L,3)+ZCAL                                        MCSP0889
      RMEAN(L,4)=RMEAN(L,4)+STATES(2,1)                                 MCSP0890
      RMEAN(L,5)=RMEAN(L,5)+STATES(N4,1)                                MCSP0891
      RMEAN(L,6)=RMEAN(L,6)+STATES(N5,1)                                MCSP0892
C                                                                       MCSP0893
C       SIMILARLY, PKSTOR'S ARE THE SUMS OF SQUARES OF ESTIMATION ERROR MCSP0894
```

```
C         AT EACH POINT                                                   MCSP0895
C                                                                         MCSP0896
      PKSTOR(1,L)=PKSTOR(1,L)+(XCAL-X)**2                                 MCSP0897
      PKSTOR(N2,L)=PKSTOR(N2,L)+(YCAL-Y)**2                               MCSP0898
      PKSTOR(N3,L)=PKSTOR(N3,L)+(ZCAL-Z)**2                               MCSP0899
      IF (KVEL.EQ.1) GO TO 1                                              MCSP0900
C                                                                         MCSP0901
C      IF TRUE VELOCITIES ARE NOT AVAILABLE THE VARIANCE ABOUT THE        MCSP0902
C         ESTIMATE, NOT TRUTH, WILL BE CALCULATED USING THE NEXT 3 CARDS  MCSP0903
C                                                                         MCSP0904
      PKSTOR(2,L)=PKSTOR(2,L)+STATES(2,1)**2                              MCSP0905
      PKSTOR(N4,L)=PKSTOR(N4,L)+STATES(N4,1)**2                           MCSP0906
      PKSTOR(N5,L)=PKSTOR(N5,L)+STATES(N5,1)**2                           MCSP0907
      RETURN                                                              MCSP0908
    1 PKSTOR(2,L)=PKSTOR(2,L)+(STATES(2,1)-TVELX)**2                      MCSP0909
      PKSTOR(N4,L)=PKSTOR(N4,L)+(STATES(N4,1)-TVELY)**2                   MCSP0910
      PKSTOR(N5,L)=PKSTOR(N5,L)+(STATES(N5,1)-TVELZ)**2                   MCSP0911
      RETURN                                                              MCSP0912
      END                                                                 MCSP0913


      SUBROUTINE GAIN (N,M,ITER,L,IFLAG,KVEL)                             MCSP0914
C                                                                         MCSP0915
C      THIS SUBROUTINE COMPUTES THE OPTIMUM GAIN MATRIX AND THE ERROR     MCSP0916
C         COVARIANCE                                                      MCSP0917
C                                                                         MCSP0918
      IMPLICIT REAL*8(P,Q,H,G,R,T)                                        MCSP0919
      REAL*8 II,AHEAD,EST                                                 MCSP0920
      DIMENSION TEMP(9,9), TEMP1(9,9), TEMP2(9,9), PHIT(9,9), HT(9,9), TMCSP0921
     1EMP3(9,9), TEMP4(9,9)                                               MCSP0922
      COMMON II(9,9),Q(9,9),H(9,9),R(9,9),G(9,9),PHI(9,9),PKK(9,9),PKKM1MCSP0923
     1(9,9),RMEAN(233,6),RSD(6),RESID(233,4),AHEAD(233),EST(9,233),PHIPRMCSP0924
     2M(9,9,40),PKSTOR(9,233)                                             MCSP0925
C                                                                         MCSP0926
C      THE FOLLOWING CARD PRECLUDES THE CALCULATION OF GAINS IF THE       MCSP0927
C         SYNTHETIC ESTIMATION INITIALIZATION IS USED                     MCSP0928
C                                                                         MCSP0929
      IF(IFLAG.EQ.1) RETURN                                               MCSP0930
C                                                                         MCSP0931
C                                                                         MCSP0932
C      G(K) = P(K/K-1)*HT*(H*P(K/K-1)*HT + R)                             MCSP0933
C                                                                         MCSP0934
C                                                                         MCSP0935
      DO 1 I=1,9                                                          MCSP0936
      DO 1 J=1,9                                                          MCSP0937
```

```
      PHIT(I,J)=0.D0                                                    MCSP0938
      HT(I,J)=0.D0                                                      MCSP0939
      TEMP(I,J)=0.D0                                                    MCSP0940
      TEMP1(I,J)=0.D0                                                   MCSP0941
      TEMP2(I,J)=0.D0                                                   MCSP0942
      TEMP3(I,J)=0.D0                                                   MCSP0943
    1 TEMP4(I,J)=0.D0                                                   MCSP0944
      CALL TRANS (H,M,N,HT)                                             MCSP0945
      CALL PROD (PKKM1,HT,N,N,M,TEMP)                                   MCSP0946
      CALL PROD (H,TEMP,M,N,M,TEMP1)                                    MCSP0947
      CALL ADD (TEMP1,R,M,M,TEMP3)                                      MCSP0948
      IF (M.EQ.1) GO TO 5                                               MCSP0949
      MD=9                                                              MCSP0950
      CALL GAUSS3 (3,EPS1,TEMP3,TEMP2,KER,MD)                           MCSP0951
      CALL PROD (TEMP,TEMP2,N,M,M,G)                                    MCSP0952
C                                                                       MCSP0953
C      NOTE HERE PKK(I,J) = P(K/K) WHERE                                MCSP0954
C        P(K/K) = (I-G(K)*H)*P(K/K-1)                                   MCSP0955
C                                                                       MCSP0956
    2 CALL PROD (G,H,N,M,N,TEMP3)                                       MCSP0957
      CALL SUB (II,TEMP3,N,N,TEMP4)                                     MCSP0958
      CALL PROD (TEMP4,PKKM1,N,N,N,PKK)                                 MCSP0959
C                                                                       MCSP0960
C      MCSP SEQUENCE NUMBERS INTERRUPTED                                MCSP0961
C                                                                       MCSP0965
C      NOTE HERE  PKKM1(I,J) = P(K/K-1) WHERE                           MCSP0966
C        P(K/K-1) = PHI*P(K-1/K-1)*PHIT + Q                             MCSP0967
C                                                                       MCSP0968
    4 CALL TRANS (PHI,N,N,PHIT)                                         MCSP0969
      CALL PROD (PKK,PHIT,N,N,N,TEMP3)                                  MCSP0970
      CALL PROD (PHI,TEMP3,N,N,N,TEMP4)                                 MCSP0971
      CALL ADD (TEMP4,Q,N,N,PKKM1)                                      MCSP0972
      RETURN                                                            MCSP0973
    5 DO 6 I=1,9                                                        MCSP0974
    6 G(I,1)=TEMP(I,1)/TEMP1(1,1)                                       MCSP0975
      GO TO 2                                                           MCSP0976
      END                                                               MCSP0977


      SUBROUTINE ADD (A,B,N,M,C)                                        MCSP0978
C                                                                       MCSP0979
C      THIS SUBROUTINE ADDS THE NXM MATRICES A AND B, STORING THE       MCSP0980
C        RESULT IN C                                                    MCSP0981
C                                                                       MCSP0982
      REAL*8 A,B,C                                                      MCSP0983
```

```
      DIMENSION A(9,9), B(9,9), C(9,9)                                  MCSP0984
      DO 1 I=1,N                                                        MCSP0985
      DO 1 J=1,M                                                        MCSP0986
    1 C(I,J)=A(I,J)+B(I,J)                                              MCSP0987
      RETURN                                                            MCSP0988
      END                                                               MCSP0989


      SUBROUTINE SUB (A,B,N,M,C)                                        MCSP0990
C                                                                       MCSP0991
C      THIS SUBROUTINE SUBTRACTS THE NXM MATRIX B FROM THE NXM MATRIX  MCSP0992
C        A AND STORES THE RESULT IN C                                   MCSP0993
C                                                                       MCSP0994
      REAL*8 A,B,C                                                      MCSP0995
      DIMENSION A(9,9), B(9,9), C(9,9)                                  MCSP0996
      DO 1 I=1,N                                                        MCSP0997
      DO 1 J=1,M                                                        MCSP0998
    1 C(I,J)=A(I,J)-B(I,J)                                              MCSP0999
      RETURN                                                            MCSP1000
      END                                                               MCSP1001


      SUBROUTINE PROD (A,B,N,M,L,C)                                     MCSP1002
C                                                                       MCSP1003
C      THIS SUBROUTINE COMPUTES THE MATRIX PRODUCT AB AND STORES THE   MCSP1004
C        RESULT IN C                                                    MCSP1005
C        A = NXM    B = MXL    C = NXL                                  MCSP1006
C                                                                       MCSP1007
      REAL*8 A,B,C                                                      MCSP1008
      DIMENSION A(9,9), B(9,9), C(9,9)                                  MCSP1009
      DO 1 I=1,N                                                        MCSP1010
      DO 1 J=1,L                                                        MCSP1011
    1 C(I,J)=0.D0                                                       MCSP1012
      DO 2 I=1,N                                                        MCSP1013
      DO 2 J=1,L                                                        MCSP1014
      DO 2 K=1,M                                                        MCSP1015
    2 C(I,J)=C(I,J)+A(I,K)*B(K,J)                                       MCSP1016
      RETURN                                                            MCSP1017
      END                                                               MCSP1018
```

```
      SUBROUTINE TRANS (A,N,M,C)                                        MCSP1019
C                                                                       MCSP1020
C      THIS SUBROUTINE FORMS THE MATRIX TRANSPOSE OF A STORING THE      MCSP1021
C        RESULT IN C                                                    MCSP1022
C        A = NXM     C = MXN                                            MCSP1023
C                                                                       MCSP1024
      REAL*8 A,C                                                        MCSP1025
      DIMENSION A(9,9), C(9,9)                                          MCSP1026
      DO 1 I=1,N                                                        MCSP1027
      DO 1 J=1,M                                                        MCSP1028
    1 C(J,I)=A(I,J)                                                     MCSP1029
      RETURN                                                            MCSP1030
      END                                                               MCSP1031


      SUBROUTINE MREAD (A,N,M)                                          MCSP1032
C                                                                       MCSP1033
C      THIS SUBROUTINE READS AN NXM MATRIX A ACCORDING TO THE FORMAT    MCSP1034
C      8D10.5.  THE ENTRIES IN THE FIRST ROW OF A ARE READ FIRST, THEN  MCSP1035
C      THE ENTRIES IN THE SECOND ROW, ETC.                              MCSP1036
C                                                                       MCSP1037
      REAL*8 A                                                          MCSP1038
      DIMENSION A(9,9)                                                  MCSP1039
      DO 1 I=1,N                                                        MCSP1040
    1 READ (5,2) (A(I,J),J=1,M)                                         MCSP1041
    2 FORMAT (8D10.5)                                                   MCSP1042
      RETURN                                                            MCSP1043
      END                                                               MCSP1044


      SUBROUTINE MREAD1 (A,N,M)                                         MCSP1045
C                                                                       MCSP1046
C      THIS SUBROUTINE READS AN NXM MATRIX A ACCORDING TO THE FORMAT    MCSP1047
C      10X,3D20.8).  THE ENTRIES IN THE FIRST ROW OF A ARE READ FIRST, TMCSP1048
C      THE ENTRIES IN THE SECOND ROW, ETC.                              MCSP1049
C                                                                       MCSP1050
      REAL*8 A                                                          MCSP1051
      DIMENSION A(9,9)                                                  MCSP1052
      DO 1 I=1,N                                                        MCSP1053
    1 READ (5,2) (A(I,J),J=1,M)                                         MCSP1054
    2 FORMAT (10X,3D20.8)                                               MCSP1055
      RETURN                                                            MCSP1056
```

```
      END                                                               MCSP1057


      SUBROUTINE MWRITE (A,N,M)                                         MCSP1058
C                                                                       MCSP1059
C      THIS SUBROUTINE WRITES THE ENTRIES OF THE NXM MATRIX A           MCSP1060
C                                                                       MCSP1061
      REAL*8 A                                                          MCSP1062
      DIMENSION A(9,9)                                                  MCSP1063
      DO 1 I=1,N                                                        MCSP1064
    1 WRITE (6,2) (I,J,A(I,J),J=1,M)                                    MCSP1065
    2 FORMAT (2(3X,'(',I1,',',I1,')= ',1PD19.12))                       MCSP1066
      RETURN                                                            MCSP1067
      END                                                               MCSP1068


