MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963 A
DEVELOPMENT OF REAL-TIME ERROR ELLIPSES AS AN INDICATOR OF KALMAN FILTER PERFORMANCE

by

Joseph Jaros

March 1984

Thesis Advisor: A. Gerba, Jr.

Approved for public release, distribution unlimited
**Title:** Development of Real-Time Error Ellipses as an Indicator of Kalman Filter Performance  

**Type of Report & Period Covered:** Master's Thesis; March 1984  

**Authors:** Joseph Jaros  

**Performing Organization Name and Address:** Naval Postgraduate School, Monterey, California 93943  

**Controlling Office Name and Address:** Naval Postgraduate School, Monterey, California 93943  

**Distribution Statement:** Approved for public release, distribution unlimited  

**Security Classification:** UNCLASSIFIED  

**Abstract:** An error ellipse plotting routine was developed to provide real-time indication of Kalman filter performance. The study included an evaluation of the Hewlett-Packard HP-88 computer system's capability for providing real-time tracking information and an evaluation of the computer's possible use on the three-dimensional underwater tracking range at the Naval Underwater Weapons Engineering Station, Keyport, Washington. A series of tracking...
runs were used to demonstrate both linear and extended Kalman filtering. Information obtained from the error ellipses was used to modify filter parameters for improved filter performance. It was found that the error ellipse was useful as a tool for indicating filter performance and for making decisions regarding filter parameter modification. The HP-86 provided accurate, reliable results and it could be used for on-line graphics. However, the computing speed of the HP-86 computer as used in this study was too slow for on-line processing of the three-dimensional tracking problem.
ABSTRACT

An error ellipse plotting routine was developed to provide real-time indication of Kalman filter performance. The study included an evaluation of the Hewlett-Packard HP-86 computer system's capability for providing real-time tracking information and an evaluation of the computer's possible use on the three-dimensional underwater tracking range at the Naval Underwater Weapons Engineering Station, Keyport, Washington. A series of tracking runs were used to demonstrate both linear and extended Kalman filtering. Information obtained from the error ellipses was used to modify filter parameters for improved filter performance. It was found that the error ellipse was useful as a tool for indicating filter performance and for making decisions regarding filter parameter modification. The HP-86 provided accurate, reliable results and it could be used for on-line graphics. However, the computing speed of the HP-86 computer as used in this study was too slow for on-line processing of the three-dimensional tracking problem.
TABLE OF CONTENTS

I. INTRODUCTION------------------------------------- 7

II. KALMAN FILTER THEORY---------------------------------- 9
   A. LINEAR MATHEMATICAL MODEL-------------------------- 9
      1. The Plant-------------------------------------- 9
      2. Noise Processes----------------------------- 10
      3. Initial State Description---------------------- 11
   B. DISCRETE TIME ESTIMATION----------------------------- 12
      1. The Estimator Equations--------------------- 12
      2. Gain and Covariance Equations------------------ 14
   C. NONLINEAR ESTIMATION - EXTENDED KALMAN FILTER------------------- 17
      1. Nonlinear Model----------------------------- 17
      2. Extended Kalman Filter Equations------------- 17

III. ERROR ELLIPSOIDS--------------------------------------- 20
   A. THEORY---------------------------------------- 20
   B. ERROR ELLIPSOIDS AND FILTER DIVERGENCE---------- 25

IV. PROBLEM DEFINITION-------------------------------------- 27
   A. PROBLEM DESIGN---------------------------------- 27
      1. Linear Tracking----------------------------- 27
      2. Nonlinear Tracking-------------------------- 32
   B. COMPUTER SIMULATION------------------------------- 34
      1. Computer Hardware/Software------------------- 34
      2. Track Generation----------------------------- 35
      3. Noise Generation----------------------------- 35
      4. Gating Scheme-------------------------------- 36
5. Collection of Statistics

V. TARGET TRACKING AND ERROR ELLIPSE ANALYSIS
   A. LINEAR TRACKING
   B. NONLINEAR TRACKING

VI. CONCLUSIONS AND RECOMMENDATIONS
   A. ERROR ELLIPSE
   B. COMPUTER PERFORMANCE

APPENDIX A - TRACK GENERATION
APPENDIX B - COMPUTER PROGRAM EXPLANATION
APPENDIX C - PROGRAM LISTING
LIST OF REFERENCES
BIBLIOGRAPHY
INITIAL DISTRIBUTION LIST
I. INTRODUCTION

The Kalman filter's importance as an estimator and predictor is well documented. Providing real-time information concerning filter performance so that on-line adjustments to filter parameters can be made continues to be an area of high interest. This study investigates the usefulness of the error ellipse as a tool for providing a real-time indication of filter performance.

Part of the investigation involves an evaluation of the Hewlett-Packard HP-86 computer system's capacity to operate in a real time tracking environment, and its capabilities for providing information concerning filter performance. The rationale behind this investigation is based on the requirement for the Naval Underwater Weapons Engineering Station, Keyport, Washington, to accurately acoustically track torpedoes on a three-dimensional underwater tracking range. A knowledge of the range operation is not within the scope of this study. It is sufficient to know that presently the range receives four time measurements every 1.31 seconds, and these measurements are nonlinear functions of the torpedo position.

To gain a better understanding of the error ellipse, a 4-state tracking scenario was chosen for this study.
Initially, the linear tracking problem is discussed, followed by an investigation of the nonlinear problem. Of primary interest are on-line methods to improve filter performance using information provided by the error ellipse for filter parameter modification.
II. KALMAN FILTER THEORY

A. LINEAR MATHEMATICAL MODEL

1. The Plant

For this model the state and measurement equations for the plant are linear. Hence the discrete form is used. The assumed plant model is described by a linear, vector difference equation:

\[ x(k+1) = Ax(k) + Au(k) + Aw(k) \]  (State Equation)  (2-1)

and a linear, vector measurement equation:

\[ z(k) = Ox(k) + v(k) \]  (2-2)

where:

- \( x(k) \) is an n-dimensional column vector, denoting the state of the plant at "time" \( k \).
- \( u(k) \) is the deterministic control input, an m-vector, at time \( k \).
- \( w(k) \) is a p-dimensional vector representing any random forcing inputs at time \( k \).
- \( z(k) \) is a q-dimensional vector representing measurements made at time \( k \).
- \( v(k) \) is a q-dimensional vector representing random measurement made at time \( k \).

\( A, C, R, \) and \( Q \) are assumed constant coefficient matrices of dimension \( nxn, nxm, nxp, \) and \( qxn \) respectively.
2. Noise Processes

In order to place probabilistic structure on the noise processes \( v(k) \) and \( w(k) \) the following assumptions are made:

(a) \( v(k) \) and \( w(k) \) are individually white processes, that is, for any \( k \) and \( l \), with \( k \neq l \), \( v(k) \) and \( v(l) \) are independent random variables, and \( w(k) \) and \( w(l) \) are independent random variables.

(b) \( v(k) \) and \( w(k) \) are individually zero mean, Gaussian processes with known covariances.

(c) \( v(k) \) and \( w(k) \) are independent processes.

Thus for the measurement noise:

**Mean:** \( \mathbb{E}[v(k)] = 0 \) (\( k=0,1,2,3,\ldots \)) (2-3)

**Covariance:**

\[
\mathbb{E}[v(k)v^T(l)] = \mathbb{E}[v(k)]\mathbb{E}[v^T(l)]
\]

\[
= \begin{cases} 
0 & k \neq l \\
R_k & k = l
\end{cases}
\]

or \( \mathbb{E}[v(k)v^T(l)] = R_k \delta_{kl} \) \( (k,l=0,1,2,\ldots) \) (2-4)

where \( \delta_{kl} \) is the Kronecker delta function defined as:

\[
\delta_{kl} = \begin{cases} 
0, & k \neq l \\
1, & k = l
\end{cases}
\]
Likewise for the random forcing input:

\[ E[w(k)] = 0 \quad (k=0,1,2,3,...) \quad (2-5) \]

\[ E[w(k)w^T(1)] = \Omega_k \delta_{kl} \quad k,l=0,1,2,3,... \quad (2-6) \]

\( \Omega_k \) and \( R_k \) are nonnegative definite symmetric for all \( k \). Also since \( y(k) \) and \( \hat{y}(k) \) are zero mean and independent then:

\[ E[y(k)\hat{y}^T(1)] = 0 \quad (2-7) \]

For the purposes of this study, unless otherwise specified \( \Omega_k \) and \( R_k \) are considered to be known and constant, although both may be time varying.

3. Initial State Description

For the initial state of the difference equation (2-1) it is unlikely that \( x_0 \) will be available. Hence, it is assumed that \( x_0 \) is a Gaussian random variable of known mean \( \overline{x}_0 \) and known covariance \( P_0 \), i.e.,

\[ E[x(0)] = \overline{x}_0 \]

\[ E\left\{[(x_0 - \overline{x}_0)[(x_0 - \overline{x}_0)^T] = P_0 \]
This choice for the initial state has the advantage of causing the subsequent estimation scheme to be unbiased for all \( t \). [Ref. 1] Further it is assumed that the initial state and the measurement noise are uncorrelated:

\[
E[x(0)v^T(k)] = E[y(k)x^T(0)] = 0 \quad (k=0,1,2,3,...)
\]

Also the initial state and the random forcing input are uncorrelated:

\[
E[x(0)w^T(k)] = E[w(k)x^T(0)] = 0 \quad (k=0,1,2,3,...)
\]

B. DISCRETE-TIME ESTIMATION

1. The Estimator Equations

The estimation problem involves generating an optimal estimate for \( x(j) \) for the system described by the difference equation (2-1) from the noisy measurements \( z(0), z(1), ..., z(j) \). This estimate will be denoted by \( \hat{x}(j/j) \), which means the estimate of \( x \) at time \( j \) given measurements at times up to and including time \( j \). The estimate must be optimal in the sense that the expected value of the sum of the squares of the error in the estimate is a minimum, i.e.:

\[
E\left\{ (\hat{x}(k/k) - x(k))(\hat{x}(k/k) - x(k))^T \right\} = \text{minimum}
\]
The estimator is characterized by the linear relationship:

\[ \hat{x}(k/k) = \hat{x}(k/k-1) + G(k)[z(k) - \hat{c}x(k/k-1)] \quad (k=0,1,2,\ldots) \]  

(2-8)

where

- \( \hat{x}(k/k) \) is the optimal (minimum variance) estimate of \( x(k) \) given observations at times up to and including \( k \).
- \( \hat{x}(k/k-1) \) is the optimal one-step prediction of \( x(k) \) given observations at times up to and including \( k-1 \).
- \( G(k) \) is the optimal estimation gain matrix which will minimize the variance of estimation error.

For the initial estimate \( \hat{x}(0/0) \), the estimator equation (2-8) is initialized with \( \hat{x}(0/-1) \), which is not a random variable. If \( \hat{x}(0/-1) \) is selected such that:

\[ \hat{x}(0/-1) = E[x(0)] = \bar{x}_0 \]

it can be shown that this choice of \( \hat{x}(0/-1) \) makes the estimator unbiased for all \( k \). [Ref. 1] The estimator's best available information concerning \( x(k-1) \) is the estimate \( \hat{x}(k-1/k-1) \), therefore it is reasonable to assume that

\[ \hat{x}(k/k-1) = \phi \hat{x}(k-1/k-1) + \Delta u(k-1) \]  

(2-9)

is the best prediction.
In summary, equations (2-8) and (2-9) are the estimator equations, with \( \hat{x}(0/-1) = \bar{x}_0 \) as the initial condition.

2. Gain and Covariance Equations

Without going into detailed derivations, the optimal estimator gains, \( G(k) \), used in the estimator equation (2-8), are those which satisfy:

\[
G(k) = P(k/k-1)C^T [CP(k/k-1)C^T + R]^{-1} \tag{2-10}
\]

\[
P(k/k) = [I - G(k)C]P(k/k-1) \tag{2-11}
\]

\[
P(k+1/k) = \phi P(k/k) \phi^T + Q \tag{2-12}
\]

with the initial conditions:

\[
P(0/-1) = P_0 = E\{[\hat{x}(0) - \bar{x}_0][\hat{x}(0) - \bar{x}_0]^T\}
\]

where

\[
P(k/k) = E\{[\hat{x}(k/k) - \hat{x}(k)][\hat{x}(k/k) - \hat{x}(k)]^T\}
\]

is the covariance of estimation error matrix.
\[ P(k/k-1) = E\{[\hat{x}(k/k-1) - x(k)][\hat{x}(k/k-1) - x(k)]^T\} \]

is the covariance of one-step prediction error matrix.

\[ Q = E[\hat{r}(k) \cdot \hat{w}(k) \cdot \hat{w}^T(k) \cdot \hat{r}^T(k)] \]

is the state excitation matrix.

\( P(k/k) \) and \( P(k/k-1) \) are symmetric, positive definite matrices.

Several observations can be made concerning the linear Kalman gain (2-10), covariance (2-11, 2-12) and estimator (2-8) equations.

(a) The estimator gains, \( G(k) \), do not depend on the measurement data and hence can be precomputed, stored, and used as the processing measurements become available.

(b) Although not obvious from the equation, the time-varying gain, \( G(k) \), depends in time as:

\[ G(k) = \frac{1}{(k + 1)} \quad [\text{Ref. 2}] \quad (2-13) \]

Thus the effect is to weight the correction term, \( [z(k) - \hat{x}(k/k-1)] \), in the estimator equation (2-8) less heavily as time progresses. The advantage of a greater initial weight allows for possibly large differences between \( z(k) \) and \( \hat{x}(k/k-1) \) during the initial observations, and a large gain will result in a significant change in the
next estimate. This advantage is also borne out in that there is less confidence in the quality of the estimates during the early observations compared with the quality after numerous observations. Hence the later an observation, the less drastic an estimate will be altered or affected by an isolated observation discrepancy.

(c) In general, the variance of estimation error decreases in a manner analogous to the gain schedule (2-13), i.e., it decreases as k grown larger, reflecting greater confidence in the estimate as the number of observations increases. Selection of the proper initial condition, \( P_0 \), is important when studying the effect of measurement errors on the behavior of the estimate. So \( P_0 \) should be assigned pessimistic values which would correspond to a lack of information about the initial state. In cases when the initial state is completely unknown, then \( P_0 \rightarrow I \). [Ref. 3]

(d) The \( Q \) matrix serves to compensate for model errors and prevents the covariance matrix from becoming too small or optimistic. A small covariance matrix would result in a small filter gain, and subsequent observations are essentially ignored, which could result in filter divergence. The \( Q \) matrix prevents \( G(k) \) from approaching zero by adding uncertainty to the system which is reflected in a degradation of certainty (increase in \( P(k+1/k) \)).
C. NONLINEAR ESTIMATION - EXTENDED KALMAN FILTER

In many practical applications, the state equations and/or measurement equations are nonlinear. Before the Kalman filter equations can be used, the problem must be linearized and the Kalman filter equations are applied with some modification.

1. Nonlinear Model

Consider a nonlinear discrete system of state and observation equations given by:

\begin{align}
    x(k+1) &= f(x(k), u(k), k) + w(k) \tag{2-14} \\
    z(k) &= h(x(k), k) + v(k) \tag{2-15}
\end{align}

In these equations \( f \) and \( h \) are nonlinear functions of the state variable \( x \), \( w(k) \) is the plant excitation noise, and \( v(k) \) is the measurement noise. The plant noise and measurement noise are assumed to be uncorrelated, zero-mean, and white. The same equations (2-3 thru 2-7) apply as for the linear model.

2. Extended Kalman Filter Equations

In order to apply the linear filter equations, equations (2-14) and (2-15) are expanded about the best estimate of the state at that time and only the first-order terms are kept.
That is, defining $A(k)$ as:

$$A(k) = \frac{\partial f}{\partial x} \mid (\hat{x}(k/k), \hat{u}(k), \hat{k})$$

and

$$H(k) = \frac{\partial h}{\partial x} \mid (\hat{x}(k/k-1))$$

As can be seen from the above equations, the filter estimates, $\hat{x}(k/k)$ and $\hat{x}(k/k-1)$ are used as the "best" estimates about which the linearization is performed. The matrices $A(k)$ and $H(k)$ must be used to generate $G(k)$ so it is available to process $z(k)$ when it is obtained. The modified extended Kalman filter equations are then:

Gain Equation:

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

(2-16)

Filter Update Equation:

$$\hat{x}(k/k) = \hat{x}(k/k-1) + G(k)[z(k) - h(\hat{x}(k/k-1))]$$

(2-17)
Prediction Equation:

\[ \hat{x}(k+1|k) = \hat{f}(\hat{x}(k|k), u(k), k) \]  
(2-18)

Covariance of Estimation Error Equations:

\[ P(k|k-1) = A(k-1)P(k-1|k-1)A^T(k-1) + Q(k-1) \]  
(2-19)

\[ P(k|k) = [I - G(k)H(k)]P(k|k-1) \]  
(2-20)

For the initial estimate \( \hat{x}(0/0) \), equation (2-17) is initialized with \( \hat{x}(0/-1) \) with

\[ \hat{x}(0/-1) = E[x(0)] = \bar{x}_0 \]

\( \hat{x}(0/-1) \) is also used to initially evaluate \( H(k) \).

As in the linear case:

\[ P(0/-1) = P_0 = E[(x_0 - \bar{x}_0)(x_0 - \bar{x}_0)^T] \]
III. ERROR ELLIPSOIDS

A. THEORY

Since the estimate \( \hat{x}(k/k) \) is unbiased in the Kalman filter equations, the \( P(k/k) \) matrix represents the covariance of the error in the estimate. If the estimate were biased, \( P(k/k) \) would represent the second-moment matrix rather than the covariance matrix. Hence, \( P(k/k) \) provides significant information about the accuracy of the estimate. If the physical model is accurately described by the state and measurement equations (2-1, 2-2), then \( P(k/k) \) can be used to describe the manner in which the estimate converges (or diverges) to the true state. Examination of the \( P(k/k) \) matrix directly, element by element, is not a realistic approach, since the matrix contains \( n^2 \) elements, where \( n \) is the number of state variables. To simplify the situation the concept of the error ellipsoid is used. [Ref. 4]

As discussed earlier, the assumptions are made that the initial state of the plant \( x_0 \) is Gaussian, as are the random processes \( v(k) \) and \( w(k) \). Using these assumptions it follows that \( x(k) \) and \( \hat{x}(k/k) \) are also Gaussian since they are linear combinations of Gaussian variables and deterministic quantities. Using the same rationale, the estimation error, defined as:
\[ e(k/k) \triangleq \hat{x}(k/k) - x(k) \]

is also Gaussian. Using the fact that the mean of the estimation error is 0, the probability density function for \( e(k/k) \) is:

\[
p_e[e(k/k)] = \frac{1}{(2\pi)^{n/2}|P(k/k)|^{1/2}} \exp[-\frac{1}{2}e^T(k/k)P^{-1}(k/k)e(k/k)]
\]

(3-1)

The density function, \( p_e[e(k/k)] \) will have a constant value whenever the exponent has a constant value. That is:

\[
-\frac{1}{2}e^T(k/k)P^{-1}(k/k)e(k/k) = c
\]

or

\[
e^T(k/k)P^{-1}(k/k)e(k/k) = c^2
\]

(3-2)

where \( c \) is an arbitrary constant.

As demonstrated by Sorenson [Ref. 1] and Kirk [Ref. 2], it can be shown that the locus of points \( e(k/k) \) which satisfy equation (3-2) are hyperellipsoids. For the two-dimensional case which is of concern, equation (3-2) describes an ellipse. This can be seen by fixing time and rewriting (3-2) as:
\[ e^T w e = c^2 \]  \hspace{1cm} (3-3)

where
\[ w = P^{-1} \left( \begin{array}{c} k_1 \\ k_2 \end{array} \right) \] (a 2 x 2 symmetric matrix)

Expanding the left side of (3-3) gives:
\[ w_1 e_1^2 + w_1 w_2 e_1 e_2 + w_2 e_2^2 = c^2 \]

which because of symmetry gives:
\[ w_1 e_1^2 + 2w_1 w_2 e_1 e_2 + w_2 e_2^2 = c \]  \hspace{1cm} (3-4)

Since \( w_{11} > 0 \), \( w_{22} > 0 \), and \( w_{11} w_{22} > w_{12}^2 \), equation (3-4) describes an ellipse, in which the principal axes do not coincide with the coordinate axes. The ellipse is rewritten in terms of \( y_1 \) and \( y_2 \) as the coordinate axis, and it can be shown that \( y_1 \) and \( y_2 \) are the eigenvectors of \( w \) with \( \lambda_1 \) and \( \lambda_2 \) defined as the corresponding eigenvalues. [Ref. 2]

(See Figure 3-1). The equation for the ellipse can be rewritten in terms of a coordinate system having unit vectors in the directions of \( y_1 \) and \( y_2 \) as the basis vectors.

The ellipse equation becomes:
\[ \lambda_1 y_1^2 + \lambda_2 y_2^2 = c^2 \]  \hspace{1cm} (3-5)
Remembering that $\mathbf{w} = \mathbf{P}^{-1}(\mathbf{k}/\mathbf{k})$, it can be shown that the corresponding eigenvectors and eigenvalues for $\mathbf{w}^{-1} = \mathbf{P}(\mathbf{k}/\mathbf{k})$ are $\mathbf{y}(1), \mathbf{y}(2), a_1$, and $a_2$, where $a_1 = \frac{1}{\lambda_1}$ and $a_2 = \frac{1}{\lambda_2}$.

Equation (3-5) can be rewritten:

$$\frac{\mathbf{y}_2^2(1)}{a_1} + \frac{\mathbf{y}_2^2(2)}{a_2} = c^2 \quad (3-6)$$
For the purposes of this study, the term error ellipsoid refers to the specific case when $c = 1$. So equation (7-9) becomes:

$$\frac{y^2(1)}{a_1} + \frac{y^2(2)}{a_2} = 1$$

In terms of the Cartesian coordinates $x', y'$, which use $e_1$ and $e_2$ as basis vectors:

$$x' = x\cos\theta + y\sin\theta$$
$$y' = -x\sin\theta + y\cos\theta$$

where $\theta$ is the angle of rotation of the axes and can be computed from:

$$\theta = \frac{1}{2} \tan^{-1} \left( \frac{2\text{cov}(e_x e_y)}{\text{var}(e_x) - \text{var}(e_y)} \right)$$

[Ref. 5]

For a given value of $c$ it is possible to integrate the probability density over the surface of the error ellipse to obtain the probability that a particular sample point will lie within the ellipsoid. For this study, $n = 2$, $c = 1$, and the probability the error is inside the ellipse is 0.39 u.

In summary, the error ellipsoid can be used to characterize the concentration of the estimate about the
true value of the state. A decrease in the magnitude of an axis of the ellipse is an indication that the error in the estimate is decreasing in that direction.

One important item that needs to be pointed out is that often the components of the state vector represent entirely different types of variables, for example the components might represent range, velocity, and depth. Since the two-dimensional ellipses are determined by using two components of the state vector, it is reasonable to examine submatrices relating state variables of the same character. Doing so will preclude most scaling difficulties when plotting the ellipses, and provide more meaningful insight into the results.

3. ERROR ELLIPSOIDS AND FILTER DIVERGENCE

Thus far the discussion has centered around using submatrices of the $P(k/k)$ covariance of estimation error matrix as the input for the error ellipse to indicate filter performance. As proposed by Heffes [Ref. 6] and Nishimura [Ref. 7], $P(k/k)$ can be considered as a "design" covariance matrix $P_d$, using the assumption that the only errors are in $Q_k$, $R_k$, and $P_0$ with the following inequalities holding for all $k$:

$$
Q_d \geq Q_k, \quad R_d \geq R_k, \quad P_d \geq P_0
$$

(3-7)
with the subscript "d" indicating designed and "a" indicating actual. Equation (3-7) implies more input noise, more measurement noise, and more initial state uncertainty in the design than actually exists. This conservative filter design results in a somewhat pessimistic design error covariance $P_d(k/k)$. The actual error covariance $P_a(k/k)$ resulting from using a filter designed with $Q_d$, $R_d$, and $P_o$ is related to the design error covariance in the following manner:

$$P_d(k/k) > P_a(k/k)$$

This result is particularly useful when one simply does not know accurately the noise covariance of the input or output, but an upper bound is known. Designing assuming the noise covariance is at its upper bound will result in $P_a(k/k)$ being upper bounded by $P_d(k/k)$. In some sense a worst case design results. Filter divergence exists when the design error covariance $P_d(k/k)$ remains bounded while the error performance matrix $P_a(k/k)$ becomes very large relative to $P_d(k/k)$ or is, in fact, unbounded.
IV. PROBLEM DEFINITION

A. PROBLEM DESIGN

The purpose of the tracking problem is to study the use of error ellipsoids as real-time indicators of filter performance. In order to keep the design model realistic albeit reasonably simplified for ease of study, a two-dimensional tracking problem using several different tracks has been selected.

1. Linear Tracking

All tracks are based on an x-y coordinate system with the target moving in the x or y direction relative to the sensor located at the origin. Thus for aircraft tracking, altitude is considered constant, as is depth for torpedo tracking.

Defining the state variables as:

\[ x_1 = x \] x-coordinate of the target location.
\[ x_2 = \dot{x} \] velocity of target \( (v_x) \) in x-direction.
\[ x_3 = y \] y-coordinate of the target location.
\[ x_4 = \dot{y} \] velocity of target \( (v_y) \) in y-direction.
resulting in a state vector:

\[
X = \begin{bmatrix}
    x \\ 
    \dot{x} \\ 
    y \\ 
    \dot{y}
\end{bmatrix}
\]  

The following are the state equations:

\[
\begin{align*}
\dot{x}_1(t) &= x_2(t) \\
\dot{x}_2(t) &= w_1(t) \\
\dot{x}_3(t) &= x_4(t) \\
\dot{x}_4(t) &= w_2(t)
\end{align*}
\]  

where \(w_1(t)\) and \(w_2(t)\) are assumed to be uncorrelated, random processes that account for unknown target accelerations and nonlinear target motions. Writing the discrete form of the state equations gives:
\[
x_1(k+1) = x_1(k) + T \cdot x_2(k) + \frac{\sigma_1^2}{2} \cdot \omega_1(k)
\]

\[
x_2(k+1) = x_2(k) + T \cdot \omega_1(k)
\]  \hspace{1cm} (4-3)

\[
x_3(k+1) = x_3(k) + T \cdot x_4(k) + \frac{\sigma_2^2}{2} \cdot \omega_2(k)
\]

\[
x_4(k+1) = x_4(k) + T \cdot \omega_2(k)
\]

or

\[
x(k+1) = \Phi x(k) + \Gamma \omega(k)
\]  \hspace{1cm} (4-4)

With T, the sampling period equal to 1 second, the matrices \( \Phi \) and \( \Gamma \) are:

\[
\Phi = \begin{bmatrix}
1 & 1 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 1 \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

\[
\Gamma = \begin{bmatrix}
.5 & 0 \\
0 & 1 \\
0 & .5 \\
0 & 1
\end{bmatrix}
\]  \hspace{1cm} (4-5)

It is assumed that the sensor gives noisy, but uncorrelated measurements of x and y. Hence the discrete measurement equations are:
\[ z_1(k) = x_1(k) + v_1(k) \]  \hfill (4-6)
\[ z_2(k) = x_3(k) + v_2(k) \]

with \( v_1(k) \) and \( v_2(k) \) uncorrelated random noise.