      SUBROUTINE GAUSS3 (N,EPS1,A,X,KER,K)                              MCSP1069
C                                                                       MCSP1070
C      THIS SUBROUTINE INVERTS AN N X N MATRIX THAT HAS BEEN            MCSP1071
C        DIMENSIONED K X K IN THE CALLING PROGRAM                       MCSP1072
C                                                                       MCSP1073
      REAL*8 A,X,Y,D                                                    MCSP1074
      DIMENSION A(1), X(1), L(9), M(9), Y(9,9)                          MCSP1075
      DO 1 I=1,N                                                        MCSP1076
      DO 1 J=1,N                                                        MCSP1077
      IND=(I-1)*K+J                                                     MCSP1078
    1 Y(I,J)=A(IND)                                                     MCSP1079
C     KER = 1                                                           MCSP1080
      N2=2*N                                                            MCSP1081
      CALL ARRAY (2,N,N,9,9,Y,Y)                                        MCSP1082
      CALL DMINV (Y,N,D,L,M)                                            MCSP1083
      CALL ARRAY (1,N,N,9,9,Y,Y)                                        MCSP1084
      IF (D.EQ.0) KER=2                                                 MCSP1085
      DO 2 I=1,N                                                        MCSP1086
      DO 2 J=1,N                                                        MCSP1087
      IND=(I-1)*K+J                                                     MCSP1088
    2 X(IND)=Y(I,J)                                                     MCSP1089
      RETURN                                                            MCSP1090
      END                                                               MCSP1091
```

```
      SUBROUTINE ARRAY (MODE,I,J,N,M,S,D)                                MCSP1092
      DIMENSICN S(1), D(1)                                               MCSP1093
      REAL*8 S,D                                                         MCSP1094
      NI=N-I                                                             MCSP1095
      IF (MODE-1) 1,1,3                                                  MCSP1096
    1 IJ=I*J+1                                                           MCSP1097
      NM=N*J+1                                                           MCSP1098
      DC 2 K=1,J                                                         MCSP1099
      NM=NM-NI                                                           MCSP1100
      DC 2 L=1,I                                                         MCSP1101
      IJ=IJ-1                                                            MCSP1102
      NM=NM-1                                                            MCSP1103
    2 D(NM)=S(IJ)                                                        MCSP1104
      GO TO 6                                                            MCSP1105
    3 IJ=0                                                               MCSP1106
      NM=0                                                               MCSP1107
      DC 5 K=1,J                                                         MCSP1108
      DC 4 L=1,I                                                         MCSP1109
      IJ=IJ+1                                                            MCSP1110
      NM=NM+1                                                            MCSP1111
    4 S(IJ)=D(NM)                                                        MCSP1112
    5 NM = NM+NI                                                         MCSP1113
    6 RETURN                                                             MCSP1114
      END                                                                MCSP1115


      SUBROUTINE RNOISE (XM,RB1,RRO,RE1,N)                               MCSP1116
C                                                                        MCSP1117
C      THIS SUBROUTINE CALCULATES THE COVARIANCE CF MEASUREMENT NOISE   MCSP1118
C        MATRIX, R, FOR ONLINE CALCULATION IF REQUESTED                  MCSP1119
C                                                                        MCSP1120
C                                                                        MCSP1121
C       RB1,RR1,RE1 ARE INPUT AS STANDARD DEVIATICNS                     MCSP1122
C                                                                        MCSP1123
      IMPLICIT REAL*8(P,Q,H,G,R,T,B)                                     MCSP1124
      REAL*8 II,X,DUMMY,RAN,BEAR,ELEV,SINB,SINE,CCSB,COSE,SSQB,SSQE,CSQBMCSP1125
     1,CSQE,RSQ,XM,AHEAD,EST                                             MCSP1126
      COMMON II(9,9),Q(9,9),H(9,9),R(9,9),G(9,9),PHI(9,9),PKK(9,9),PKKM1MCSP1127
     1(9,9),RMEAN(233,6),RSD(6),RESID(233,4),AHEAD(233),EST(9,233),PHIPRMCSP1128
     2M(9,9,40),PKSTOR(9,233)                                            MCSP1129
      DIMENSICN X(9), XM(9,9)                                            MCSP1130
      DC 1 I=1,9                                                         MCSP1131
    1 X(I)=XM(I,1)                                                       MCSP1132
      PI=3.14159265358979323846                                          MCSP1133
      N2=N/3+1                                                           MCSP1134
```

```
      N3=N*2/3+1                                                        MCSP1135
      RAN=DSQRT(X(1)**2+X(N2)**2+X(N3)**2)                              MCSP1136
      RR1=RRO+.001DO*RAN                                                MCSP1137
      IF (X(1).EQ.0.0) GO TO 2                                          MCSP1138
      IF (X(N2).EQ.0.DO.AND.X(1).GT.0.DO) BEAR=0.CO                     MCSP1139
      IF (X(N2).EQ.0.DO.AND.X(1).LT.0.DO) BEAR=PI                       MCSP1140
      IF (X(N2).NE.0.DO) BEAR=DATAN(X(N2)/X(1))                         MCSP1141
      GO TO 3                                                           MCSP1142
    2 IF (X(N2).GT.0.0) BEAR = PI/2.0                                   MCSP1143
      IF (X(N2).LT.0.0) BEAR=3.*PI/2.                                   MCSP1144
      IF (X(N2).EQ.0.0) GO TO 4                                         MCSP1145
    3 DUMMY=X(N3)/RAN                                                   MCSP1146
      GO TO 5                                                           MCSP1147
    4 BEAR=0.DO                                                         MCSP1148
      DUMMY=1.DO                                                        MCSP1149
    5 IF (DUMMY.EQ.1.0) ELEV=PI/2.0                                     MCSP1150
      IF (DUMMY.NE.1.0) ELEV=DATAN(DUMMY/DSQRT(1.-DUMMY**2))            MCSP1151
      SINB=DSIN(BEAR)                                                   MCSP1152
      SINE=DSIN(ELEV)                                                   MCSP1153
      COSB=DCOS(BEAR)                                                   MCSP1154
      COSE=DCOS(ELEV)                                                   MCSP1155
      SSQB=SINB*SINB                                                    MCSP1156
      SSQE=SINE*SINE                                                    MCSP1157
      CSQB=COSB*COSB                                                    MCSP1158
      CSQE=COSE*COSE                                                    MCSP1159
      RSQ=RAN*RAN                                                       MCSP1160
      RB=RB1*RB1                                                        MCSP1161
      RR=RR1*RR1                                                        MCSP1162
      RE=RE1*RE1                                                        MCSP1163
      R(1,1)=RSQ*RE*CSQB*SSQE+RSQ*RB*SSQB*CSQE+RSQ*RB*RE*SSQB*SSQE+RR*CSMCSP1164
     1QB*CSQE                                                           MCSP1165
      R(1,2)=RSQ*COSB*SINB*SSQE*(RE-RB*RE)+SINB*COSB*CSQE*(RR-RSQ*RB)   MCSP1166
      R(2,1)=R(1,2)                                                     MCSP1167
      R(2,2)=RSQ*RE*SSQB*SSQE+RSQ*RB*CSQB*CSQE+RSQ*RB*RE*CSQB*SSQE+RR*SSMCSP1168
     1QB*CSQE                                                           MCSP1169
      R(2,3)=SINB*SINE*COSE*(RR-RSQ*RE)                                 MCSP1170
      R(3,2)=R(2,3)                                                     MCSP1171
      R(1,3)=COSE*SINE*COSB*(RR-RSQ*RE)                                 MCSP1172
      R(3,1)=R(1,3)                                                     MCSP1173
      R(3,3)=RSQ*CSQE*RE+SSQE*RR                                        MCSP1174
      RETURN                                                            MCSP1175
      END                                                               MCSP1176


      SUBROUTINE ACCUR (X,Y,Z,L,N,ITER)                                 MCSP1177
```

```
C
C     THIS SUBROUTINE CALCULATES THE SHELL AT TARGET RESIDUALS IF
C       REQUESTED
C
      IMPLICIT REAL*8(Q,H,R,G,P,A,B,T,S,X,Y,Z,E)
      REAL*8 DIFF,II
      DIMENSION A(9,9), TEMP(9,9), TRY(9,9)
      COMMON II(9,9),Q(9,9),H(9,9),R(9,9),G(9,9),PHI(9,9),PKK(9,9),PKKM1
     1(9,9),RMEAN(233,6),RSD(6),RESID(233,4),AHEAD(233),EST(9,233),PHIPR
     2M(9,9,40),PKSTOR(9,233)
      DIFF=FLOAT(L)-4.DO*AHEAD(L)
      LESS=DIFF
      DO 2 I=1,9
      DO 1 J=2,9
    1 A(I,J)=0.DO
    2 A(I,1)=EST(I,LESS)
      IAHEAD=AHEAD(L)*4.DO+1.DO
      DO 3 I=1,9
      DO 3 J=1,9
    3 TEMP(I,J)=PHIPRM(I,J,IAHEAD)
      CALL PROD (TEMP,A,N,N,1,TRY)
      N2=N/3+1
      N3=N*2/3+1
      XCAL=TRY(1,1)
      YCAL=TRY(N2,1)
      ZCAL=TRY(N3,1)
      RESID(L,1)=RESID(L,1)+(XCAL-X)
      RESID(L,2)=RESID(L,2)+(YCAL-Y)
      RESID(L,3)=RESID(L,3)+(ZCAL-Z)
      RETURN
      END


      SUBROUTINE STUDY (I,X,Y,Z,RESIDX,RESIDY,RESIDZ,AHEAD,RMISS)
C
C     THIS ROUTINE OUTPUTS THE SHELL AT TARGET RESIDUAL
C       PERFORMANCE TABLE IF REQUESTED
C
      REAL*8 X,Y,Z,RESIDX,RESIDY,RESIDZ,CALX,CALY,CALZ,AHEAD,TIME,RMISS
      CALX=X+RESIDX
      CALY=Y+RESIDY
      CALZ=Z+RESIDZ
      TIME=FLOAT(I)-AHEAD*4.DO
      WRITE (6,1) I,X,CALX,RESIDX,AHEAD
      WRITE (6,2) Y,CALY,RESIDY,RMISS
```

```
      WRITE (6,3) Z,CALZ,RESIDZ,TIME                                      MCSP1221
    1 FORMAT (1H0,I4,' X=',3D15.6,D14.5)                                  MCSP1222
    2 FORMAT (1H ,5X,'Y=',3D15.6,'  ----------  ',D15.6)                  MCSP1223
    3 FORMAT (1H ,5X,'Z=',3D15.6,D14.5,/)                                 MCSP1224
      RETURN                                                              MCSP1225
      END                                                                 MCSP1226
```

APPENDIX D

LISTING OF SHELL FLIGHT TIME INTERPOLATION PROGRAM

The program that interpolates in the 5 in - 54 gunfire table (included after the program listing) to generate the shell-at-target flight times is included in this appendix. In its present form it can provide times for tracks operating in a region bounded by

minimum range	- 500	yards
maximum range	-7500	yards
minimum elevation angle	- 15	degrees
minimum elevation angle	- 90	degrees.

If track data points fall outside of these limits the program can be easily expanded by enlarging the arrays and revising the linear interpolation equations.

```
C
C     THIS PROGRAM GENERATES THE SHELL FLIGHT TIMES TO POINTS ON A
C       TRACK BY INTERPOLATING IN A 5 IN - 54 GUNFIRE TABLE.
C       BOTH THE INTERPOLATION ARRAY AND THE TRACK DATA ARE INPUT,
C       AND THE FLIGHT TIMES ARE PUNCHED.
C
      REAL*8 X,Y,Z,R,E,ARRAY,TIME,RSTEP,ESTEP,RSTEP1,ESTEP1,T1,T2,FRAC
      DIMENSION ARRAY(15,16)
C
C      READ THE INTERPOLATION ARRAY BY ROWS
C
      READ(5,100) ((ARRAY(I,J),J=2,16),I=1,15)
  100 FORMAT(8D10.0,/,7D10.0)
C
C      THE FOLLOWING READ STATEMENT IS A SUPPLEMENT TO READ
C        THE FIRST COLUMN
C
      READ(5,100) (ARRAY(I,1),I=1,15)
      WRITE(6,200) ((ARRAY(I,J),J=1,8),I=1,15)
  200 FORMAT(1H1,10X,'INTERPOLATION ARRAY',//,(2X,8D16.4))
      WRITE(6,300) ((ARRAY(I,J),J=9,16),I=1,15)
  300 FORMAT(1H0,//,(2X,8D16.4))
      WRITE(6,50)
   50 FORMAT(1H1,14X,'X',14X,'Y',14X,'Z',14X,'R',14X,'E',12X,
     1 'TIME',10X,'CL TIME',9X,'STEPS',//)
      DO 1000 I=1,233
C
C      READ THE INPUT TRACK DATA
C
      READ(5,150) X,Y,Z
  150 FORMAT(3X,3D20.0)
      R=DSQRT(X**2+Y**2+Z**2)
      E=DATAN2(Z,DSQRT(X**2+Y**2))
      E=E*360.D0/(2.D0*3.1415926539D0)
      RSTEP = 500.D0
      ESTEP = 15.D0
      INDR = 1
      INDE = 1
    6 IF(R.LT.RSTEP) GO TO 7
      RSTEP = RSTEP+500.D0
      INDR = INDR+1
      GO TO 6
    7 RSTEP1=  RSTEP-500.D0
    8 IF(E.LT.ESTEP) GO TO 10
      ESTEP = ESTEP+5.D0
      INDE = INDE+1
      GO TO 8
   10 ESTEP1 = ESTEP-5.D0
```

```
      FRAC = (R-(INDR-1)*500.D0)/500.D0
      T1 = ARRAY(INDR-1,INDE-1)+FRAC*(ARRAY(INDR,INDE-1)-ARRAY(INDR-1,
     1 INDE-1))
      T2 = ARRAY(INDR-1,INDE)+FRAC*(ARRAY(INDR,INDE)-ARRAY(INDR-1,INDE))
C
C       THE FLIGHT TIME IS CALCULATED IN SECS
C
      TIME = T1+(E-ESTEP1       )/5.D0*(T2-T1)
      WRITE(6,51) I,X,Y,Z,R,E,TIME
   51 FORMAT(1H0,I3,6D15.6,5X,'__________',10X,'__________')
 1000 CONTINUE
      STOP
      END
```

5 INCH 54 CALIBER GUN I.V. 2500 F.S.

TIME OF FLIGHT IN SECONDS

POSITION ANGLE IN DEGREES

SLANT RANGE YARDS	0	5	10	15	20	25	30	35	40	45	50	55	60	65	70	75	80	85	90
500	0.61	0.61	0.61	0.61	0.61	0.61	0.61	0.61	0.61	0.61	0.61	0.61	0.61	0.61	0.61	0.61	0.61	0.61	0.61
1000	1.24	1.24	1.24	1.24	1.24	1.24	1.24	1.24	1.24	1.24	1.24	1.24	1.24	1.24	1.24	1.24	1.24	1.25	1.25
1500	1.88	1.88	1.89	1.89	1.89	1.89	1.89	1.89	1.90	1.90	1.90	1.90	1.90	1.90	1.90	1.90	1.90	1.90	1.90
2000	2.55	2.55	2.56	2.56	2.56	2.56	2.57	2.57	2.57	2.57	2.58	2.58	2.58	2.58	2.58	2.58	2.58	2.58	2.58
2500	3.24	3.24	3.25	3.25	3.25	3.26	3.26	3.27	3.27	3.27	3.28	3.28	3.28	3.29	3.29	3.29	3.29	3.29	3.29
3000	3.95	3.95	3.96	3.97	3.97	3.98	3.98	3.99	4.00	4.00	4.01	4.01	4.01	4.02	4.02	4.02	4.02	4.02	4.02
3500	4.68	4.69	4.70	4.71	4.71	4.72	4.73	4.74	4.75	4.75	4.76	4.76	4.77	4.77	4.78	4.78	4.78	4.78	4.78
4000	5.44	5.45	5.46	5.47	5.48	5.49	5.50	5.51	5.52	5.53	5.54	5.55	5.55	5.56	5.56	5.57	5.57	5.57	5.57
4500	6.22	6.24	6.25	6.26	6.28	6.29	6.30	6.31	6.33	6.34	6.35	6.36	6.37	6.37	6.38	6.38	6.39	6.39	6.39
5000	7.04	7.05	7.07	7.08	7.10	7.11	7.13	7.14	7.16	7.17	7.19	7.20	7.21	7.22	7.23	7.23	7.24	7.24	7.24
5500	7.88	7.89	7.91	7.93	7.95	7.97	7.99	8.01	8.02	8.04	8.06	8.07	8.08	8.09	8.10	8.11	8.12	8.12	8.12
6000	8.75	8.77	8.79	8.81	8.83	8.85	8.87	8.90	8.92	8.94	8.96	8.98	8.99	9.01	9.02	9.03	9.03	9.04	9.04
6500	9.65	9.67	9.70	9.72	9.74	9.77	9.80	9.82	9.85	9.87	9.89	9.91	9.93	9.95	9.96	9.98	9.98	9.99	9.99
7000	10.59	10.61	10.64	10.66	10.69	10.72	10.75	10.78	10.81	10.84	10.87	10.89	10.91	10.93	10.95	10.96	10.97	10.98	10.98
7500	11.56	11.59	11.61	11.64	11.67	11.71	11.74	11.78	11.81	11.84	11.88	11.91	11.93	11.96	11.98	11.99	12.01	12.01	12.02
8000	12.57	12.60	12.63	12.66	12.69	12.73	12.77	12.81	12.85	12.89	12.93	12.96	13.00	13.02	13.05	13.07	13.08	13.09	13.09
8500	13.62	13.65	13.68	13.71	13.75	13.79	13.84	13.88	13.93	13.98	14.02	14.06	14.10	14.14	14.17	14.19	14.21	14.22	14.22
9000	14.71	14.74	14.77	14.81	14.85	14.90	14.95	15.00	15.06	15.11	15.16	15.21	15.26	15.30	15.34	15.36	15.38	15.40	15.40
9500	15.84	15.87	15.90	15.95	15.99	16.05	16.11	16.17	16.23	16.29	16.36	16.42	16.47	16.52	16.56	16.59	16.62	16.63	16.64
10000	17.02	17.05	17.08	17.13	17.18	17.24	17.31	17.38	17.46	17.53	17.60	17.67	17.74	17.80	17.85	17.89	17.92	17.94	17.94
10500	18.25	18.27	18.31	18.36	18.42	18.49	18.57	18.65	18.74	18.83	18.91	19.00	19.07	19.14	19.20	19.25	19.29	19.31	19.32
11000	19.53	19.55	19.59	19.64	19.71	19.79	19.88	19.98	20.08	20.18	20.29	20.39	20.48	20.57	20.64	20.70	20.74	20.77	20.78
11500	20.86	20.88	20.92	20.98	21.06	21.15	21.25	21.37	21.49	21.61	21.74	21.86	21.97	22.07	22.16	22.23	22.29	22.32	22.33
12000	22.23	22.26	22.30	22.37	22.46	22.57	22.69	22.83	22.97	23.12	23.27	23.41	23.55	23.67	23.78	23.87	23.93	23.97	23.99
12500	23.66	23.69	23.74	23.82	23.92	24.05	24.20	24.36	24.53	24.71	24.89	25.06	25.22	25.37	25.50	25.61	25.69	25.74	25.75
13000	25.13	25.16	25.23	25.33	25.45	25.60	25.77	25.97	26.17	26.38	26.60	26.81	27.01	27.19	27.35	27.48	27.58	27.64	27.67
13500	26.62	26.68	26.77	26.88	27.03	27.21	27.42	27.65	27.89	28.15	28.41	28.67	28.92	29.15	29.35	29.52	29.65	29.73	29.75
14000	28.15	28.23	28.34	28.49	28.67	28.89	29.14	29.41	29.71	30.03	30.35	30.67	30.99	31.29	31.55	31.77	31.94	32.05	32.08
14500	29.71	29.82	29.96	30.15	30.37	30.63	30.93	31.27	31.64	32.03	32.44	32.86	33.27	33.67	34.02	34.32	34.56	34.70	34.75
15000	31.30	31.44	31.63	31.85	32.13	32.45	32.82	33.24	33.70	34.20	34.73	35.29	35.85	36.39	36.89	37.33	37.68	37.90	37.97
15500	32.92	33.11	33.33	33.62	33.95	34.35	34.81	35.34	35.93	36.59	37.30	38.06	38.86	39.68	40.47	41.20	41.80	42.21	42.35
16000	34.58	34.31	35.09	35.44	35.86	36.35	36.93	37.60	38.38	39.27	40.27	41.41	42.70	44.17	45.91	48.26			
16500	36.27	36.56	36.91	37.33	37.85	38.47	39.21	40.10	41.15	42.42	43.98	46.01	49.31						
17000	38.01	38.36	38.79	39.31	39.95	40.74	41.70	42.90	44.43	46.47	49.76								
17500	39.78	40.21	40.73	41.38	42.18	43.20	44.50	46.22	48.73										
18000	41.61	42.12	42.76	43.56	44.58	45.93	47.77												
18500	43.48	44.11	44.88	45.88	47.21	49.07													
19000	45.42	46.17	47.13	48.39															
19500	47.44	48.34	49.52																
20000	49.53	48.67																	

APPENDIX E

SAMPLE SIMULATION

A sample problem is included to display a representative input data set and the output that can be expected from the MCSP. A description of the desired simulation follows:

Number of Monte-Carlo runs - 100,

Track has 233 points with 4-Hz, sampling rate with approximate velocity data available,

Direction of approach - 45°,

Order of filter - 6,

Initialization by synthetic measurements is desired,

Adaptive $\underline{Q}$ with $K_1 = .9$, $K_2 = .1$,

Noise statistics are $\sigma_R = 5.0 + .001 \text{ X Range}$

$\sigma_B = \sigma_E = .002$ radians,

Shell flight times are available, and shell-at-target accuracy calculations are desired,

$\underline{\phi}$, $\underline{\Gamma}$, and $\underline{H}$ are standard 6th order matrices,

Desired output includes all available plots.

An abridged listing of the input data appears next followed by representative portions of the output produced.

```
NOTE:  ALL CARDS THAT ARE NOT ACTUAL DATA CARDS BUT FOR CLARIFICATION
       HAVE ********** IN THE FIRST 10 COLUMNS
**********
**********
**********         COLUMN HEADINGS THAT FOLLOW ARE FOR CLARIFICATION
123456789 123456789 123456789 123456789 123456789 123456789 123456789 123456789
**********
**********
       100         1          11111111111111          .25D0
    6  233    1    0    0    1    0    0  ADAPTIVE Q - THESIS SAMPLE
  5.D0           .002D0        .002D0         .90D0          .10D0
    1    1    1
   0          0          0
   0          0          0
   0          0          0
   0          0          0
   0          0          0
   0          0          0
   0          0          0
   0          0          0
   0          0          0
  1.D0    0.25D0       0         0         0         0
   0      1.D0         0         0         0         0
   0         0       1.D0      0.25D0      0         0
   0         0         0       1.D0        0         0
   0         0         0         0       1.D0      0.25D0
   0         0         0         0         0       1.D0
  1.D0       0         0         0         0         0
   0         0       1.D0        0         0         0
   0         0         0         0       1.D0        0
       1    0.5750000000D 04    0.0                 0.4333333333D 04
       1   -0.1391666667D 03    0.0                 0.0
       2    0.5715211093D 04    0.0                 0.4332953830D 04
       2   -0.1391335488D 03    0.0                -0.3035904834D 01
       3    0.5680438744D 04    0.0                 0.4331815501D 04
       3   -0.1390342108D 03    0.0                -0.6070364742D 01
       4    0.5645699502D 04    0.0                 0.4329918889D 04
       4   -0.1388687002D 03    0.0                -0.9101935485D 01
       5    0.5611009902D 04    0.0                 0.4327264895D 04
       5   -0.1386370955D 03    0.0                -0.1212917420D 02
       6    0.5576386453D 04    0.0                 0.4323854783D 04
       6   -0.1383395071D 03    0.0                -0.1515064008D 02
       7    0.5541845635D 04    0.0                 0.4319690175D 04
       7   -0.1379760765D 03    0.0                -0.1816489508D 02
       8    0.5507403886D 04    0.0                 0.4314773055D 04
       8   -0.1375469769D 03    0.0                -0.2117050458D 02
       9    0.5473077602D 04    0.0                 0.4309105763D 04
       9   -0.1370524123D 03    0.0                -0.2416603806D 02
```