Thus for the measurement equation:
\[ z(k) = cx(k) + v(k) \]  \hfill (4-7)

the matrix \( c \) is:

\[
\begin{bmatrix}
1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

For the initial run and unless otherwise noted the following values for the Gaussian random processes will be used:

\[ E[v(k)] = 0 \text{ for all } k \]

\[
E[v(k)v^T(k)] = \begin{bmatrix}
-20 \times 10^3 & 0 \\
0 & 20 \times 10^3 \\
\end{bmatrix}
\]

\( m^2 = R \text{ for all } k \)
\[
\sigma_y = \begin{bmatrix}
-150 \\
-150
\end{bmatrix}
\] 
\(m = \) the standard deviation of measurement noise.

\(E[w(k)] = 0 \) for all \(k\)

\[
E[w(k)w^T(k)] = \begin{bmatrix}
100 & 0 \\
0 & 100
\end{bmatrix} \quad (\text{m/sec}^2)^2 = \text{cov} \ w \text{ for all } k
\]

\[
\sigma_w = \begin{bmatrix}
-10 \\
-10
\end{bmatrix}
\] 
\(\text{m/sec}^2 = \) the standard deviations of the random forcing input.

The covariance of estimation error matrix is initialized:

\[
P_0 = P(0/-1) = \begin{bmatrix}
-10^2 & 0 & 0 & 0 \\
0 & 10^2 & 0 & 0 \\
0 & 0 & 10^2 & 0 \\
0 & 0 & 0 & 10^2
\end{bmatrix}
\]
Since the filter is to be unbiased, the initialization:

\[ \hat{x}(0/-1) = \bar{x}_0 = \text{Initial condition of the problem.} \]

2. **Nonlinear Tracking**

The state equations are the same as for the linear tracking problem. The measurement equation is considered as a noisy range measurement by the tracking sensor and is characterized as:

\[ z(k) = [x_1^2(k) + x_2^2(k)]^{1/2} + v_1(k) \quad (4-8) \]

Thus \( z(k) \) is a nonlinear function of the states. Using equation (2-14) and with \( T = 1 \) second:

\[ \bar{f}(x(k), u(k), k) = \begin{bmatrix} -x_1(k) + x_2(k) \\ x_2(k) \\ x_3(k) + x_4(k) \\ -x_4(k) \end{bmatrix} \]
Taking partial derivatives of $f$ with respect to $x$: 

$$A(k) = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \hat{x}(k/k), u(k), k \end{bmatrix} = \begin{bmatrix} -1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

using equation (2-15):

$$h(x(k), k) = [x_1^2(k) + x_3^2(k)]^{1/2}$$

and taking the partial derivative with respect to $x$ gives:

$$H(k) = \begin{bmatrix} \frac{\partial h}{\partial x} \\ \hat{x}(k/k-1) \end{bmatrix} = \begin{bmatrix} x_1(k) \\ \frac{x_1^2(k) + x_3^2(k)}{[x_1^2(k) + x_3^2(k)]^{1/2}} \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} x_3(k) \\ \frac{x_3^2(k)}{[x_1^2(k) + x_3^2(k)]^{1/2}} \\ 0 \end{bmatrix}$$

$\hat{x}(k/k-1)$
Using the above results with the same values for the random noise processes as previously stated for the linear case, the extended Kalman filter equations can now be applied to the nonlinear tracking problem.

B. COMPUTER SIMULATION

1. Computer Hardware/Software
   a. Hardware

   All computer simulations were run on the Hewlett-Packard HP-86 personal computer. This particular model was chosen to evaluate its capabilities in determining its usefulness in actual torpedo tracking at the underwater tracking range at Naval Underwater Weapons Engineering Station, Keyport, Washington. The HP-86 system used included keyboard, 9 inch CRT monitor connected through an integrated monitor interface, and two HP Flexible Disc Drives connected through an integrated disc interface. Plotting was done on a HP-7225B Plotter and printing on a HP-2631G Printer. These peripherals were interfaced using a HP-IB Interface module. Because the system uses interface select codes, the HP-IB factory preset code was set at 7, which is the select code for the printer/disc interface. This code however did not work when interfacing with the external plotter and printer, since duplicate select codes are not allowed. So the internally set select code of the HP-IB was set to 8 for proper system operation.
b. Software

The HP-86 has 60K built-in, useable bytes of computer memory, expandable to 572K using either 32K, 64K, or 128K Memory Modules. A HP-86 plug-in ROM was required to operate the external plotter. Also a Matrix ROM was used to reduce program length and computer run time.

All programs were written in BASIC programming language using REAL (full) precision, which provides 15 digit precision. Appendix B provides an explanation of the program options and Appendix C contains the program listings used for this study.

2. Track Generation

To evaluate the real-time use of the error ellipse as an indicator of filter performance, Monte Carlo simulation runs were made. Four tracks were generated by separate programs and one second incremental values of $x$, $y$, $v_x$, and $v_y$ were stored in data files. Appendix I contains an explanation of the generation of tracks three and four.

3. Noise Generation

In order to simulate the random noise processes, the computer's random number generator was used and the generated numbers scaled accordingly. For each track and each different value of noise sigma, a different generator "seed number" was used. These noise values produced were added to the applicable true track values to simulate a sensor measurement corrupted by independent Gaussian noise. For all
cases where filter parameters were varied for a particular track under a specific noise condition, one noisy track was generated, stored, and used throughout that particular simulation. This was done for ease of filter performance comparison.

4. Gating Scheme

In order to preclude catastrophic filter failure due to excessive measurement noise, a bound was established for the maximum acceptable limits of measurement noise. A three-sigma gate was designed using the covariance of the measurement noise, $R$, and the predicted covariance of error matrix $P(k/k-1)$. The gate is defined as:

$$\text{Gate}(k) = 3(p_{ii}(k/k-1) + R_{ii})^{1/2}$$

This gate is the maximum error allowable for the measurement at time $k$. If the absolute difference between the actual measurement received and the predicted measurement is greater than the three-sigma gate, then that particular measurement data is rejected as unacceptable. When this occurs, the filter gain, $G(k)$, is set to zero, resulting in that measurement being ignored and the prediction of the states set equal to the estimate, that is:

$$x(k/k) = x(k/k+1)$$
5. Collection of Statistics

In order to study error ellipses as an indicator of filter performance, statistics were calculated, on line, after each measurement during the Monte Carlo run. The statistics computed were relative error (in some cases the error was normalized), error mean, error variance, and error covariance for the positional variables. The following equations apply:

Relative Error = \( e(k/k) = \hat{x}(k) - \bar{x}(k/k) \)

(filter error residual)

Error Mean = \( \bar{e}(k/k) = \frac{1}{k} \sum_{j=1}^{k} e(j/j) \)

Error Variance = \( \text{Var}[e_i(k/k)] = \frac{1}{k} \sum_{j=1}^{k} [e_i(j/j) - \bar{e}_i(k/k)]^2 \)

\( i=x_1, x_2, x_3, x_4 \)

Positional Error Covariance Matrix

\[
\begin{bmatrix}
\text{Var}[e_{x_1}(k/k)] & 1/k \sum_{j=1}^{k} [e_{x_1}(j/j)][e_{x_1}(j/j)] & 1/k \sum_{j=1}^{k} [e_{x_1}(j/j)][e_{x_3}(j/j)] \\
1/k \sum_{j=1}^{k} [e_{x_1}(j/j)][e_{x_3}(j/j)] & \text{Var}[e_{x_3}(k/k)] & 1/k \sum_{j=1}^{k} [e_{x_3}(j/j)][e_{x_3}(j/j)] \\
-1/k \sum_{j=1}^{k} [e_{x_1}(j/j)][e_{x_3}(j/j)] & -1/k \sum_{j=1}^{k} [e_{x_3}(j/j)][e_{x_3}(j/j)] & \text{Var}[e_{x_3}(k/k)]
\end{bmatrix}
\]
V. TARGET TRACKING AND ERROR ELLIPSE ANALYSIS

A. LINEAR TRACKING

Track 1 depicts a target approaching at a constant velocity of 223.6 feet per second. The solid line of Figure 5.1 indicates the true track of the target and the numbers along the track indicate the time in seconds. The target was tracked in a measurement noise, $\sigma_y = 150$, with the random forcing noise $\sigma_w = 1$. $R$ was set for 20,000. The dots indicate the filtered track using the linear Kalman filter equations. Figures 5.2 and 5.3 are the filter error ellipses for this run, computed at increments of 10 seconds. As can be seen the ellipse size decreases with increasing time indicating filter convergence. The computed ellipse surface areas shown on the figures confirm this. Figure 5.4 shows the filtered track for the case where $\sigma_y$ has been increased to 300 and all other parameters remain the same. The error ellipses of Figure 5.5 computed for 10-second increments show increasing area indicating filter divergence. Figure 5.6 shows the error ellipses for the same track run but this time the ellipses were computed using a "statistics window" of 10. By this is meant that the ellipses were derived from statistics computed for the last 10 data measurements. All previous data is disregarded. Using this
method, the ellipses of Figure 5.6 show filter convergence from iterations 15 to 25. The filtered track of Figure 5.4 confirms this. Figure 5.7 shows the results for the same track parameters, except in this case a statistics window of 5 was used. The window 5 ellipse area at time 25 (5173 sq ft) is much less than the area of either the run with the statistic window of 10 (11,540 sq ft) or the run with no window at all (65,500 sq ft). This is expected since the filter is essentially "locked on" at time 20, and the window 5 ellipse at time 25 disregards all data previous to time 20. Figure 5.8 shows the error ellipses for the same track but the measurement noise was increased to $\sigma_m = 400$, while $R$ was kept at 20,000. The error ellipses indicate filter divergence, and indeed the filtered track headed off in the wrong direction.

Track 2 depicts a target approaching at a constant speed of 500 feet per second in the -y direction. Figure 5.9 depicts the solid line track and the dots indicate the linear filtered track with $\sigma_v = 150, \sigma_m = 1$ and $R = 20,000$. Figure 5.10 and 5.11 are the error ellipses for the run. No statistics window was used. Other than the fact that the ellipse area is decreasing, the shape of the ellipse provides little additional information. Figure 5.12 and 5.13 show the ellipses for the normalized error. With the same measurement noise sigma for both the x-position and y-position measurements, the normalized error ellipse's shape and orientation reflect
the target track proximity to either axis. In Figure 5.13, the ellipse major axis indicates a large normalized error in the x-direction. This is to be expected since the target maintains a constant x-position of 500 feet, while the y-position is initially 20,000 feet. However, near time 40 as the target approaches the x-axis, the normalized y error increases and becomes so large at the x-axis crossing that the normalized x error becomes insignificant in comparison. This can be seen in Figure 5.13 for the ellipses at time 45 and 55 compared with the ellipse at time 35. The normalized error ellipse is an excellent indicator of target proximity to an axis, but the rapid shifts in ellipse surface area make it difficult to determine filter convergence.

Track 3 depicts a target approaching on a parabolic track at a speed of 203 feet per second. Figure 5.14 is the true track, with a linearly filtered track indicated by the dots. For this run $a_v=150$, $a_w=10$, and $R=20,000$. Figure 5.15 are the error ellipse plots for times 40, 50, and 60, the period of the highest rate of change in x- and y-velocity. The ellipse areas increase with time indicating divergence. Figure 5.16 and 5.17 are the ellipse plots using a 10-data point and a 5-data point statistics window respectively. In both cases the ellipse areas for time 60 are less than time 50 indicating the filter has tracked around the curve.
Using error ellipses based on a statistics window in this tracking situation provides a better indicator of filter performance.

For the second run of track 3, $a_y$ was increased to 300 and all other parameters remained the same. Figure 5.18 shows the resulting filtered track, which obviously didn't track around the curve. With the large amount of measurement noise and considering that the maximum random forcing input, i.e. the maximum acceleration in the x- and y-direction, occurs between times 40 and 60, it is a logical place to lose track. Another factor to be considered is the decrease in gain as $k$ increases. By time 40 the gains have little influence. So a system was incorporated in the program to "reinitialize" the filter by setting the gains to $G(0)$, if certain conditions were met. After several trial and error runs, it was determined that if a statistics window of 10 were used, and the gains reinitialized if the error ellipse area increased consecutively a certain number of times, that the filtered track would follow around the curve. Figure 5.19 is the filtered track using a statistics window of 10 and reinitializing the filter if the error ellipse area increased consecutively during five 1-second increments. The error ellipses for that run are shown in Figures 5.20 and 5.21. As indicated on the figures, the filter was reinitialized four times and the ellipse areas for the 10-second increments increased, until reinitializing the
filter at time 52 locked the filter in. Consequently the error ellipse areas for time 60 and 70 decreased, indicating convergence.

In the final run, the filter was reinitialized after 10 consecutive 1-second error ellipse increases, with all other parameters remaining the same. Figure 5.22 is the filtered track; Figures 5.23 and 5.24 are the error ellipse plots for this run. As indicated on Figure 5.24 the filter was reinitialized only once at time 57. A comparison of ellipse areas for the last two runs shows that, with the exception of time 70, the areas were larger in the first run when the filter was reinitialized 4 times. This is expected since reinitializing results in larger gains producing more widely vary estimates, and hence greater error variance. At time 70 the error ellipse area of the first run is less since in the first run the last filter reinitialization occurred at time 52 versus time 57 in the second run. The first filtered track had more time to settle out by time 70, resulting in less error.