```
      10    0.5438883117D 04    0.0                  0.4302690995D 04
      10   -0.1364926182D 03    0.0                 -0.2715006981D 02
**********
**********
**********          TRACK DATA CONTINUES IN SAME FCRMAT
**********
**********
     230   -0.5170291629D 04   -0.1489867376D 04    0.1832311536D 04
     230   -0.2276785160D 03   -0.9307867866D 02    0.4337111381D 02
     231   -0.5228338731D 04   -0.1512873277D 04    0.1843097838D 04
     231   -0.2296257805D 03   -0.8816545797D 02    0.4337111381D 02
     232   -0.5286866722D 04   -0.1534626708D 04    0.1853871649D 04
     232   -0.2314670337D 03   -0.8321153392D 02    0.4337111381D 02
     233   -0.5345848581D 04   -0.1555117629D 04    0.1864631726D 04
     233   -0.2332014254D 03   -0.7821919359D 02    0.4337111381D 02
      0.111920540 02
      0.111370800 02
      0.110813100 02
      0.110247610 02
      0.109674520 02
      0.109093990 02
      0.108506220 02
      0.107914330 02
      0.107337050 02
      0.106753370 02
**********
**********
**********          SHELL FLIGHT TIMES CONTINUE IN SAME FCRMAT
**********
**********
      0.816023150 01
      0.826923710 01
      0.837852630 01
      0.848804230 01
      0.859773040 01
**********
**********
**********          END OF DATA SET FOR THIS EXAMPLE
```

```
ADAPTIVE 6 - THESIS SAMPLE
      ORDER OF FILTER=    6

      NUMBER OF MONTE CARLO RUNS= 100

      CODE FOR DIRECTION OF APPROACH=    1

      TRANSLATION OF X DATA POINTS=        0.0
                     Y DATA POINTS=        0.0
                     Z DATA POINTS=        0.0

      NOISE CODE=  1


PHI MATRIX

  (1,1)=  1.000000000000D 00    (1,2)=  2.500000000000D-01
  (1,3)=  0.0                   (1,4)=  0.0
  (1,5)=  0.0                   (1,6)=  0.0
  (2,1)=  0.0                   (2,2)=  1.000000000000D 00
  (2,3)=  0.0                   (2,4)=  0.0
  (2,5)=  0.0                   (2,6)=  0.0
  (3,1)=  0.0                   (3,2)=  0.0
  (3,3)=  1.000000000000D 00    (3,4)=  2.500000000000D-01
  (3,5)=  0.0                   (3,6)=  0.0
  (4,1)=  0.0                   (4,2)=  0.0
  (4,3)=  0.0                   (4,4)=  1.000000000000D 00
  (4,5)=  0.0                   (4,6)=  0.0
  (5,1)=  0.0                   (5,2)=  0.0
  (5,3)=  0.0                   (5,4)=  0.0
  (5,5)=  1.000000000000D 00    (5,6)=  2.500000000000D-01
  (6,1)=  0.0                   (6,2)=  0.0
  (6,3)=  0.0                   (6,4)=  0.0
  (6,5)=  0.0                   (6,6)=  1.000000000000D 00

H OR MEASUREMENT MATRIX

  (1,1)=  1.000000000000D 00    (1,2)=  0.0
  (1,3)=  0.0                   (1,4)=  0.0
  (1,5)=  0.0                   (1,6)=  0.0
  (2,1)=  0.0                   (2,2)=  0.0
  (2,3)=  1.000000000000D 00    (2,4)=  0.0
  (2,5)=  0.0                   (2,6)=  0.0
  (3,1)=  0.0                   (3,2)=  0.0
  (3,3)=  0.0                   (3,4)=  0.0
  (3,5)=  1.000000000000D 00    (3,6)=  0.0

GAMMA MATRIX

  (1,1)=  0.0                   (1,2)=  0.0
  (1,3)=  0.0                   (
  (2,1)=  0.0                   (2,2)=  0.0
  (2,3)=  0.0                   (
  (3,1)=  0.0                   (3,2)=  0.0
  (3,3)=  0.0                   (
  (4,1)=  0.0                   (4,2)=  0.0
  (4,3)=  0.0                   (
  (5,1)=  0.0                   (5,2)=  0.0
  (5,3)=  0.0                   (
  (6,1)=  0.0                   (6,2)=  0.0
  (6,3)=  0.0                   (
```

```
P(K/K-1) OR PREOICTION COVARIANCE MATRIX

  (1,1)=  0.0                      (1,2)=  0.0
  (1,3)=  0.0                      (1,4)=  0.0
  (1,5)=  0.0                      (1,6)=  0.0
  (2,1)=  0.0                      (2,2)=  0.0
  (2,3)=  0.0                      (2,4)=  0.0
  (2,5)=  0.0                      (2,6)=  0.0
  (3,1)=  0.0                      (3,2)=  0.0
  (3,3)=  0.0                      (3,4)=  0.0
  (3,5)=  0.0                      (3,6)=  0.0
  (4,1)=  0.0                      (4,2)=  0.0
  (4,3)=  0.0                      (4,4)=  0.0
  (4,5)=  0.0                      (4,6)=  0.0
  (5,1)=  0.0                      (5,2)=  0.0
  (5,3)=  0.0                      (5,4)=  0.0
  (5,5)=  0.0                      (5,6)=  0.0
  (6,1)=  0.0                      (6,2)=  0.0
  (6,3)=  0.0                      (6,4)=  0.0
  (6,5)=  0.0                      (6,6)=  0.0

INITIAL COVARIANCES OF RANOOM FORCING MATRIX OF
STATE EXCITATION FOR AOAPTIVE Q FILTER

WEIGHTING FACTORS FCR C MATRIX UPOATE ARE - K1=        0.9000000 00
                                            K2=        0.1000000 00

  (1,1)=  0.0                      (1,2)=  0.0
  (1,3)=  0.0                      (
  (2,1)=  0.0                      (2,2)=  0.0
  (2,3)=  0.0                      (
  (3,1)=  0.0                      (3,2)=  0.0
  (3,3)=  0.0                      (

R CR COVARIANCE OF MEAS. ERROR MATRIX GENERATEO BY PROGRAM

         RANGE STAN OEV,RR=  5.0000000000000 00
         BEARING STAN OEV,RB=  2.0000000000000-03
         ELEVATION STAN OEV,RE=  2.0000000000000-03

THE FIRST  2 ESTIMATES OF POSITION WILL BE THE OBSERVEO POSITION

STATE INITIAL CONOITIONS ARE

  (1,1)=  4.0727172264900 03     (
  (2,1)=  0.0                    (
  (3,1)=  4.0632459995820 03     (
  (4,1)=  0.0                    (
  (5,1)=  4.3510848373670 03     (
  (6,1)=  0.0                    (
```

PCINT NUMBER		TRUE TRACK PDINTS	FILTER PRED POINTS	RESIDUALS	TIME OF FLT / TRACK TIME WHEN FIRED	MEAN MISS DISTANCE
37	X=	0.326415D 04	0.326289D 04	-0.125767D 01	0.89630D 01	
	Y=	0.326415D 04	0.200396D 04	-0.126019D 04	----------	D.151564D 04
	Z=	0.386900D 04	0.471108D 04	D.842077D 03	D.11482D D1	
38	X=	C.324446D 04	D.320473D 04	-0.397256D 02	D.88961D D1	
	Y=	0.324446D 04	0.317025D 04	-0.742033D D2	----------	D.4169D3D 03
	Z=	0.384564D 04	D.425396D 04	0.408318D 03	0.24154D 01	
39	X=	C.322458D 04	D.305646D 04	-D.168118D 03	0.88301D D1	
	Y=	0.322458D 04	0.282690D 04	-D.397681D 03	----------	0.65D739D D3
	Z=	0.382206D 04	0.430893D 04	0.486876D D3	D.36796D 01	
40	X=	C.320452D 04	0.305947D 04	-0.145048D 03	0.87634D 01	
	Y=	0.320452D 04	0.233172D 04	-0.872796D 03	----------	D.114916D 04
	Z=	0.379825D 04	0.453156D 04	0.733315D 03	0.49463D D1	
41	X=	0.318427D 04	0.295278D 04	-0.231485D 03	D.86961D D1	
	Y=	0.318427D 04	0.214959D 04	-0.103468D 04	----------	D.134310D D4
	Z=	0.377422D 04	0.459870D 04	0.824478D 03	0.62154D D1	
42	X=	0.316383D 04	0.283771D 04	-0.326121D 03	0.86282D D1	
	Y=	0.316383D 04	0.212337D 04	-0.104046D 04	----------	D.135560D D4
	Z=	0.374997D D4	0.455540D 04	0.805432D 03	0.74870D D1	
43	X=	C.314321D 04	0.278787D 04	-0.355335D 03	0.85597D D1	
	Y=	C.314321D 04	0.221997D 04	-0.923242D 03	----------	D.124153D 04
	Z=	0.372549D 04	0.447567D 04	0.750179D 03	0.87612D D1	
44	X=	0.312240D 04	0.270835D 04	-0.414049D D3	D.84906D D1	
	Y=	0.312240D 04	0.224469D 04	-0.877706D 03	----------	D.117076D 04
	Z=	C.370080D 04	0.435568D 04	0.654886D D3	0.10038D D2	