B. NONLINEAR TRACKING

Track 4 depicts a target moving at a constant velocity of 50 feet per second (30 kts) in the x-direction for 15 seconds, at which time the target turns and travels in the -y-direction. (See Figure 5.25) Using the extended Kalman filter with $Q_x=30$ and $R=900$, a series of tracking runs were
made for various values of COVW (\(\sigma^2_w\) from 20 to 200). In none of these runs did the filter successfully track the target around the turn. Figure 5.26 shows the results for the case when \(COV=150\). In this instance the filter lost track as the target came out of the turn. Trying to track through a turn using a constant COVW (and hence, a constant \(Q\)) did not work. So a scheme was devised to vary COVW dependent on information derived from the error ellipse. After several trial runs for this particular track, it was determined that if the error ellipse area increased consecutively for 7 iterations of \(k\), COVW would be doubled, and if the ellipse area decreased consecutively for 5 iterations of \(k\), COVW would be halved. Initially the trial runs were made without a statistics window, and the filter did not track successfully. Without the statistics window, the old data weighted down the statistics, and the error ellipses were not indicative of what was currently happening. So it was decided to use a statistics window. Windows of 5, 10, and 15 were tried. Window 5 proved to be too responsive and window 15 not responsive enough. So a statistics window of 10 was chosen for the tracking run. With \(P_o=10^2\), \(a_v=30\), \(R=900\), and COVW initially set at 20, the tracking run was made. Figure 5.27 depicts the filtered track output, and Figures 5.28-5.30 are the 10-second incremental error ellipses for the run. As can be seen, the filter did track around the curve. Figure 5.29 shows the ellipse areas are becoming less between times 55 and 65. Also indicated below the plots are the values of
COVW for the k time the plot was computed. During this particular run COVW varied from an initial value of 20 up to 160, and then decreased to 40 by the end of the run.

Using the same filter parameters as above except $a_y$ was increased to 200, and $R$ to 40,000, another run was made. The filter did not track at all. During this run COVW varied from an initial value of 20 up to 40 and decreased to 5 by the end of the run. Obviously, the criteria for increasing and decreasing COVW was not effective. More trial runs were necessary to determine the optimum consecutive increases or decreases of the ellipse areas before adjusting COVW accordingly.

A second approach to the nonlinear tracking problem, one that was used earlier for the linear case, is to reinitialize the filter if certain conditions are met. Again, using trail and error runs with and without statistics windows, it was determined that using a statistics window of 10 gave the best results. Using initial conditions of COVW=150, $p_o=10^2$, $a_y=30$, and $R=900$, several runs were made, reinitializing the filter if the error ellipse area increased consecutively for a certain number of iterations of k. Of the runs attempted, the best results were obtained when the filter was reinitialized if the area increased for 5 consecutive iterations of k. Figure 5.31 is the filtered track for this run, and Figures 5.32-5.34 are the error ellipse plots. Indicated below the plots are the times, k, when the filter
was reinitialized. For this particular run the filter was reinitialized 5 times. It should be noted that when reinitializing the filter, \( P(k/k-1) \) was reset to \( 10^2 \times P_0 \). 

\( P_0 \) was not large enough to be effective in getting the filter back on track. The initial value of \( 10^2 \) was used for \( P_0 \) to reflect the high confidence in the initial state conditions. Other values of \( P_0 \) did not work as well.

Using the same parameters as above except the filter was reinitialized after 7 consecutive error ellipse area increases, another run was made with the resulting track depicted in Figure 5.35. A comparison with Figure 5.31 reveals that using 7 area increases as the criterion for filter reinitialization resulted in poorer filter performance, as can be shown from the error ellipses.

The final nonlinear filter tracking runs attempted involved simultaneously varying \( COVW \) and filter reinitialization. The results were disastrous, and highly unpredictable. It was an interesting experiment in futility, and no meaningful results could be obtained.
Figure 5.1  Solid Line Track 1, Vel 223.6 ft/sec; Dots Indicate Filtered Track, $a_y = 150$, $a_m = 1$, $R = 20,000$
Figure 5.2 Filtered Track 1 Error Ellipses at 10 Second Increments, $g_y=150$
Figure 5.3 Filtered Track 1 Error Ellipses at 10 Second Increments, $a_y=150$
Figure 5.4 Filtered Track 1, $a_y=300$, $a_w=1$, $R=20,000$
Figure 5.5 Filtered Track 1 Error Ellipses for $a_y = 300$
Figure 5.6  Filtered Track 1 Error Ellipses for $z_w=300$, Using Statistics Window of 10
Figure 5.7 Filtered Track 1 Error Ellipses for $z_y = 300$, Using Statistics Window of 5
Figure 3.3 Filtered Track 1 Error Ellipse for $\sigma_y = 400$
Figure 5.9 Solid Line Track 2, Vel 500 ft/sec, Dots Indicate Filtered Track for $\alpha_u=150$, $\alpha_w=1$, $R=20,000$
Figure 5.10 Filtered Track 2 Error Ellipses, $\sigma_y=150$
Figure 5.11 Filtered Track 2 Error Ellipses, $\sigma_y=150$
Figure 5.12 Filtered Track 2 Error Ellipses, \( \sigma_x = 150 \), Error Normalized
Figure 5.13 Filtered Track 2 Error Ellipses, $a_y = 150$, Error Normalized
Figure 5.14  Solid Line Track 3, Vel 200 ft/sec, Dots Indicate Filtered Track for $\sigma_{x}=150$, $\sigma_{y}=10$, $R=20,000$
Figure 5.15 Filtered Track 3 Error Ellipses, \( \sigma_y = 150 \)
Figure 5.16 Filtered Track & Error Ellipses, $q_y=150$, Statistics Window 10
Figure 5.17 Filtered Track 3 Error Ellipses, $\gamma_v = 150$, Statistics Window 5
Figure 5.18 Filtered Track 3 for $q_y=300$, $q_n=10$, $R=20,000$
Figure 5.19 Filtered Track 3 for $q_y=300$, $q_w=10$, $R=20,000$
Statistics Window 10, Reinitialized at 5
Consecutive Ellipse Area Increments
Figure 5.20 Filtered Track 3 Error Ellipses, $g_o=300$, Statistics Window 10, Reinitialize"5

COVW=100  SIGV=300  R=20000  TIME AREA (SQ FT) LEG RESET

30  4734E+001  --  29
40  7035E+001  ...  35
50  2698E+002  --  44
Figure 5.21 Filtered Track 3 Error Ellipses, $g = 300$, Statistics Window 15, Reinitialize 5
Figure 3.33  Filtered Track 3 for $g_y=300$, $g_w=10$, $R=10,000$
Statistics Window 10, Reinitialize 10
COVW=100  SIGV=300  R=20000  TIME AREA (SQ FT) LEG RESET

30  10245E+001  ---  0
40  1161E+001  ....  0
50  9186E+001  ---  0

Figure 5.20  Filtered Track 3 Error Ellipses, g = 300,
Statistics Window 10, Reinitialize 10
Figure 5.24 Filtered Track 3 Error Ellipses, $a=300$,
Statistics Window 10, Reinitialize 10
Figure 5.25 Track 4, Vel 80 ft/sec
Figure 5.26  Filtered Track 4, $\sigma_y = 30$, $R = 900$, $\sigma_w^2 = 150$
Figure 5.27 Filtered Track 4, $\sigma_v = 30$, $R = 900$, Statistics Window 10, Varying $\sigma_{\dot{v}}$
<table>
<thead>
<tr>
<th>Time</th>
<th>Area (sq ft)</th>
<th>Leg. COVW</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>9688E-002</td>
<td>20</td>
</tr>
<tr>
<td>25</td>
<td>8097E-001</td>
<td>40</td>
</tr>
<tr>
<td>35</td>
<td>2740E+000</td>
<td>160</td>
</tr>
</tbody>
</table>

Figure 5.28 Filtered Track 4 Error Ellipses, Statistics Window 10, Varying COVW
Figure 5.29 Filtered Track 4 Error Ellipses, Statistics Window 10, Varying COVW
Figure 5.30 Filtered Track 4 Error Ellipses, Statistics
Window 10, Varying COVW
Figure 5.31 Filtered Track 4, $\sigma_y = 30$, $R = 900$, $\sigma^2 = 150$, Reinitialize 5
Figure 5.32 Filtered Track 4 Error Ellipses, \( \lambda_y = 30 \),
Reinitialize 5
Figure 5.33 Filtered Track 4 Error Ellipses, \( \omega_\varphi = 30 \),
Reinitialize 5
Figure 5.34 Filtered Track & Error Ellipses, $\gamma_\nu = 30$, Reinitialize S
Figure 5.35 Filtered Track 4, $\Delta v = 30$, $R = 900$, $\hat{d}_W = 150$, Reinitialize 7
VI. CONCLUSIONS AND RECOMMENDATIONS

A. ERROR ELLIPSE

The filter error ellipse proved useful as a tool for indicating filter performance. The information provided by the ellipse, particularly surface area changes, was used to make decisions concerning the alteration of the filter parameters. Several approaches for using the error ellipse were applied to both the linear and nonlinear tracking problem and the results are summarized below:

<table>
<thead>
<tr>
<th>Procedure</th>
<th>Applicable Filter (Linear or Extended)</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Statistics Window</td>
<td>Both</td>
<td>Useful in keeping the error ellipse current and responsive to present data. The ellipse reflects most recent data; old data is disregarded. Beneficial when making a decision concerning filter parameter modification. Normally a better indicator of filter convergence or divergence than without a statistics window.</td>
</tr>
<tr>
<td>Normalized Error Ellipse</td>
<td>Both</td>
<td>Aid in displaying error trends as target approaches coordinate axis or origin. Not practical in determining filter convergence/divergence due to rapid ellipse changes in vicinity of axes.</td>
</tr>
<tr>
<td>Reinitialize Linear-Filter</td>
<td>Set $F(1)$</td>
<td></td>
</tr>
<tr>
<td>---------------------------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>Increasing Extended-Set</td>
<td>$F(k+1/x)$</td>
<td></td>
</tr>
<tr>
<td>Ellipse Area $F(k+1/x)$</td>
<td>$L_{11}^{(1)} + L_{12}^{(1)}$</td>
<td></td>
</tr>
</tbody>
</table>

Use increasing ellipse size as an indicator of divergence, and is a decision-making device to reinitialize filter. Particularly valuable later in track where gains and $F(k+1/x)$ have settled out. More effective when used in conjunction with statistics window.

<table>
<thead>
<tr>
<th>Error Ellipse Expansion/Compression to vary COVW (Adaptive Q)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error Extended Ellispe Expansion/Compression to vary COVW (Adaptive Q)</td>
</tr>
</tbody>
</table>

This technique of varying COVW based on error ellipse area increase or decrease is particularly useful in a tracking environment containing large variations in the random forcing input. The procedure is more effective when used in conjunction with a statistics window. Using this technique with filter reinitialization was unsuccessful.

B. COMPUTER PERFORMANCE

The HP-86 proved to be an extremely reliable computer with no downtime experienced during the 5 month period of operation. Full (Real) precision was used throughout the study providing 15 digit precision, which was more than adequate. The use of the Matrix ROM reduced program length by 30% and increased computing speed by a factor of about 6. Although not used in this study, a Statistics ROM would have undoubtedly further increased computing speed. For any further study using the HP-86, it is recommended that a Statistics ROM be procured.

It took approximately 2 seconds for each incremental time measurement data to be sequenced through the filter equations, both for the linear and nonlinear case. This
computing time also included all statistics computations. With an incoming data rate of 1 set of measurement data per second, this computing speed is not sufficient for on-line processing. As previously mentioned, the Naval Underwater Tracking Range receives a series of 4 measurement times sequentially every 1.31 seconds. The range's three-dimensional tracking problem will necessarily involve more than the 4 state variables used in this study. Hence greater matrix dimensions resulting in longer computing times can be expected.

The HP-86 CRT graphics were used extensively to provide error ellipse plots during the tracking runs. Using a "no frills" approach to plotting, i.e. plotting without x-y axis or labelling, it took approximately 2.5 seconds per ellipse plot. The ellipse plotting routine used involved sines and cosines, plotted point by point in 30 degree increments for a total of 360 degrees. This method was somewhat slow. Had there been available a graphics program that would sketch in the ellipse around the intersected major and minor axis, the graphics presentation could have been speeded up. But since ellipse plotting for every 3 to 5 increments of time provided sufficient "real-time" information, the time of 2.5 seconds per plot was tolerable.

Summarizing, the HP-86 could be used to compute statistics and provide graphics in the real-time underwater tracking environment, if the graphics were required not more
often than 3 to 5 seconds. However, before the HP-86 can be considered feasible for real-time Kalman filter processing, more investigation is needed in finding ways to speed up computer processing time such as parallel processing, additional use of manufacturer-provided ROMs and machine language programming.
APPENDIX A

TRACK GENERATION

1. TRACK THREE

Target movement is in the x-y plane with the tracking sensor located at the origin of the cartesian axes. The target follows a parabolic track (see Figure A-1) at a constant speed of 200 feet per second.