********** PERFORMANCE TABLE OUTOUT CONTINUES FOR REMAINING TRACK POINTS **********

AVERAGE SHELL AT TARGET MISS DISTANCE = 0.18277774D 03 YARDS

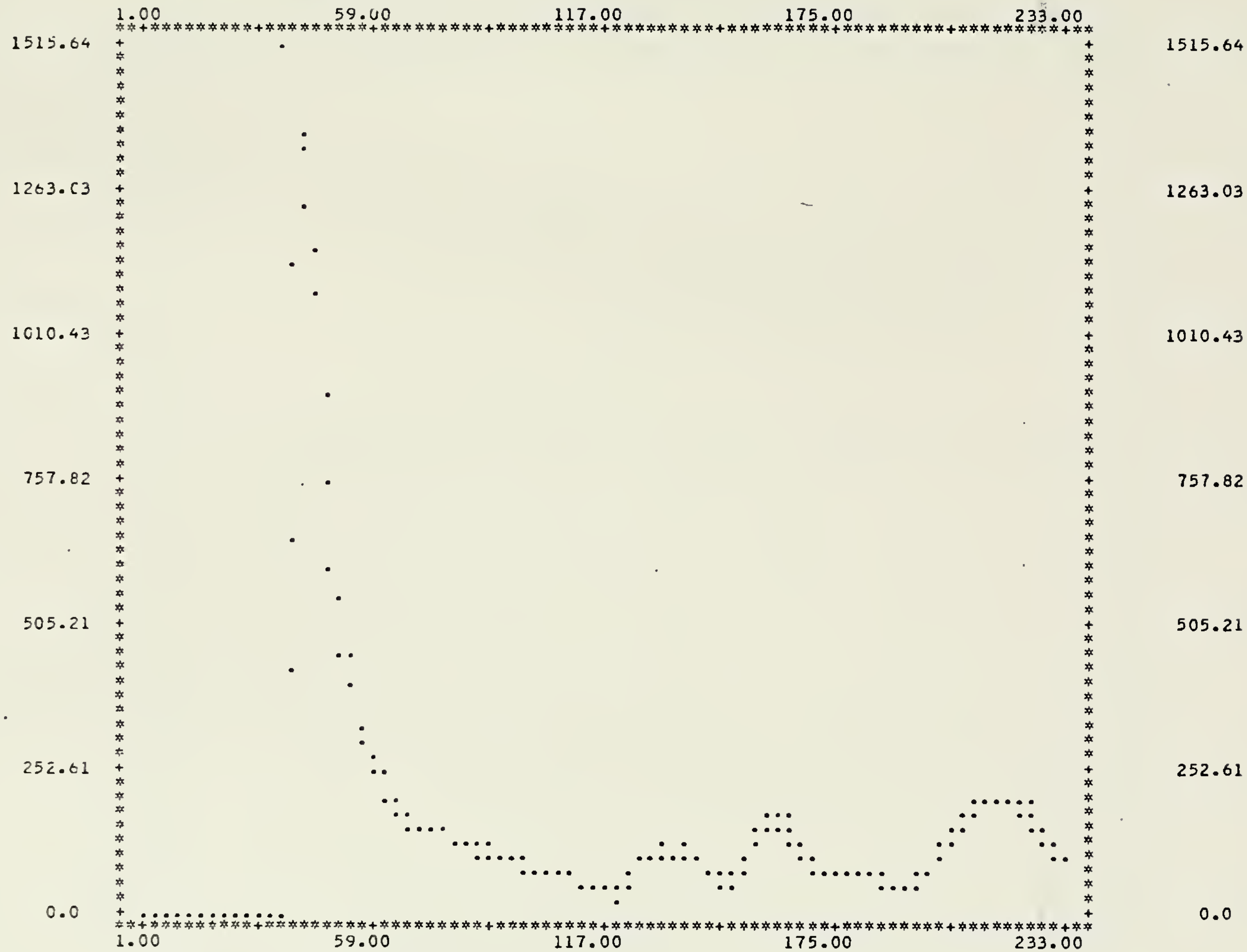
PERFCRMANCE FACTCR CF FILTER, AVERAGE FILTER PDSITION ESTIMATE ERROR = 0.26512609D 01 YARDS

AVERAGE FILTER VELCCITY ESTIMATE ERROR = 0.22880825D 02 YARDS/SEC

POINT NO		POSITION ESTIMATE ERRORS		POSITION VARIANCES	VELOCITY /ERRORS		VELOCITY VARIANCES AROUND ESTIMATE/ AROUND TRUTH
1	X:	-0.586950D 00	X:	0.1264470 03	0.9251410 01	X:	0.100071D 03
	Y:	0.887720D 00	Y:	0.1589200 03	-0.1307940 03	Y:	0.5753950 03
	Z:	0.266785D 01	Z:	0.1409170 03	0.4167560 02	Z:	0.400377D 02
2	X:	0.754557D 00	X:	0.1515110 03	0.5350410 01	X:	0.4147960 04
	Y:	0.1257540 01	Y:	0.1315970 03	0.1463670 01	Y:	0.464109D 04
	Z:	0.8300160 00	Z:	0.1596390 03	-0.5833430 01	Z:	0.4518960 04
3	X:	0.7057070 00	X:	0.5628680 03	-0.8456680 01	X:	0.3795090 04
	Y:	0.1470990 01	Y:	0.5842220 03	-0.3404860 02	Y:	0.8318880 04
	Z:	-0.3008100 01	Z:	0.5516390 03	0.3861960 01	Z:	0.5082250 04
4	X:	0.1747570 01	X:	0.5450270 03	-0.562593D 01	X:	0.1923290 04
	Y:	-0.9144870 01	Y:	0.2939770 03	-0.8527660 02	Y:	0.972173D 04
	Z:	0.1500260 00	Z:	0.7351480 03	0.3149020 02	Z:	0.2291400 04
5	X:	0.614489D 01	X:	0.3272940 03	-0.8143510 01	X:	0.1897520 04
	Y:	-0.1184530 02	Y:	0.183775D 03	-0.1035920 03	Y:	0.1131200 05
	Z:	0.6170080 01	Z:	0.5795850 03	0.455099D 02	Z:	0.2083000 04
6	X:	0.7955340 01	X:	0.1883330 03	-0.162674D 02	X:	0.2371090 04
	Y:	-0.1154870 02	Y:	0.2489670 03	-0.1058320 03	Y:	0.1567610 05
	Z:	0.9196870 01	Z:	0.5278870 03	0.4551000 02	Z:	0.231941D 04
7	X:	0.764069D 01	X:	0.1438700 03	-0.2684820 02	X:	0.1768330 04
	Y:	-0.8484210 01	Y:	0.1250970 03	-0.1066440 03	Y:	0.1290470 05
	Z:	0.9774840 01	Z:	0.3943640 03	0.4398590 02	Z:	0.1874850 04
8	X:	0.620506D 01	X:	0.1343870 03	-0.2990000 02	X:	0.2034090 04
	Y:	-0.420654D 01	Y:	0.1444630 03	-0.9361390 02	Y:	0.1340990 05
	Z:	0.875707D 01	Z:	0.3234990 03	0.3855810 02	Z:	0.1934440 04
9	X:	0.3108800 01	X:	0.136575D 03	-0.3800480 02	X:	0.2036340 04
	Y:	-0.1214540 01	Y:	0.1237680 03	-0.9702130 02	Y:	0.1147520 05
	Z:	0.718876D 01	Z:	0.1886110 03	0.389531D 02	Z:	0.1995420 04
10	X:	0.1329180 01	X:	0.9378140 02	-0.3746770 02	X:	0.183592D 04
	Y:	-0.1445250 01	Y:	0.1117670 03	-0.9168930 02	Y:	0.9681160 04
	Z:	0.492281D 01	Z:	0.1326700 03	0.3280510 02	Z:	0.1624800 04

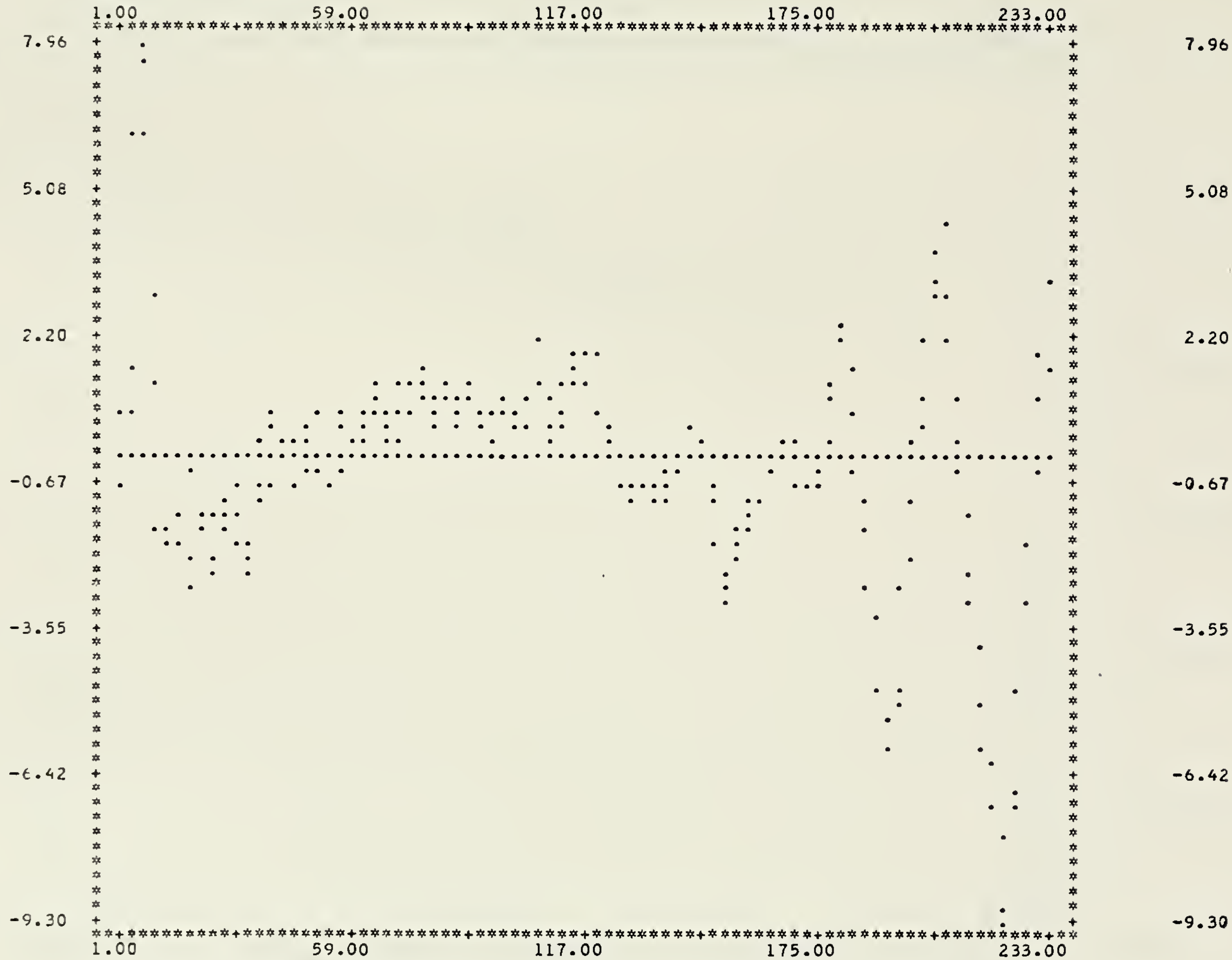
********** FILTER ERROR TABLE CONTINUES FOR REMAINING TRACK POINTS **********

KETRON MK-86 THESIS
MEAN MISS DISTANCE WHEN SHELL ARRIVES IN YDS.



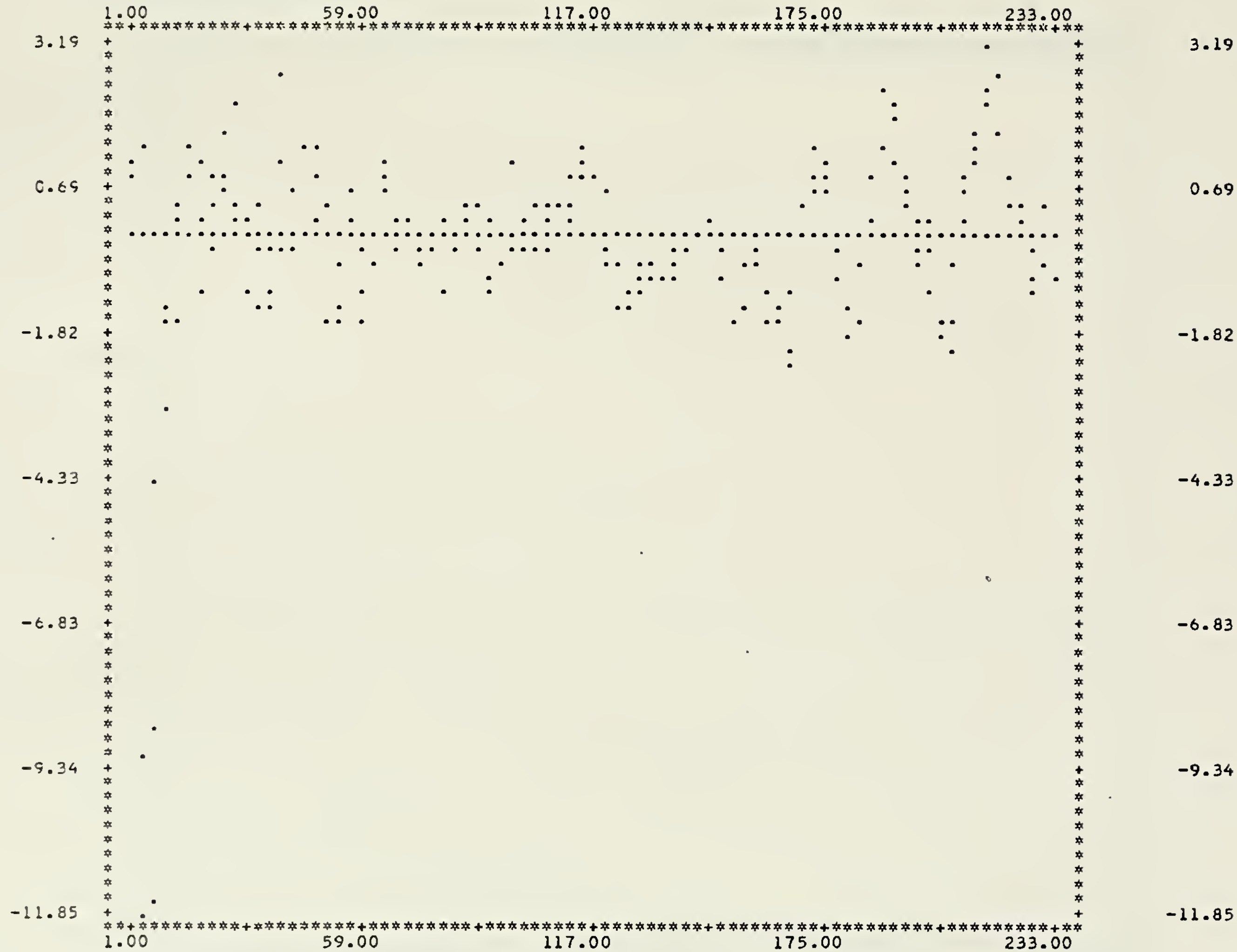
X-SCALE: "*"= 0.290E 01 UNITS
Y-SCALE: "*"= 0.253E 02 UNITS

KETRON MK-86 THESIS
MEAN ERROR IN X POSITION



X-SCALE: "*"= 0.290E 01 UNITS
Y-SCALE: "*"= 0.288E 00 UNITS

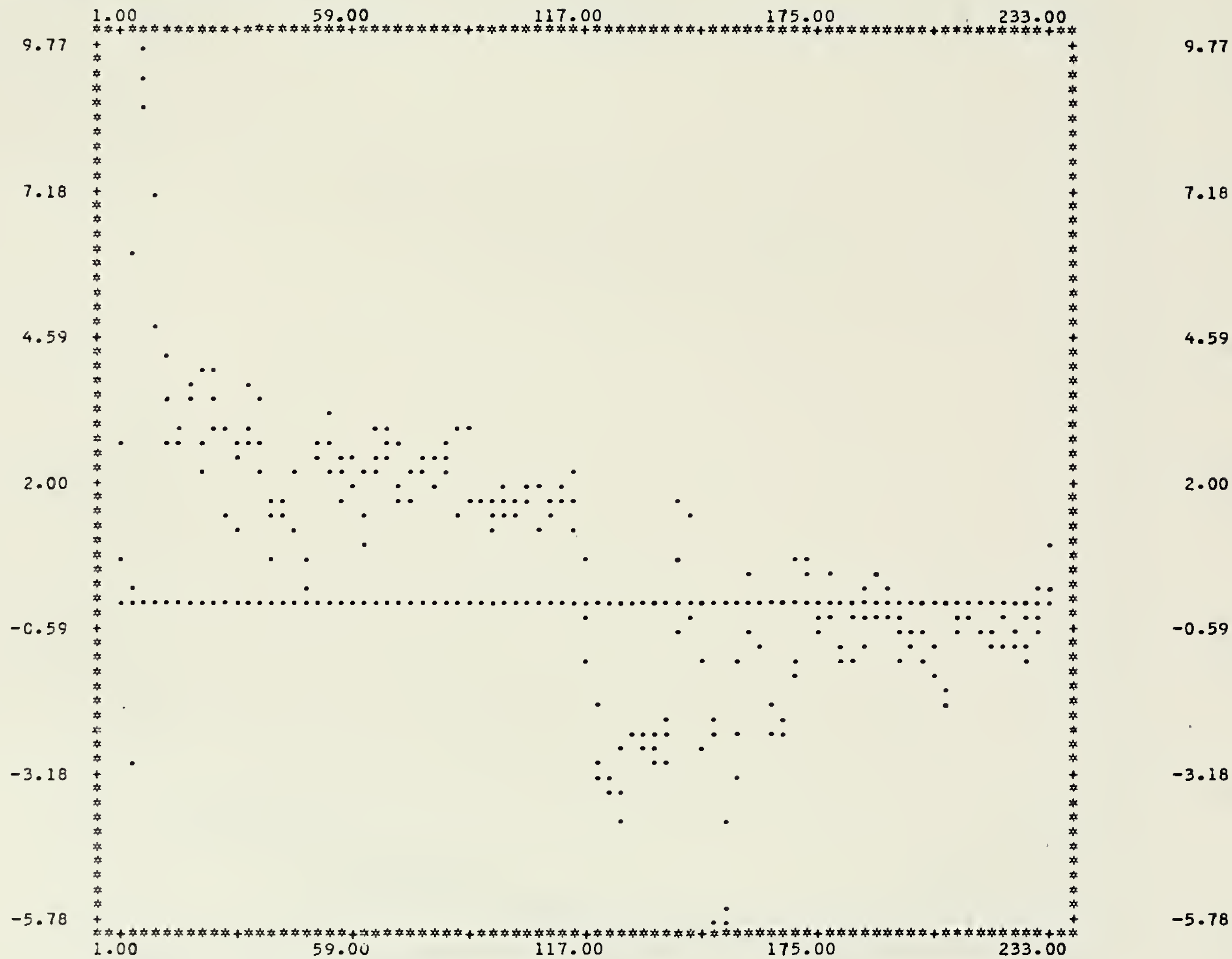
KETRON MK-86 THESIS
MEAN ERROR IN Y POSITION



X-SCALE: "*"= 0.290E 01 UNITS

Y-SCALE: "*"= 0.251E 00 UNITS

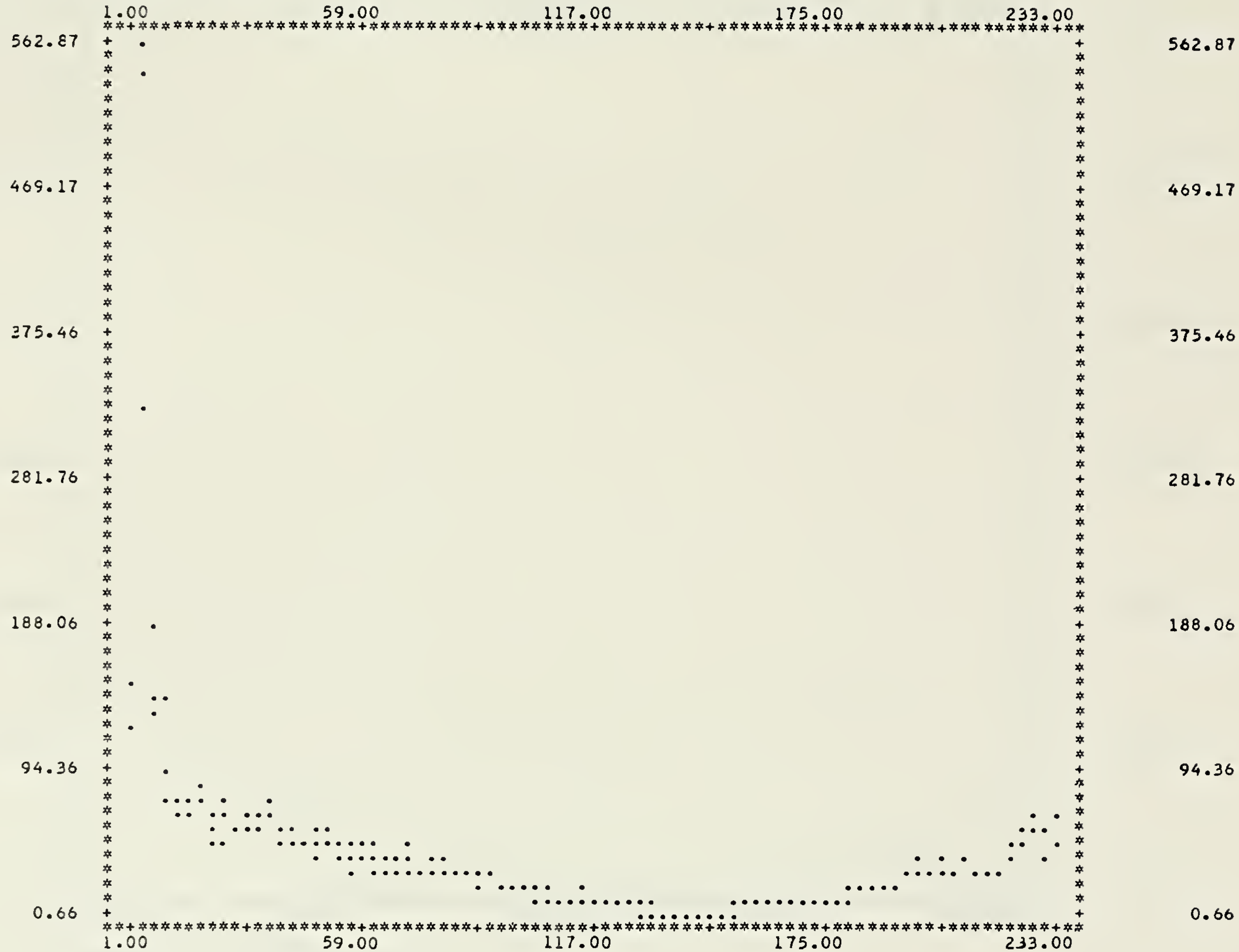
KETRON MK-86 THESIS
MEAN ERROR IN Z POSITION



X-SCALE: "*"= 0.290E 01 UNITS

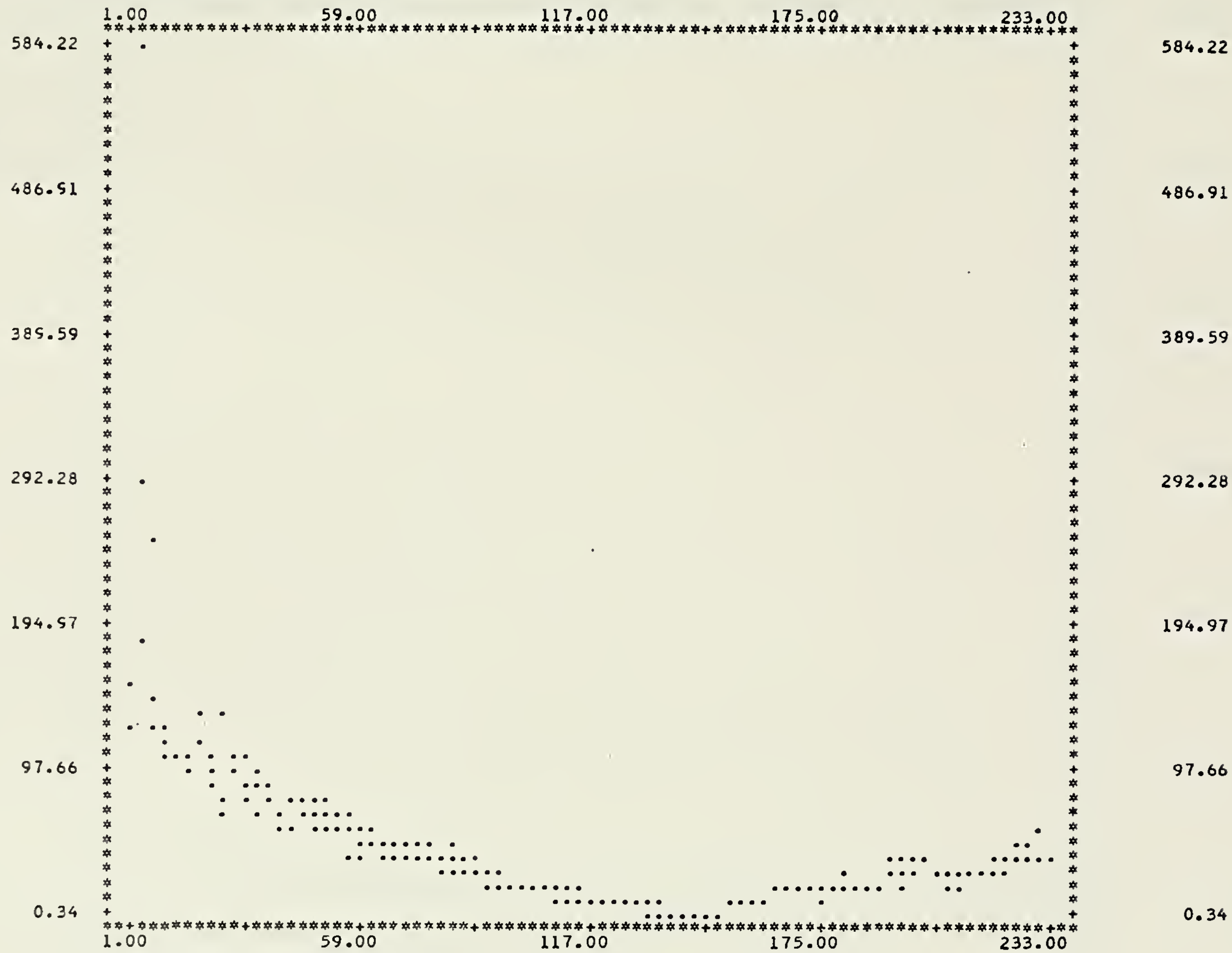
Y-SCALE: "*"= 0.259E 00 UNITS

KETRON MK-86 THESIS
X POSITION VARIANCE



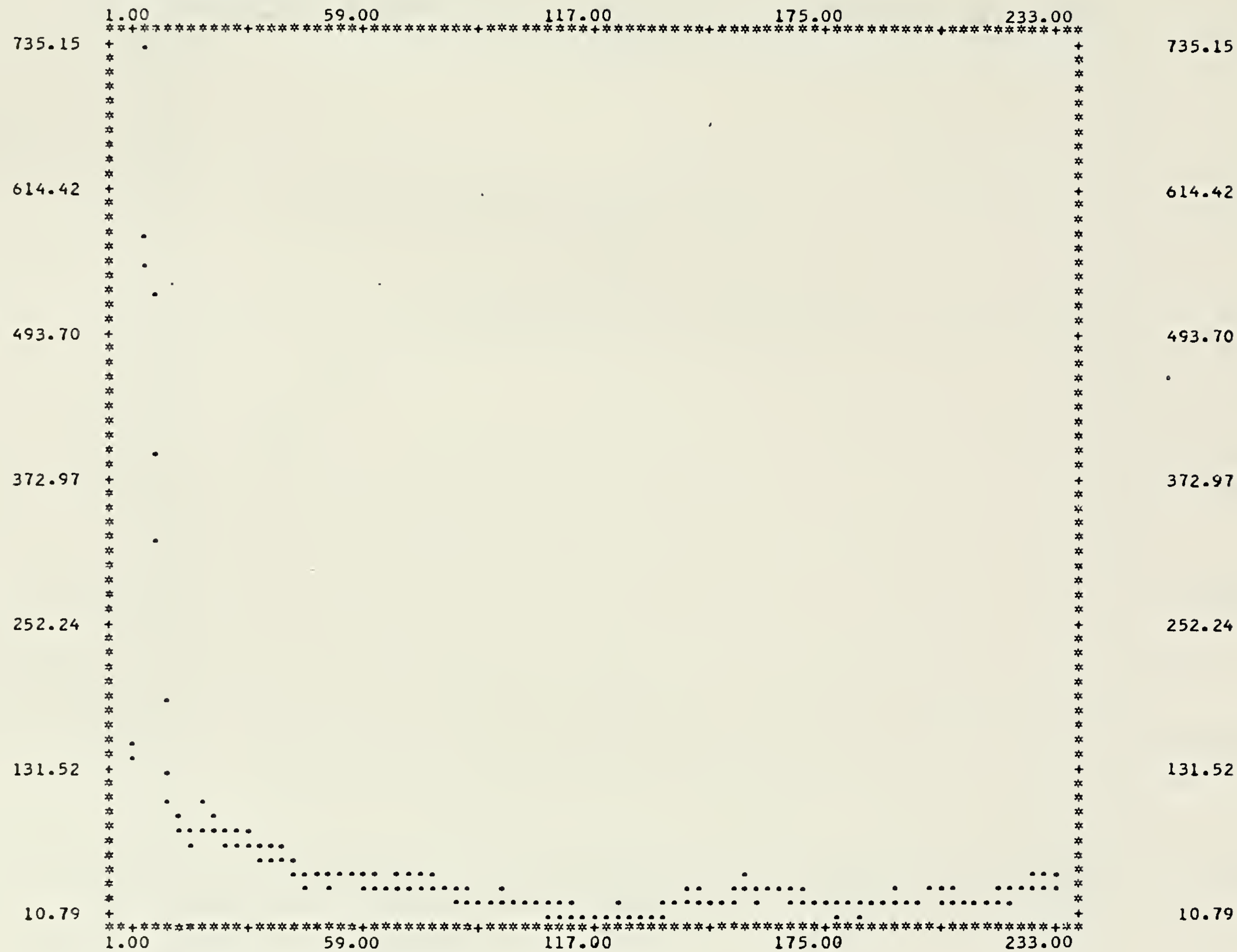
X-SCALE: "*"= 0.290E 01 UNITS
Y-SCALE: "*"= 0.937E 01 UNITS

KETRON MK-86 THESIS
Y POSITION VARIANCE



X-SCALE: "*"= 0.290E 01 UNITS
Y-SCALE: "*"= 0.973E 01 UNITS

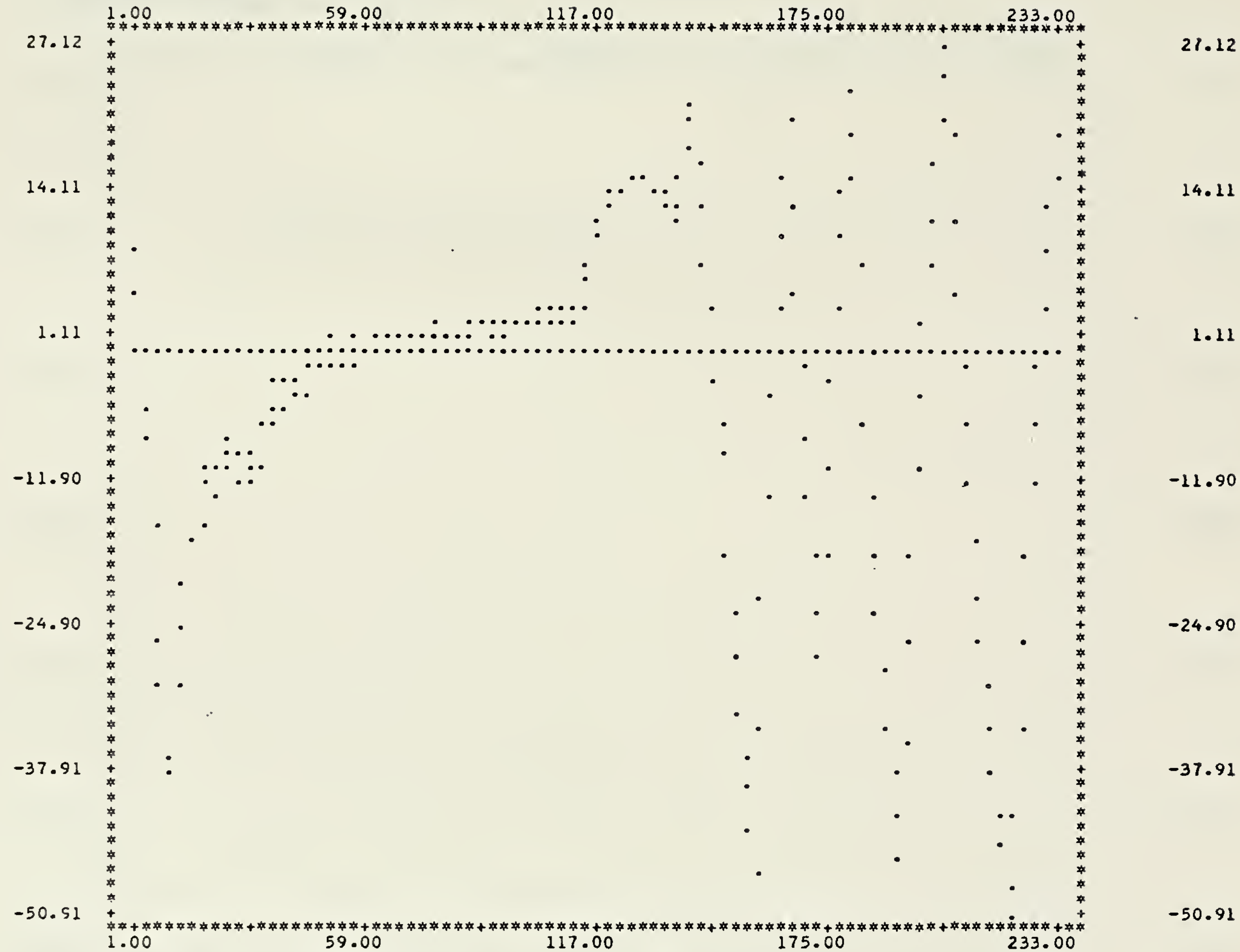
KETRON MK-86 THESIS
Z POSITION VARIANCE



X-SCALE: "*"= 0.290E 01 UNITS

Y-SCALE: "*"= 0.121E 02 UNITS

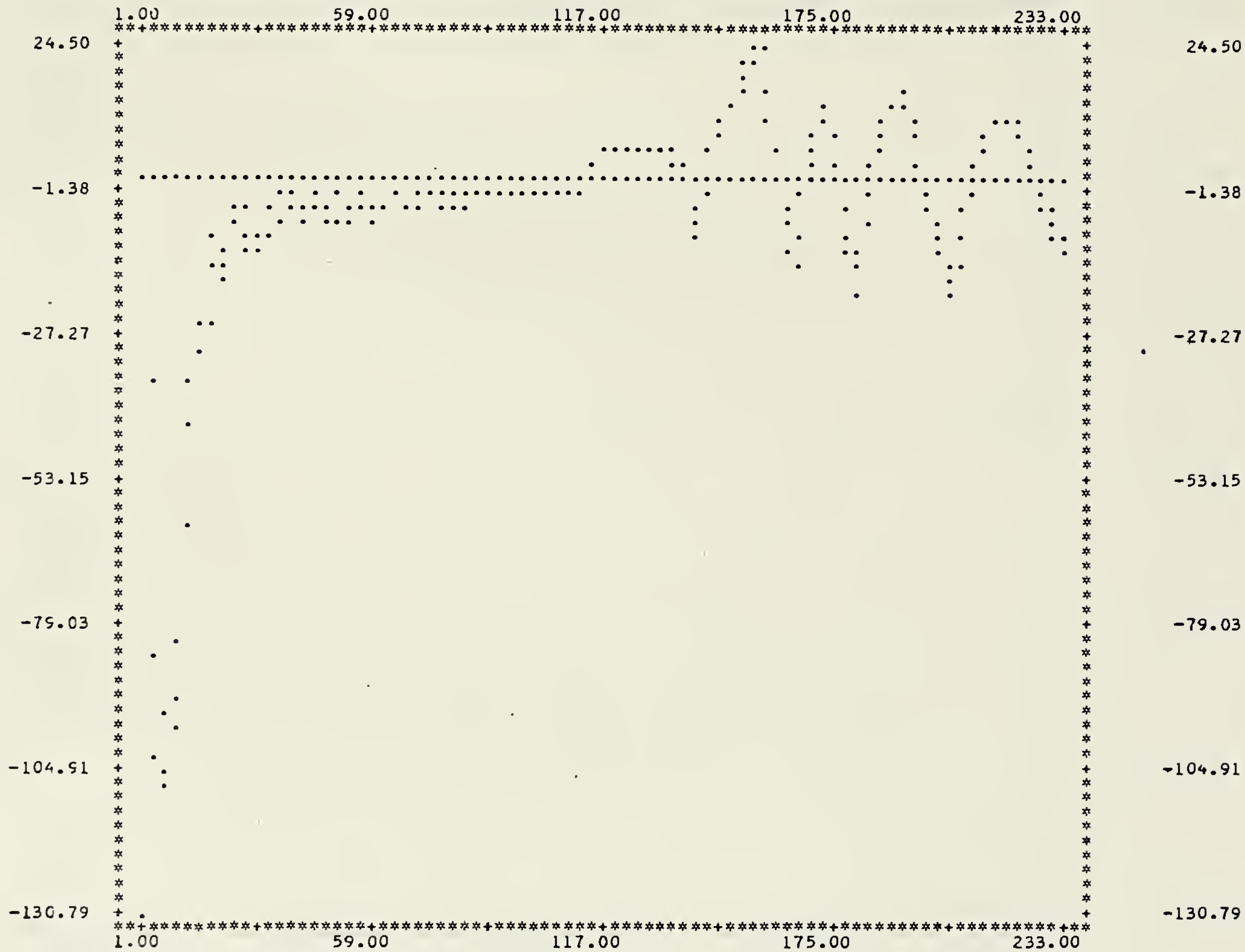
KETRON MK-86 THESIS
MEAN VELOCITY ERROR IN X DIRECTION



X-SCALE: "*"= 0.290E 01 UNITS

Y-SCALE: "*"= 0.130E 01 UNITS

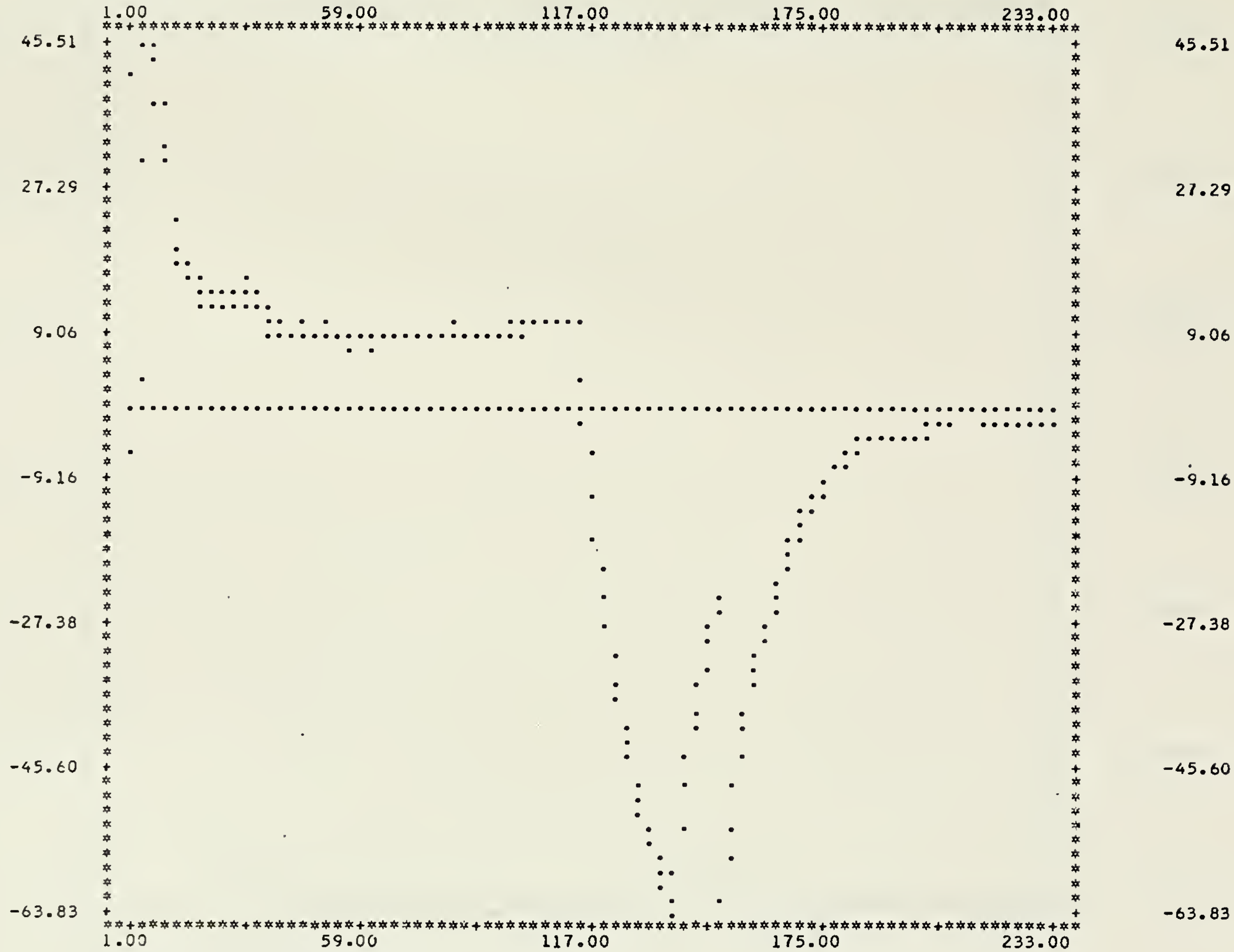
KETRON MK-86 THESIS
MEAN VELOCITY ERROR IN Y DIRECTION



X-SCALE: "*"= 0.290E 01 UNITS

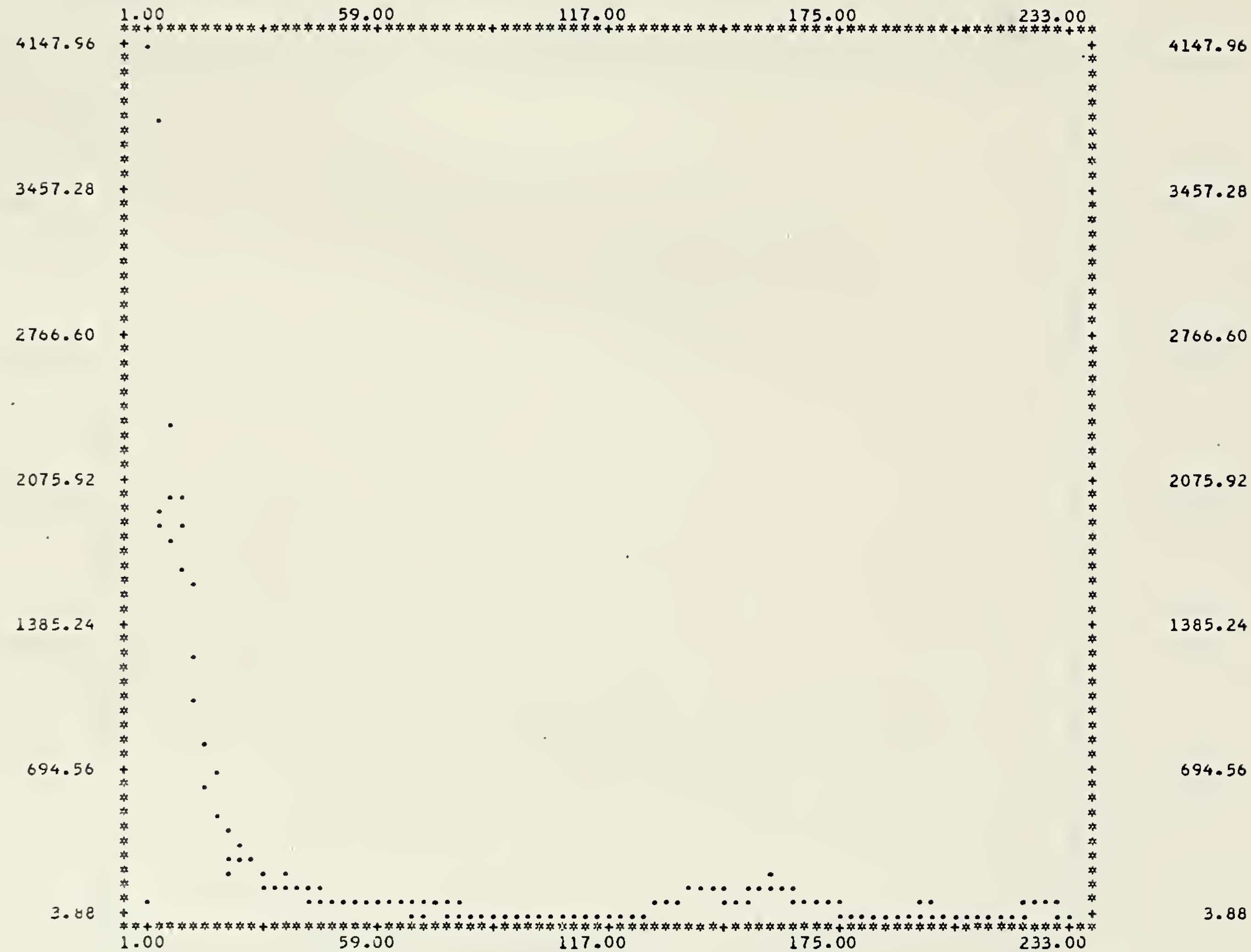