Figure A-1 Track 3
The parabolic equation is:

$$y^2 = 4p(x-h)$$

where \( p = 1000 \) and \( h = 1000 \), resulting in:

$$y^2 = 4000(x-1000)$$

Initial target location is \((x,y) = (8000, 5291.5)\). Target x-direction velocity is given by:

$$v_x = v \cos(\text{Angle}) \quad (A-1)$$

and target y-direction velocity is:

$$v_y = v \sin(\text{Angle}) \quad (A-2)$$

where the Angle is obtained from:

$$\text{Angle} = \tan^{-1}\left(\frac{\Delta y}{\Delta x}\right)$$

and \( v \) is obtained from:

$$v = (v_x^2 + v_y^2)^{1/2}$$
where the argument of the inverse tangent is the slope of a small increment (less than 1 second) at each successive data point. Using a sampling period of one second, the data points \((x,y)\) can be obtained from:

\[
x(k+1) = x(k) + v_x(k)
\]

\[
y(k+1) = y(k) + v_y(k)
\]

Table A-1 gives the numerical values for the four states.

2. TRACK FOUR

Target movement simulates a 30-knot torpedo (velocity 50 feet per second) at a constant depth. The target's initial position is \((x,y) = (3250, 5000)\), and it is moving in the \(v_x\) direction with \(v_y = 0\). (See Figure A-2.) The target remains on a straight course for 15 seconds, and then executes a 90 degree turn and travels in the \(-y\) direction at \(v_y = -50\) ft/sec. The trajectory of the target turn is described by a 90 degree arc of a circle of radius, \(R = 1000\), with the circle centered at \((x,y) = (4000, 4000)\). The 90 deg arc will be traversed in:

\[
2\pi R / 4v \text{ sec.} = 10\pi \text{ sec. (where } v = 50 \text{ ft/sec)}
\]
<table>
<thead>
<tr>
<th>K</th>
<th>X</th>
<th>X-VEL</th>
<th>Y</th>
<th>Y-VEL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8000.00</td>
<td>-186.90</td>
<td>5291.50</td>
<td>-71.15</td>
</tr>
<tr>
<td>2</td>
<td>7813.10</td>
<td>-186.93</td>
<td>5220.38</td>
<td>-71.13</td>
</tr>
<tr>
<td>3</td>
<td>7626.17</td>
<td>-186.60</td>
<td>5148.27</td>
<td>-71.98</td>
</tr>
<tr>
<td>4</td>
<td>7439.56</td>
<td>-185.89</td>
<td>5075.26</td>
<td>-72.37</td>
</tr>
<tr>
<td>5</td>
<td>7253.33</td>
<td>-185.51</td>
<td>4926.43</td>
<td>-74.74</td>
</tr>
<tr>
<td>6</td>
<td>7067.44</td>
<td>-185.11</td>
<td>4850.54</td>
<td>-75.73</td>
</tr>
<tr>
<td>7</td>
<td>6881.93</td>
<td>-184.68</td>
<td>4773.60</td>
<td>-76.76</td>
</tr>
<tr>
<td>8</td>
<td>6696.62</td>
<td>-184.24</td>
<td>4695.59</td>
<td>-77.83</td>
</tr>
<tr>
<td>9</td>
<td>6512.14</td>
<td>-183.76</td>
<td>4616.45</td>
<td>-78.94</td>
</tr>
<tr>
<td>10</td>
<td>6327.90</td>
<td>-183.26</td>
<td>4536.14</td>
<td>-80.09</td>
</tr>
<tr>
<td>11</td>
<td>6144.14</td>
<td>-182.73</td>
<td>4454.60</td>
<td>-81.30</td>
</tr>
<tr>
<td>12</td>
<td>5960.87</td>
<td>-182.17</td>
<td>4371.79</td>
<td>-82.56</td>
</tr>
<tr>
<td>13</td>
<td>5778.14</td>
<td>-181.57</td>
<td>4287.65</td>
<td>-83.87</td>
</tr>
<tr>
<td>14</td>
<td>5595.98</td>
<td>-180.92</td>
<td>4202.10</td>
<td>-85.24</td>
</tr>
<tr>
<td>15</td>
<td>5414.41</td>
<td>-180.24</td>
<td>4115.09</td>
<td>-86.68</td>
</tr>
<tr>
<td>16</td>
<td>5233.49</td>
<td>-179.51</td>
<td>4026.54</td>
<td>-88.19</td>
</tr>
<tr>
<td>17</td>
<td>5053.25</td>
<td>-178.72</td>
<td>3936.37</td>
<td>-89.78</td>
</tr>
<tr>
<td>18</td>
<td>4873.74</td>
<td>-177.87</td>
<td>3844.49</td>
<td>-91.44</td>
</tr>
<tr>
<td>19</td>
<td>4695.02</td>
<td>-177.96</td>
<td>3750.81</td>
<td>-93.19</td>
</tr>
<tr>
<td>20</td>
<td>4517.15</td>
<td>-177.97</td>
<td>3665.24</td>
<td>-95.04</td>
</tr>
<tr>
<td>21</td>
<td>4340.19</td>
<td>-177.91</td>
<td>3575.65</td>
<td>-97.00</td>
</tr>
<tr>
<td>22</td>
<td>4164.22</td>
<td>-177.74</td>
<td>3487.93</td>
<td>-99.06</td>
</tr>
<tr>
<td>23</td>
<td>3989.31</td>
<td>-177.48</td>
<td>3355.93</td>
<td>-101.25</td>
</tr>
<tr>
<td>24</td>
<td>3815.57</td>
<td>-177.09</td>
<td>3251.52</td>
<td>-103.57</td>
</tr>
<tr>
<td>25</td>
<td>3643.05</td>
<td>-176.57</td>
<td>3144.52</td>
<td>-106.05</td>
</tr>
<tr>
<td>26</td>
<td>3472.00</td>
<td>-176.09</td>
<td>3034.75</td>
<td>-108.68</td>
</tr>
<tr>
<td>27</td>
<td>3302.43</td>
<td>-175.57</td>
<td>2922.01</td>
<td>-111.50</td>
</tr>
<tr>
<td>28</td>
<td>3134.54</td>
<td>-175.04</td>
<td>2806.06</td>
<td>-114.51</td>
</tr>
<tr>
<td>29</td>
<td>2968.50</td>
<td>-174.98</td>
<td>2686.65</td>
<td>-117.74</td>
</tr>
<tr>
<td>30</td>
<td>2804.52</td>
<td>-174.67</td>
<td>2563.47</td>
<td>-121.21</td>
</tr>
<tr>
<td>31</td>
<td>2642.85</td>
<td>-174.09</td>
<td>2436.20</td>
<td>-124.04</td>
</tr>
<tr>
<td>32</td>
<td>2483.76</td>
<td>-173.17</td>
<td>2304.42</td>
<td>-128.98</td>
</tr>
<tr>
<td>33</td>
<td>2327.59</td>
<td>-172.86</td>
<td>2167.70</td>
<td>-133.33</td>
</tr>
<tr>
<td>34</td>
<td>2174.74</td>
<td>-172.07</td>
<td>2025.50</td>
<td>-138.05</td>
</tr>
<tr>
<td>35</td>
<td>2025.66</td>
<td>-171.72</td>
<td>1874.18</td>
<td>-143.15</td>
</tr>
<tr>
<td>36</td>
<td>1880.95</td>
<td>-171.37</td>
<td>1721.95</td>
<td>-148.67</td>
</tr>
<tr>
<td>37</td>
<td>1741.28</td>
<td>-171.06</td>
<td>1558.85</td>
<td>-154.64</td>
</tr>
<tr>
<td>38</td>
<td>1607.50</td>
<td>-170.83</td>
<td>1386.61</td>
<td>-161.05</td>
</tr>
<tr>
<td>39</td>
<td>1480.67</td>
<td>-170.59</td>
<td>1203.46</td>
<td>-167.38</td>
</tr>
<tr>
<td>40</td>
<td>1362.08</td>
<td>-170.70</td>
<td>1006.73</td>
<td>-175.05</td>
</tr>
<tr>
<td>41</td>
<td>1253.38</td>
<td>-170.73</td>
<td>791.58</td>
<td>-182.41</td>
</tr>
<tr>
<td>42</td>
<td>1156.65</td>
<td>-170.01</td>
<td>546.41</td>
<td>-189.67</td>
</tr>
<tr>
<td>43</td>
<td>1074.64</td>
<td>-169.44</td>
<td>211.63</td>
<td>-196.50</td>
</tr>
<tr>
<td>44</td>
<td>1011.20</td>
<td>-168.74</td>
<td>0.00</td>
<td>-200.00</td>
</tr>
<tr>
<td>45</td>
<td>9000.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>
### TABLE A-1 (CONT.)

<table>
<thead>
<tr>
<th>K</th>
<th>X</th>
<th>X-VEL</th>
<th>Y</th>
<th>Y-VEL</th>
</tr>
</thead>
<tbody>
<tr>
<td>46</td>
<td>1011.20</td>
<td>10.57</td>
<td>-211.63</td>
<td>-199.72</td>
</tr>
<tr>
<td>47</td>
<td>1021.77</td>
<td>25.14</td>
<td>-295.07</td>
<td>-198.41</td>
</tr>
<tr>
<td>48</td>
<td>1046.90</td>
<td>35.82</td>
<td>-433.13</td>
<td>-196.77</td>
</tr>
<tr>
<td>49</td>
<td>1082.72</td>
<td>48.89</td>
<td>-725.57</td>
<td>-193.93</td>
</tr>
<tr>
<td>50</td>
<td>1131.61</td>
<td>74.49</td>
<td>-879.69</td>
<td>-185.61</td>
</tr>
<tr>
<td>51</td>
<td>1207.95</td>
<td>86.36</td>
<td>-1035.28</td>
<td>-180.39</td>
</tr>
<tr>
<td>52</td>
<td>1354.31</td>
<td>97.25</td>
<td>-1190.48</td>
<td>-174.77</td>
</tr>
<tr>
<td>53</td>
<td>1451.56</td>
<td>107.04</td>
<td>-1342.96</td>
<td>-168.94</td>
</tr>
<tr>
<td>54</td>
<td>1558.60</td>
<td>115.75</td>
<td>-1494.80</td>
<td>-163.10</td>
</tr>
<tr>
<td>55</td>
<td>1674.35</td>
<td>123.43</td>
<td>-1642.38</td>
<td>-157.37</td>
</tr>
<tr>
<td>56</td>
<td>1797.78</td>
<td>130.16</td>
<td>-1786.37</td>
<td>-151.35</td>
</tr>
<tr>
<td>57</td>
<td>1927.94</td>
<td>136.06</td>
<td>-1926.60</td>
<td>-146.58</td>
</tr>
<tr>
<td>58</td>
<td>2064.01</td>
<td>141.24</td>
<td>-2063.01</td>
<td>-141.61</td>
</tr>
<tr>
<td>59</td>
<td>2205.24</td>
<td>145.78</td>
<td>-2195.67</td>
<td>-136.92</td>
</tr>
<tr>
<td>60</td>
<td>2351.02</td>
<td>149.79</td>
<td>-2324.67</td>
<td>-132.54</td>
</tr>
<tr>
<td>61</td>
<td>2500.80</td>
<td>153.13</td>
<td>-2450.14</td>
<td>-128.43</td>
</tr>
<tr>
<td>62</td>
<td>2654.11</td>
<td>156.45</td>
<td>-2572.25</td>
<td>-124.60</td>
</tr>
<tr>
<td>63</td>
<td>2810.56</td>
<td>159.24</td>
<td>-2691.14</td>
<td>-121.01</td>
</tr>
<tr>
<td>64</td>
<td>2969.80</td>
<td>161.73</td>
<td>-2806.99</td>
<td>-117.66</td>
</tr>
<tr>
<td>65</td>
<td>3131.52</td>
<td>163.97</td>
<td>-2919.55</td>
<td>-114.52</td>
</tr>
<tr>
<td>66</td>
<td>3293.45</td>
<td>165.98</td>
<td>-3030.17</td>
<td>-111.58</td>
</tr>
<tr>
<td>67</td>
<td>3461.47</td>
<td>167.80</td>
<td>-3137.61</td>
<td>-108.82</td>
</tr>
<tr>
<td>68</td>
<td>3629.27</td>
<td>169.46</td>
<td>-3243.01</td>
<td>-106.23</td>
</tr>
<tr>
<td>69</td>
<td>3798.73</td>
<td>170.96</td>
<td>-3345.88</td>
<td>-103.79</td>
</tr>
<tr>
<td>70</td>
<td>3969.69</td>
<td>172.34</td>
<td>-3446.56</td>
<td>-101.49</td>
</tr>
<tr>
<td>71</td>
<td>4142.03</td>
<td>173.60</td>
<td>-3545.15</td>
<td>-99.32</td>
</tr>
<tr>
<td>72</td>
<td>4315.63</td>
<td>174.76</td>
<td>-3641.77</td>
<td>-97.26</td>
</tr>
<tr>
<td>73</td>
<td>4490.38</td>
<td>175.82</td>
<td>-3736.51</td>
<td>-95.32</td>
</tr>
<tr>
<td>74</td>
<td>4666.21</td>
<td>176.81</td>
<td>-3829.47</td>
<td>-93.48</td>
</tr>
<tr>
<td>75</td>
<td>4843.02</td>
<td>177.73</td>
<td>-3920.72</td>
<td>-91.73</td>
</tr>
<tr>
<td>76</td>
<td>5020.74</td>
<td>178.57</td>
<td>-4010.36</td>
<td>-90.06</td>
</tr>
<tr>
<td>77</td>
<td>5199.32</td>
<td>179.36</td>
<td>-4098.45</td>
<td>-88.48</td>
</tr>
<tr>
<td>78</td>
<td>5378.68</td>
<td>180.10</td>
<td>-4185.06</td>
<td>-86.97</td>
</tr>
<tr>
<td>79</td>
<td>5558.78</td>
<td>180.79</td>
<td>-4270.26</td>
<td>-85.53</td>
</tr>
<tr>
<td>80</td>
<td>5739.57</td>
<td>181.44</td>
<td>-4354.11</td>
<td>-84.15</td>
</tr>
<tr>
<td>81</td>
<td>5921.01</td>
<td>182.04</td>
<td>-4436.67</td>
<td>-82.83</td>
</tr>
<tr>
<td>82</td>
<td>6103.05</td>
<td>182.61</td>
<td>-4517.99</td>
<td>-81.57</td>
</tr>
<tr>
<td>83</td>
<td>6285.66</td>
<td>183.14</td>
<td>-4598.11</td>
<td>-80.36</td>
</tr>
<tr>
<td>84</td>
<td>6468.80</td>
<td>183.65</td>
<td>-4677.09</td>
<td>-79.20</td>
</tr>
<tr>
<td>85</td>
<td>6652.45</td>
<td>184.13</td>
<td>-4754.98</td>
<td>-78.09</td>
</tr>
<tr>
<td>86</td>
<td>6836.58</td>
<td>184.58</td>
<td>-4831.80</td>
<td>-77.01</td>
</tr>
<tr>
<td>87</td>
<td>7021.16</td>
<td>185.00</td>
<td>-4907.61</td>
<td>-75.98</td>
</tr>
<tr>
<td>88</td>
<td>7206.16</td>
<td>185.41</td>
<td>-4982.43</td>
<td>-74.99</td>
</tr>
<tr>
<td>89</td>
<td>7391.57</td>
<td>185.79</td>
<td>-5056.31</td>
<td>-74.03</td>
</tr>
</tbody>
</table>
So each second:

\[
\frac{2\pi/4}{10\pi} = 0.05 \text{ radians will be traversed}
\]

Using the trigonometric identity:

\[
\sin^2(A) + \cos^2(A) = 1
\]
And the equation for a circle:

\[ x^2 + y^2 = c^2 \]

It follows that the arc of Figure A-2 is described by:

\[ x(k) = 4000 + 1000\sin(0.05k) \]
\[ y(k) = 4000 + 1000\cos(0.05k) \]

where the angle argument is in radians and \( k=0,1,2,...,31 \) seconds. The velocities \( v_x \) and \( v_y \) can be obtained from:

\[ v_x(k) = 50\cos(0.05k) \]
\[ v_y(k) = -50\sin(0.05k) \]

using the same angle argument.

The track values for the track arc are contained in Table A-2.
### TABLE A-2

**Numerical Values of States for Track Four**

<table>
<thead>
<tr>
<th>K</th>
<th>X</th>
<th>X-VEL</th>
<th>Y</th>
<th>Y-VEL</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>3049.95</td>
<td>49.98</td>
<td>4999.38</td>
<td>-1.25</td>
</tr>
<tr>
<td>13</td>
<td>3099.56</td>
<td>49.94</td>
<td>4997.50</td>
<td>-2.50</td>
</tr>
<tr>
<td>14</td>
<td>3149.66</td>
<td>49.86</td>
<td>4994.38</td>
<td>-3.75</td>
</tr>
<tr>
<td>15</td>
<td>3199.67</td>
<td>49.75</td>
<td>4990.01</td>
<td>-4.99</td>
</tr>
<tr>
<td>16</td>
<td>3249.35</td>
<td>49.61</td>
<td>4984.40</td>
<td>-6.23</td>
</tr>
<tr>
<td>17</td>
<td>3298.88</td>
<td>49.44</td>
<td>4977.54</td>
<td>-7.47</td>
</tr>
<tr>
<td>18</td>
<td>3348.22</td>
<td>49.24</td>
<td>4969.45</td>
<td>-8.71</td>
</tr>
<tr>
<td>19</td>
<td>3397.34</td>
<td>49.00</td>
<td>4960.13</td>
<td>-9.93</td>
</tr>
<tr>
<td>20</td>
<td>3446.21</td>
<td>48.74</td>
<td>4949.59</td>
<td>-11.16</td>
</tr>
<tr>
<td>21</td>
<td>3494.61</td>
<td>48.45</td>
<td>4937.82</td>
<td>-12.37</td>
</tr>
<tr>
<td>22</td>
<td>3543.05</td>
<td>48.12</td>
<td>4924.65</td>
<td>-13.58</td>
</tr>
<tr>
<td>23</td>
<td>3591.04</td>
<td>47.77</td>
<td>4910.67</td>
<td>-14.78</td>
</tr>
<tr>
<td>24</td>
<td>3638.62</td>
<td>47.38</td>
<td>4895.30</td>
<td>-15.97</td>
</tr>
<tr>
<td>25</td>
<td>3685.80</td>
<td>46.97</td>
<td>4878.75</td>
<td>-17.14</td>
</tr>
<tr>
<td>26</td>
<td>3732.55</td>
<td>46.53</td>
<td>4861.02</td>
<td>-18.31</td>
</tr>
<tr>
<td>27</td>
<td>3778.64</td>
<td>46.05</td>
<td>4842.12</td>
<td>-19.47</td>
</tr>
<tr>
<td>28</td>
<td>3824.64</td>
<td>45.55</td>
<td>4822.08</td>
<td>-20.62</td>
</tr>
<tr>
<td>29</td>
<td>3869.53</td>
<td>45.02</td>
<td>4800.89</td>
<td>-21.75</td>
</tr>
<tr>
<td>30</td>
<td>3914.68</td>
<td>44.46</td>
<td>4778.59</td>
<td>-22.87</td>
</tr>
<tr>
<td>31</td>
<td>3958.65</td>
<td>43.88</td>
<td>4755.17</td>
<td>-23.97</td>
</tr>
<tr>
<td>32</td>
<td>4002.43</td>
<td>43.27</td>
<td>4730.65</td>
<td>-25.06</td>
</tr>
<tr>
<td>33</td>
<td>4045.37</td>
<td>42.63</td>
<td>4705.05</td>
<td>-26.13</td>
</tr>
<tr>
<td>34</td>
<td>4087.67</td>
<td>41.96</td>
<td>4678.38</td>
<td>-27.19</td>
</tr>
<tr>
<td>35</td>
<td>4129.26</td>
<td>41.27</td>
<td>4650.67</td>
<td>-28.23</td>
</tr>
<tr>
<td>36</td>
<td>4170.19</td>
<td>40.55</td>
<td>4621.63</td>
<td>-29.25</td>
</tr>
<tr>
<td>37</td>
<td>4210.37</td>
<td>39.80</td>
<td>4592.17</td>
<td>-30.26</td>
</tr>
<tr>
<td>38</td>
<td>4249.75</td>
<td>39.04</td>
<td>4561.41</td>
<td>-31.24</td>
</tr>
<tr>
<td>39</td>
<td>4288.44</td>
<td>38.24</td>
<td>4529.68</td>
<td>-32.21</td>
</tr>
<tr>
<td>40</td>
<td>4326.27</td>
<td>37.42</td>
<td>4497.00</td>
<td>-33.16</td>
</tr>
<tr>
<td>41</td>
<td>4363.28</td>
<td>36.58</td>
<td>4463.38</td>
<td>-34.08</td>
</tr>
<tr>
<td>42</td>
<td>4399.43</td>
<td>35.72</td>
<td>4428.84</td>
<td>-34.99</td>
</tr>
<tr>
<td>43</td>
<td>4434.71</td>
<td>34.84</td>
<td>4393.41</td>
<td>-35.87</td>
</tr>
<tr>
<td>44</td>
<td>4469.10</td>
<td>33.93</td>
<td>4357.11</td>
<td>-36.73</td>
</tr>
<tr>
<td>45</td>
<td>4502.56</td>
<td>33.00</td>
<td>4319.97</td>
<td>-37.56</td>
</tr>
<tr>
<td>46</td>
<td>4535.09</td>
<td>32.05</td>
<td>4281.59</td>
<td>-38.38</td>
</tr>
<tr>
<td>47</td>
<td>4566.65</td>
<td>31.09</td>
<td>4243.22</td>
<td>-39.17</td>
</tr>
</tbody>
</table>
### Table A-2 (Cont.)

<table>
<thead>
<tr>
<th>k</th>
<th>X</th>
<th>X-Vel</th>
<th>Y</th>
<th>Y-Vel</th>
</tr>
</thead>
<tbody>
<tr>
<td>48</td>
<td>4597.24</td>
<td>30.09</td>
<td>4203.67</td>
<td>-39.93</td>
</tr>
<tr>
<td>49</td>
<td>4626.63</td>
<td>29.08</td>
<td>4163.37</td>
<td>-40.67</td>
</tr>
<tr>
<td>50</td>
<td>4655.40</td>
<td>28.06</td>
<td>4122.34</td>
<td>-41.39</td>
</tr>
<tr>
<td>51</td>
<td>4682.54</td>
<td>27.02</td>
<td>4080.60</td>
<td>-42.07</td>
</tr>
<tr>
<td>52</td>
<td>4709.43</td>
<td>25.95</td>
<td>4038.20</td>
<td>-42.74</td>
</tr>
<tr>
<td>53</td>
<td>4734.65</td>
<td>24.88</td>
<td>3995.14</td>
<td>-43.37</td>
</tr>
<tr>
<td>54</td>
<td>4759.18</td>
<td>23.79</td>
<td>3951.46</td>
<td>-43.98</td>
</tr>
<tr>
<td>55</td>
<td>4782.41</td>
<td>22.68</td>
<td>3907.19</td>
<td>-44.56</td>
</tr>
<tr>
<td>56</td>
<td>4804.54</td>
<td>21.56</td>
<td>3862.35</td>
<td>-45.11</td>
</tr>
<tr>
<td>57</td>
<td>4825.53</td>
<td>20.42</td>
<td>3816.97</td>
<td>-45.64</td>
</tr>
<tr>
<td>58</td>
<td>4845.38</td>
<td>19.28</td>
<td>3771.09</td>
<td>-46.13</td>
</tr>
<tr>
<td>59</td>
<td>4864.08</td>
<td>18.12</td>
<td>3724.72</td>
<td>-46.60</td>
</tr>
<tr>
<td>60</td>
<td>4881.61</td>
<td>16.95</td>
<td>3677.69</td>
<td>-47.04</td>
</tr>
<tr>
<td>61</td>
<td>4897.57</td>
<td>15.77</td>
<td>3630.64</td>
<td>-47.45</td>
</tr>
<tr>
<td>62</td>
<td>4913.14</td>
<td>14.56</td>
<td>3583.60</td>
<td>-47.83</td>
</tr>
<tr>
<td>63</td>
<td>4927.12</td>
<td>13.37</td>
<td>3535.00</td>
<td>-48.18</td>
</tr>
<tr>
<td>64</td>
<td>4939.69</td>
<td>12.17</td>
<td>3486.66</td>
<td>-48.50</td>
</tr>
<tr>
<td>65</td>
<td>4951.45</td>
<td>10.95</td>
<td>3438.01</td>
<td>-48.79</td>
</tr>
<tr>
<td>66</td>
<td>4961.79</td>
<td>9.73</td>
<td>3389.10</td>
<td>-49.04</td>
</tr>
<tr>
<td>67</td>
<td>4970.50</td>
<td>8.50</td>
<td>3339.93</td>
<td>-49.27</td>
</tr>
<tr>
<td>68</td>
<td>4978.78</td>
<td>7.26</td>
<td>3290.56</td>
<td>-49.47</td>
</tr>
<tr>
<td>69</td>
<td>4985.43</td>
<td>6.03</td>
<td>3241.01</td>
<td>-49.64</td>
</tr>
<tr>
<td>70</td>
<td>4990.83</td>
<td>4.78</td>
<td>3191.30</td>
<td>-49.77</td>
</tr>
<tr>
<td>71</td>
<td>4994.55</td>
<td>3.54</td>
<td>3141.47</td>
<td>-49.87</td>
</tr>
<tr>
<td>72</td>
<td>4997.90</td>
<td>2.29</td>
<td>3091.56</td>
<td>-49.95</td>
</tr>
<tr>
<td>73</td>
<td>4999.57</td>
<td>1.04</td>
<td>3041.59</td>
<td>-49.99</td>
</tr>
<tr>
<td>74</td>
<td>4999.98</td>
<td>-0.21</td>
<td>2991.59</td>
<td>-50.00</td>
</tr>
</tbody>
</table>
APPENDIX 2
COMPUTER PROGRAM EXPLANATION

1. LINKAL

The LINKAL program computes the filter gains, GAIN(4,2), for the 4-state system and stores the gains in "LINGAIN.STORAG". The theoretical covariance of error matrix, PKX(4,4), is also computed and stored in "LINCOV.STORAG". Several different sets of gain and covariance values were computed and stored for various values of measurement noise, RMAT(2,2), and random forcing noise, COVW(2,2).