Y-SCALE: "*"= 0.259E 01 UNITS

KETRON MK-86 THESIS
MEAN VELOCITY ERROR IN Z DIRECTION



X-SCALE: "*"= 0.290E 01 UNITS
Y-SCALE: "*"= 0.182E 01 UNITS

KETRON MK-86 THESIS
X VELOCITY VARIANCE



X-SCALE: "*"= 0.290E 01 UNITS

Y-SCALE: "*"= 0.691E 02 UNITS

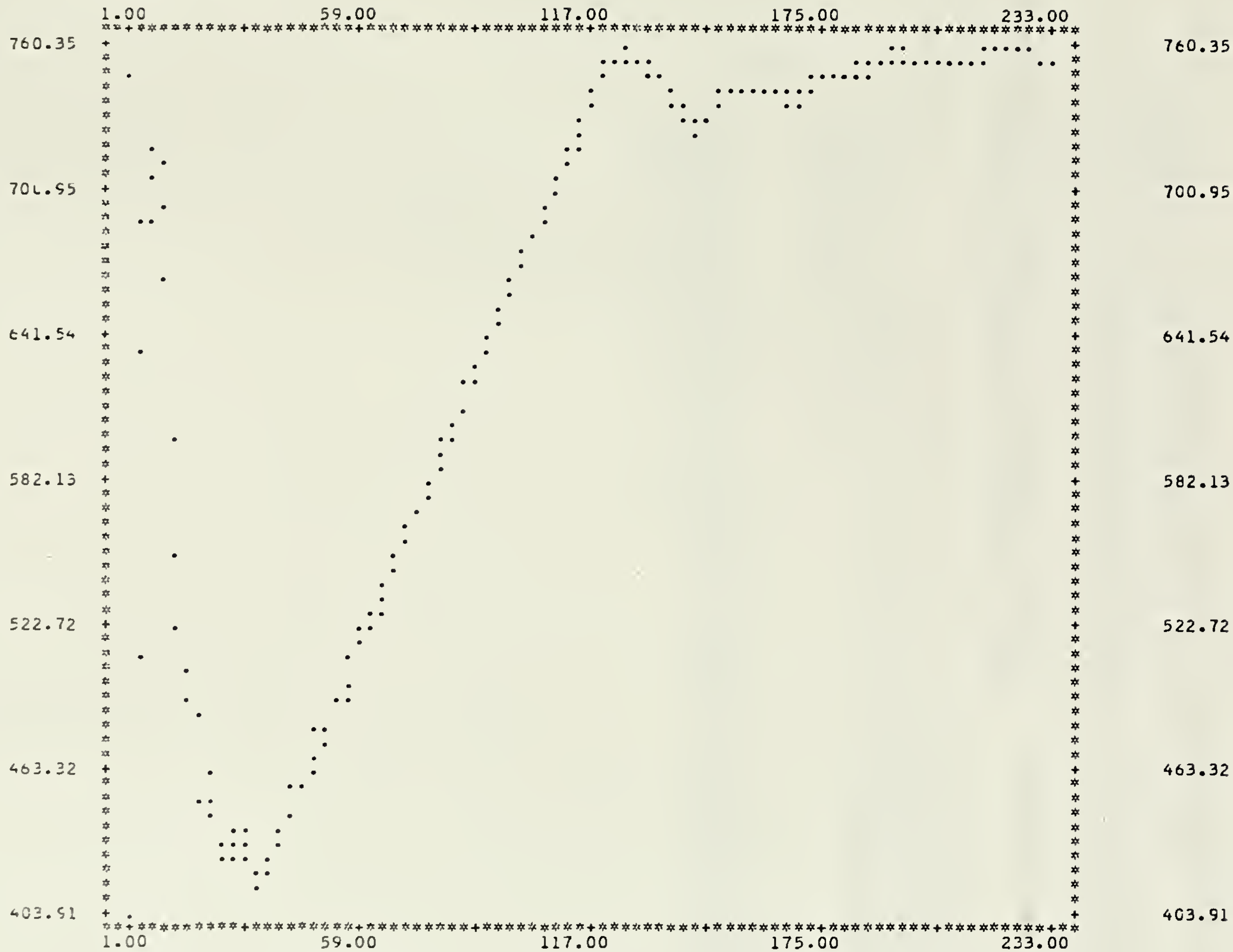
KETRON MK-86 THESIS
Y VELOCITY VARIANCE

X-SCALE: "*"= 0.290E 01 UNITS
Y-SCALE: "*"= 0.261E 03 UNITS

KETRON MK-86 THESIS
Z VELOCITY VARIANCE

X-SCALE: "*"= 0.290E 01 UNITS
Y-SCALE: "*"= 0.646E 02 UNITS

KETRON MK-86 THESIS
TARGET SPEED IN FT/SEC



X-SCALE: "*"= 0.290E 01 UNITS
Y-SCALE: "*"= 0.594E 01 UNITS

BIBLIOGRAPHY

1. Meditch, J. S., Stochastic Optimal Linear Estimation and Control, McGraw-Hill Book Co., Inc., 1969.

2. Korn, G. A., Random-Process Simulation and Measurements, McGraw-Hill Book Co., Inc., 1966.

3. Engineering Design Handbook, Section 1, Headquarters, U.S. Army Materiel Command, 1969.

4. Aldrich, G. T. and Krabill, W. B., "An Application of Kalman Techniques to Aircraft and Missile Radar Tracking," American Institute of Aeronautics and Astronautics Journal, v. 11, No. 7, p. 932-938, July 1973.

5. Will, T. J., A FORTRAN Simulation of the Target Data Filtering Section of the AA Tracking Loop in the MK-86 GFCS, Master's Thesis, Naval Postgraduate School, Monterey, 1973.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
3.	Department Chairman, Code 52 Department of Electrical Engineering Naval Postgraduate School Monterey, California 93940	2
4.	AssocProfessor D. E. Kirk, Code 52 Ki Department of Electrical Engineering Naval Postgraduate School Monterey, California 93940	2
5.	LTJG Michael Gordon Ketron NCS Box 33-B363 FPO New York, N. Y. 09540	1
6.	LT D. F. Regener, USN Naval Ship Missile System Engineering Station Space City 8 Port Hueneme, California 93043	2

152415
Thesis
K3953 Ketron
c.1 Monte-Carlo evaluation
of digital filters for
fire control systems.