2. LINEST

The LINEST program retrieves the appropriate gain schedule from storage and computes the optimal estimate, XHAT(4,1), and the optimal one-step prediction, XHK1X(4,1). The following capabilities are contained in the LINEST program:

a. Gating Scheme

If the absolute difference (DIFF) between the one-step prediction, XHK1X and the noisy track value, ZMAT, is greater than the three-sigma gate, then the GAIN matrix is disregarded and XHAT is set equal to XHK1X.

b. Track Noise

By setting NOITRAK = 1, the random number generator, RND, and the resulting simulated noise produced, W1 and W2,
are bypassed and the track values corrupted with noise, XHAT, are retrieved from data file "NOITRAK.STOPAS". If NOITRAK=1, then the random number generator is "reseeded" for each program run, producing a different set of noise values resulting in a unique noise corrupted track for each run.

c. Statistics Window

When WINDOW is set to 0, the filter state statistics are computed after each Monte Carlo run. The error mean, variance and position covariance are computed. If WINDOW is set to an integer, I, such that max k>I>0, the statistics will be computed based on the data compiled during the last I iterations of the simulation. If, for example, WINDOW=10, the computation of statistics will be based on the data obtained during the last ten iterations of k, and all previous data is disregarded.

d. Error Normalization

By setting NORM = 1, the error (ERR), which is defined as the difference between the true track value (TRAK) and the estimate (XHAT), is normalized. For all other values of NORM the relative error is used in computing the statistics.

e. Reinitializing Gains

The program has the option of reinitializing the gains to G(0). This can be done by setting HIGH to an integer I, such that max k>I>0. If the surface area of the ellipsoid...
increases I consecutive times, indicating filter divergence, the gains are reinitialized to G(0). If reinitializing gains is not desired as an option, then HIGH is set to some arbitrary large number greater than max k.

3. EXTKF

The EXTKF program computes the optimal estimate, XHAT, and the optimal one-step prediction, XHK1K, for the 4-state nonlinear tracking problem. The EXTKF program has the options: gating scheme, track noise, statistics window, and error normalization as described for the LINEST program. EXTKF has the following additional options:

a. Reinitializing the Filter

The program has the option of reinitializing the covariance of one-step prediction error matrix, PK1K, by setting it to $10^n \times P_0$, where n is zero or some small integer. If HIGH is set to an integer I, such that max k $>$ I $>$ 0, and the surface area of the ellipse increases I consecutive times, then the PK1K matrix will be reset to $10^n \times P_0$.

b. Adaptive Q

EXTKF has the option of automatically increasing or decreasing the state excitation matrix, Q, under certain conditions by changing the value of the covariance of excitation noise, COVW. INCREASE and DECREASE are set to integers I and J respectively, such that max k $>$ I, J $>$ 0. If the surface area of the error ellipse increases I
consecutive times, COVW is doubled or increased by some factor. Conversely, if the surface area of the error ellipse decreases \( J \) consecutive times, COVW is halved or decreased by some factor. If COVW is changed, increased for example, the value of COVW can be changed again later in the run, if the criteria for either increasing or decreasing is met. If the adaptive \( Q \) is not desired then INCREASE and DECREASE are set to some large number greater than \( k \).
10 !linkal - this program computes the gain schedule and the error covariance matrix schedule for the linear kal filter.

20 option base 1
30 real phi(4,4),bmat(2,4),covw(2,2),rmat(2,2)
40 real pk1k(4,4),imat(4,4),gammat(2,4),cmat(4,4)
50 real temp1(2,4),temp2(4,2),temp3(2,2),gtemp(2,4)
60 real temp4(2,2),temp5(4,2),temp6(4,4),pk1(4,4)
70 assign 1 tc "linqain5 .storag"
80 assign 2 tc "linccv5 .storag"
90 mat read phi
100 mat read bmat
110 data 1,1,0,0,0,0,0,0,1,0,0,0,1
120 mat read cmat
130 data 1,0,0,0,0,0,0,0,1,0,0,0,1
140 mat read gammat
150 data 5,0,1,0,0,0,1,0,0,0,0,1
160 mat read covw
170 data 10c,0,10c
180 mat read rmat
190 data 2500,0,2500
200 mat read pk1k
210 data 10e9,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
220 !************ matrix transpose *************
230 mat cmat = trn (cmat)
240 mat phit = trn (phi)
250 mat gammat = trn (gammat)
260 mat imat = idn
270 mat temp1 = covw*gammat
280 mat temp1 = gamma*temp1
290 !*****************************************************************************
300 ! compute gain schedule, gain() and covariance matrix, p(k/k) using equations (z-10), (z-11), and (z-12).
310 for k = 0 to 60
320 disp "k=":k
330 mat temp2 = pk1k*bmat
340 mat temp3 = cmat*temp2
350 mat gtemp = temp3 + imat
360 mat gtempin = inv (gtemp)
370 mat temp4 = cmat*gtempin
380 mat gain = pk1k*temp4
390 !***************************************************************************** print gain matrix ***************
400 disp "gain"
10 1 i1111111
20 11 this program computes the estimated track for the linear kal-
30 man filter using gain schedule and covariance matrix computed
40 off-line. It also computes statistics and covariance of
50 1 error matrix.
60 1 option base 1
70 real gain(4,2), cmat(2,4), phi(4,4), xhk1k(4,4), temp1(2,1), errsq(4)
80 real pkk(4,4), trak(4,1), zhat(2,1), temp2(2,1), temp3(4,1), akk(3)
90 real xhat(4,1), diff(2), err(4), ssus(4), mean(4), sumsq(4), var(4)
100 real covmat(4,4), mean sq(4), mean sq(4), manprod(4,4), cprod(4,4)
110 real prod(4,4), ccmx(90), wssum(4,90), wssumsq(4,90), wtprod(90)
120 real wman prod(90), wmean(4,90), wman sq(4,90)
130 n = 4 1 number of states
140 s = 1 1 x-state of error ellipse
150 t = 3 1 y-state of error ellipse
160 noise = 300
170 rs = 20000
180 sigw = 10
190 norm = 0
200 pltrea = 0
210 pl = 1
220 statplt = 0
230 noitrak = 1
240 high = 10
250 window = 10
260 first = 5
270 nr = 10
280 chg = 3
290 pt = chg + 5
300 sscale = 0
310 s = 4
320 stp = 10
330 cl = 0
340 count = 0
350 larger = 0
360 reset = 0
370 flag = 0
380 zz = 0

1 "1" if error is to normalized.
2 "1" if only area is printed out, i.e. no ellipse.
3 "1" if plotting on screen, otherwise on plotter.
4 "1" if ellipse plotting from previously computed
5 statistics.
6 "1" if using previous noisy track from "noitrak".
7 "1" if statistics window increases consecu-
8 tively before resetting gains.
9 "1" if no statistics window, otherwise indicate
10 the length of the window.
11 "1" indicate the first iteration of k to be plotted.
12 "1" plot the error ellipse every "nr" iterations of k
13 "1" clear the screen every "chg" plots.
14 "1" if scale is to be changed after clearing.
15 "1" if scale is the scale size of the ellipse.
16 iterate the plot every "stp" degrees
llarge = 0
assign 1 to "estdata storag"
assign 2 to "datatrak7 drive0"
assign 3 to "lingain3 storag"
assign 4 to "trkest3 storag"
assign 5 to "liner3 storag"
assign 6 to "statlin3 storag"
assign 7 to "lincc3 storag"
assign 8 to "noitrak7c storag"
read 1 : phi() , cmat() 
read 2 : ch1k() 
randsize
deg
mat ssum = zero
mat sumsq = zero
mat xrod = zero
for k = 1 to 70
if statplt=1 then 2080
read 7 : pkk{} 
!************ compute 3-sigma gate ***********************
maxpk = pkk(1,1)
for i = 2 to 4
if pkk(i,i)>maxpk then maxpk = pkk(i,i)
next i
gate = 3*sqrt((maxpk + r))
if reset = 0 then 700
assign 3 to "lingain3 storag"
assign 7 to "lincc3 storag"
reset = 0
l = k - 1
mat temp1 = cmat*hk1k
read 2 : trak()
if noitrak = 1 then 830
!***************************** compute track corrupted with noise
*****************************
! 
v1 = noise2*(rand - .5)
v2 = noise2*(rand - .5)
zm = trak(1,1) + v1
zm = trak(2,1) + v2
810 print# 8 : zmat()
820 gto 840
830 read# 8 : zmat()
840 !********** compute the difference for the gate test **********
850 diff(1) = abs(xhk1k[1,1] - zmat[1,1])
860 diff(2) = abs(xhk1k[3,1] - zmat[2,1])
870 !********** compute xhat(k/k) **********
880 !********** print estimated track **********
890 read# 3 : gain4()
900 if diff(1)>gate cr diff(2)>gate then mat xhat=xhk1k @ gto 980
910 mat temp2 = zmat - temp1
920 mat temp3 = gain*temp2
930 mat xhat = xhk1k + temp3
940 gto 980
950 !********** print estimated track **********
960 disp "xhat(\n1;\n)"
970 mat disp xhat,
980 print# 4 ; xhat()
990 !********** compute xhat(k+1/k) **********
1000 !********** compute statistics **********
1010 !********** compute running sum of error **********
1020 !********** compute the mean of the error **********
1030 !********** compute the variance of the error **********
1040 !********** compute the mean of the error **********
1050 !********** compute the variance of the error **********
mat mean = mean(mean) - mean(mean) - mean(mean)
for i = 2 to n
    for j = 1 to i-1
        prod(i, j) = err(i)*err(j)
        prod(i, j) = tprod(i, j) + prod(i, j)
        mprod(i, j) = prod(i, j)/k
next j
for i = 1 to n
    covmat(i, i) = var(i)
next i
for i = 2 to n
    for j = 1 to i-1
        ccovmat(i, j) = mprod(i, j) - mean(i)*mean(j)
        ccovmat(j, i) = ccovmat(i, j)
next j
next i
for i = 1 to 4
    werr(i, k) = err(i)
next i
if k = 1 then 1510
if k>window then 1830
for i = 1 to 4
    wsum(i, k) = werr(i, k) + wsum(i, k-1)
next i
for i = 1 to 4
    wsum(i, k) = werr(i, k)
next i
for i = 1 to 4
    wmean(i, k) = 1/k*wsum(i, k)
next i
for i = 1 to 4
    werrsq(i, k) = werr(i, k)*werr(i, k)
next i
1620 if k = 1 then 1670
1630 for i = 1 to 4
1640 wsumsq(i,k) = werrsq(i,k) + wsumsq(i,k-1)
1650 next i
1660 goto 1700
1670 for i = 1 to 4
1680 wsumsq(i,k) = werrsq(i,k)
1690 next i
1700 for i = 1 to 4
1710 wmeanssq(i,k) = 1/k * wsumsq(i,k)
1720 wmeanssq(i,k) = wmean(i,k) * wmean(i,k)
1730 wyar(i,k) = wmeanssq(i,k) - wmeanssq(i,k)
1740 next i
1750 !************* compute the running mean product **********************
1760 wtmpod(k) = werr(5,k) * werr(t,k)
1770 if k = 1 then 1800*
1780 wprod(k) = wtmpod(k) + wprod(k-1)
1790 goto 1810
1800 wprod(k) = wtmpod(k)
1810 wprod(k) = 1/k * wprod(k)
1820 goto 2050
1830 ! ************* compute the window statistics **********************
1840 for i = 1 to 4
1850 wssum(i,k) = werr(i,k) + wssum(i,k-1) - werr(i,k-window)
1860 next i
1870 !************* compute the mean of errcr **********************
1880 for i = 1 to 4
1890 wmean(i,k) = 1/window * wssum(i,k)
1900 next i
1910 !************* compute the variance of errcr **********************
1920 for i = 1 to 4
1930 werrsq(i,k) = werr(i,k) * werr(i,k)
1940 wssum(i,k) = werrsq(i,k) + wssum(i,k-1) - werrsq(i,k-window)
1950 next i
1960 for i = 1 to 4
1970 wmeanssq(i,k) = 1/window * wssum(i,k)
1980 wmeanssq(i,k) = wmean(i,k) * wmean(i,k)
1990 wyar(i,k) = wmeanssq(i,k) - wmeanssq(i,k)
2000 next i
2010 !************* compute the running product mean **********************
2020 wtmpod(k) = werr(5,k) * werr(t,k)
2030 wprod(k) = wtmpod(k) + wprod(k-1) - wprod(k-window)
2040 wmpnd(k) = 1/wmpnd(k)
2050 covmat(s,s) = wmpnd(s,k)
2060 covmat(t,t) = wmpnd(t,k)
2070 covmat(s,t) = wmpnd(k) - wmean(s,k)*wmean(t,k)
2080 if st<pl+1 then read 6; ccvmat(s,s),covmat(t,t),covmat(s,t)
2090 gto 2100
2100 print 6; covmat(s,s),covmat(t,t),ccvmat(s,t)
2110 if k<2 then 2210
2120 if covmax(k)>ccvmat(k-1) then larger = larger+1 else larger=0
2130 if larger<high then 2210
2140 reset = 1
2150 flag = 1
2160 mat ssum = zer
2170 mat sumsq = zer
2180 mat rcmd = zer
2190 zz = 0
2200 kreset = k
2210 !****************** plot the error ellipse *****************************
2220 if k<first then 3110
2230 if k=first or (k=first) mod nr = 0 then 2240 else 3110
2240 ccunt = ccunt + 1
2250 ekk(1) = covmat(s,s)
2260 ekk(2) = covmat(t,t)
2270 ekk(3) = ccvmat(s,t)
2280 lx = ekk(1)
2290 ly = ekk(2)
2300 max = max (lx,ly)
2310 min = min (lx,ly)
2320 if min <= 0 then 3040
2330 if max/min<=10000 then 2350
2340 if max = lx then llarge = 1 else llarge = 2
2350 if ekk(3)<> 0 then 2380
2360 theta = 0
2370 gto 2450
2380 if ekk(1)<> ekk(2) then 2410
2390 theta = 0
2400 gto 2450
2410 theta = .5*atan (2*ekk(3)/(ekk(1)-ekk(2)))
2420 lx = ekk(1) + ekk(2)/2 + ekk(3)/sin(2*theta)
2430 ly = ekk(1) + ekk(2)/2 - ekk(3)/sin(2*theta)
2440 if lx<0 or ly<0 then 3110
a = sqrt (lx)
b = sqrt (ly)
area = 3.1416 * a * b
if plotarea = 1 then :100
if k = first then 2520
if count = 1 and sscale = 1 then 2520
goto 2550
smax = max (a, k)
p scale = e*s max
ns scale = -(e*s max)
if pl = 1 then plotter is 1 else plotter is 805
if cl > 0 then 2600
pt = chg + .5
clear
if pl = 0 then limit 28,24,1,35,16
if pl = 1 then locate 5,145,16,90 else locate 30,108,18,96
scale nscale, pscale, nscale, pscale
if cl < 0 then 2720
move pscale, 1*nscale
label "y-position (ft)"
move .2*nscale, 1.05*pscale
label "y-position (ft)"
move .9*pscale, 1.05*pscale
label "time": "area(sq ft)"; "legend"
move .4*pscale, 1.3*pscale
label using 2710; "sign="; sigw; " sigw="; noise; " r=" ; r
image k, k, k, k, k, k
if sscale <> 1 and k > first then 2750
if cl = 0 then laxes pscale/2, tscale/2, 0, 0
if cl > 1 and cl = 0 then laxes pscale/2, pscale/2, 0, 0
if pl = 1 then move .8*pscale, pt/(chg + 1)*pscale @ goto 2780
move .9*pscale, pt/(chg + 1)*pscale
if llarge = 0 then 2830
if llarge = 1 then label using 2800; " ; k; " ; "area; " x-axis"
else label using 2800; " ; k; " ; "area; " y-axis
image 2a, dd, 2a, 4de, 8a
llarge = 0
if count = 1 then label using 2870; " ; k; " ; "area; " -----
if count = 2 then label using 2870; " ; k; " ; "area; " -----
if count = 3 then label using 2870; " ; k; " ; "area; " ----
2860 if count = 4 then label using 2870: "":k:""":area:"" ___"
2870 image 2a, dd, 2a, 4de, 6a
2880 if flag = 0 then 2920
2890 movc 1, *nsmallest, pt/(chg+1)*pscale
2900 label "reset at k=":k"reset"
2910 flag = 0
2920 movc 0, 0
2930 if count = 2 then line type 3
2940 if count = 3 then line type 5
2950 if count = 4 then line type 1
2960 fcr ang = 0 tc 360 step step
2970 xx = cos(ang)*a
2980 yy = sin(ang)*b
2990 x1 = xx*ccs(theta) - yy*sin(theta)
3000 y1 = yy*ccs(theta) + xx*sin(theta)
3010 rplot x1, y1
3020 next ang
3030 gcto 3050
3040 disp "k=":k;"" no ellipse plotted. cut of scale."n
3050 cl = cl + 1
3060 pt = pt - 4
3070 if cl = chg then cl = 0
3080 if count = chg then count = 0
3090 gcto 3110
3100 disp "k=":k;"area"
3110 next k
3120 assign# 1 to *
3130 assign# 2 to *
3140 assign# 3 to *
3150 assign# 4 to *
3160 assign# 5 to *
3170 assign# 6 to *
3180 assign# 7 to *
3190 assign# 8 to *
3200 end
10 ! extkfl - this program computes gain schedule, covariance matrix
20 ! and track estimates for a state tracking problem.
30 option base 1
40 deg
50 real hmat(1,4), hmat(4,1), pk1k(4,4), pk2k(4,4), rmat(1,1), trak(4,1)
60 real imat(1,4), xhk1k(4,1), phi(4,4), phi1(4,4), temp1(4,1), temp2(1,1)
70 real TEMP3(1,1), temp3(1,1), temp(4,1), temp2(1,1)
80 real temd(4,1), zmat(1,1), chk1k(1,4), temp(1,1), tempd(4,1)
90 real xhat(4,1), temp(4,1), temp2(4,1), gama(4,2), gamat(2,4)
100 real covw(2,2), temp(4,1), gama(4,2), xhat(4,1), temp(4,1)
110 real sumsq(4), var(4), covmat(4,4), skk(3), meangsq(4), war(4)
120 real meanstack(4), wnpred(4,4), wpred(4,4), wpred(4,4), wpred(4,4), wpred(4,4)
130 real wssum(4,70), wssum(4,70), wssum(4,70), wssum(4,70), wssum(4,70), wssum(4,70)
140 real werr(4,70), werr(4,70), werr(4,70), werr(4,70), werr(4,70), werr(4,70)
150 real wvar(4,70), wcovmat(4,70), wvar(4,70), wcovmat(4,70), wvar(4,70), wcovmat(4,70)
160 real reset(4,4)
170 randomize
180 assign 1 tc "data\trak8 - storag"
190 pltarea = 1  "pl" if only area is being computed, i.e. no ellipse plotting.
200 pl = 1  "pl" if plot on CRT screen, otherwise on plotter.
210 statplt = 0  "st" if plotting previously computed statistics.
220 n = 4  number of states.
230 s = 1  "x"-state of error ellipse.
240 t = 3  "y"-state of error ellipse.
250 pplt = 0  "pplt" if must error ellipse; otherwise statistics ellipse.
260 start = 1  "start" is the iteration of k to begin keeping statistics.
270 high = 70  "last k to be computed (start and <91)."
280 window = 10  "w" if no statistics window, otherwise indicate the length of the window. (plot must equal 0).
300 first = 10  "first iteration of k to be plotted (greater than or equal to start)."
320 chg = 3  "ch" clear the screen every "ch" plots.
330 sscale = 0  "scale is to be changed after clearing.
340 n = 10  "plot error ellipse every "n" iterations of k.
350 norm = 0  "norm" if error is to be normalized.
360 noitrak = 1  "noitrak" if using previously determined noisy track.
370 stp = 10  "stp" iterate plot every "stp" degrees.
380 noise = 30  "standard deviation of noise.
390 covw(1,1) = 20
400 rmat(1,1) = 900
410 increase = 9  # cf consecutive ellipse area increases before
        # increasing q.
420 decrease = 5  # cf consec. ell area decreases before dec. q.
430 cl = 0
440 up = 0
450 down = 0
460 upflag = 0
470 downflag = 0
480 count = 0
490 out = 0
500 llarge = 0
510 mat ssu = zer
520 mat sums = zer
530 mat prod = zer
540 assign# 2 to "stat2set, drive0"
550 assign# 3 to "perr8, storag"
560 assign# 4 to "statmat8a, storag"
570 assign# 5 to "errex8a, storag"
580 assign# 6 to "xhatex8a, storag"
590 assign# 7 to "noitrak0, storag"
600 mat imat = idn
610 mat reset = (100)*imat
620 mat rk1k = reset
630 read# 1 : xhk1k()
640 read# 2 : phi()
650 mat rhit = trn(phi)
660 read# 2 : gama()
670 mat qmat = trn(gama)
680 hmat[1,2] = 0
690 hmat[1,3] = 0
700 covw[1,2] = 0
710 covw[2,1] = 0
720 for k = 1 to high
730 !************ reinitialize if conditions are met ***************
740 if upflag=1 then covw(1,1)=2*covw(1,1) & upflag=0
750 if downflag=1 then ccvw(1,1)=.5*covw(1,1) & downflag=0
760 ccvw(2,2)=covw(1,1)
770 if statpix = 1 then 2480
780 mat tempg = covw*qmat
790 mat qmat = qmat*tempg
800 read# 1 ; trak()
if noitrak = 1 then read# 7 ; zmat() @ goto 850

v(1) = nclise*2*(nt - 5)
zmat(1,1) = sqtr(1); trak(1,1)**2 + trak(3,1)**2 + v(1)

print# 7 : zmat()

********************** conduct 3-sigma gate test **********************

maxpk = pklk(1,1)

fcr 1 = 2 to 4

if pklk(i,i) > maxpk then maxpk = pklk(i,i)

next i

gate = 3*sqr(maxpk + zmat(1,1))

hatrak = sqtr(zklk(1,1)**2 + xhklk(3,1)**2)

diff = ats(hatrak - zmat(1,1))

if diff < gate then 980

mat pkk = pklk

mat xhat = xhklk

print# 6 ; xhat()

goto 1210

j = k - 1

hmat(1,1) = xhklk(1,1)/sqr(xhklk(1,1)**2 + xhklk(3,1)**2)

hmat(1,3) = xhklk(3,1)/sqr(xhklk(1,1)**2 + xhklk(3,1)**2)

mat hmat = trn(hmat)

mat temp1 = pklk*hmat

mat temp2 = hmat*temp1

mat temp3 = temp2 + imat

mat temp4 = hmat*temp3

mat gain = pklk*temp4

mat temp5 = gain*imat

mat temp6 = imat - temp5

mat temp7 = temp6*pmat

mat temp8 = gain*temp7

mat xhat = xhklk*temp8

print# 6 ; xhat()

********************** compute xhat(k+1/k) **********************
**Compute p(k+1/k)**

```c
127C !************ compute p(k+1/k) ************
1280 mat temp9 = pk/k * phi
1290 mat temp5a = phi * temp9
1300 mat pk1k = temp9a + gmat
1310 if k < start then goto 3370
1320 if pplot = 0 then 1370
133C err[1] = pkk(e, s)
1340 err[2] = pkk(t, t)
1350 err[3] = pkk(e, t)
136C print# 3; err
1370 !************ compute statistics ************
139C mat err = trk - xhat
1400 if window > 0 then 1460
141C if norm > 1 then 1470
1420 for i = 1 to n
1430 if trk(i, 1) = 0 then trk(i, 1) = 1
144C next i
1450 mat err = err / trk
146C mat ssin = err * ssin
1470 print# 5; err()
1480 !************ compute the mean of the error ************
149C mat mean = (1/k) * ssin
1500 !************ compute the variance of the error ************
1510 mat errsq = err * err
1520 mat sumsq = errsq + sumsq
1530 mat meansq = (1/k) * sumsq
1540 mat meansq = mean * mean
1550 mat var = meansq - sqansq
1560 !************ compute the running product mean ************
1570 for i = 2 to n
1580 for j = 1 to n
1590 tprod(i, j) = err(i) * err(j)
1600 prod(i, j) = tprod(i, j) + prod(i, j)
161C mprod(i, j) = prod(i, j) / k
1620 next j
163C next i
1640 !************ compute the diagonal terms of the cov of error matrix**
165C for i = 1 to n
166C covmat(i, i) = var(i)
167C next i
1680 !************ compute the off-diagonal terms of cov of error matrix **
169C for i = 2 to n
```
for j = 1 to j-1
  covmat[{i},j] = mprod[i,j] - mean(i)*mean(j)
  ccovmat[{j},i] = covmat[i,j]
next j

1760 for i = 1 to 4
1770 werr[i,k] = err(l)
next i
1790 if k = 1 then 1850
1800 if k>window then 1770
1810 for i = 1 to 4
1820 wssum[i,k] = werr[i,k] + wssum[i,k-1]
next i
1830 goto 1840
1840 wssum[i,k] = werr[i,k]
next i
1880 **************************** compute the mean of the error ****************************
1890 for i = 1 to 4
1900 wmean[i,k] = (1/k)*wssum[i,k]
next i
1920 **************************** compute the variance of the error ****************************
1940 for i = 1 to 4
1950 werrsq[i,k] = werr[i,k]*werr[i,k]
next i
1960 if k = 1 then 2010
1970 for i = 1 to 4
1980 wssumq[i,k] = werrsq[i,k] + wssumq[i,k-1]
next i
2000 goto 2040
2010 wssumq[i,k] = werrsq[i,k]
next i
2030 for i = 1 to 4
2050 wmeanSQ[i,k] = (1/k)*wssumq[i,k]
2060 wmeanSQ[i,k] = wmean[i,k]*wmean[i,k]
2070 wvar[i,k] = wmeanSQ[i,k] - wmeanSQ[i,k]
2080 next i
2090 **************************** compute the running product mean ****************************
2100 wprod[k] = werr[i,k]*werr[t,k]
2110 if k = 1 then 2140
wprod(k) = wprod(k) + wprod(k-1)
goto 2150
wprod(k) = wprod(k)
wmnprod(k) = (7/k)*wprod(k)
goto 2390

compute the window statistics

for i = 1 to 4
  wsum(i,k) = werr(i,k) + wsum(i,k-1) - werr(i,k-window)
next i

compute the mean of the error

for i = 1 to 4
  wmean(i,k) = (1/window)*wsum(i,k)
next i

compute the variance of the error

for i = 1 to 4
  werrsq(i,k) = werr(i,k)*werr(i,k)
  wsumsq(i,k) = werrsq(i,k) + wsumsq(i,k-1) - werrsq(i,k-window)
next i

compute the running product mean

wprod(k) = wprod(k) + wprod(k-1) - wprod(k-window)
wmnprod(k) = (7/window)*wprod(k)
covmat(s,s) = wvar(s,k)
covmat(t,t) = wvar(t,k)
covmat(s,t) = wmnprod(k) - wmean(s,k)*wmean(t,k)
cccov(k) = covmat(s,s)*covmat(t,t)

if k = 1 then 2480
if cccov(k)>cccov(k-1) then up = up + 1 @ down = 0
if cccov(k)<cccov(k-1) then down = down + 1 @ up = 0
if up = increase * then upflag = 1 @ ur = 0
if down = decrease then downflag = 1 @ down = 0
if statplt = 1 then read # 4 ; covmat(s,s),covmat(t,t),
covmat(s,t) @ goto 2500
print # 4 ; covmat(s,s),covmat(t,t),covmat(s,t)
plot error ellipse

if kfirst then 3270
if k = first or (k-first) mod nr = 0 then 2530 else 3370
count = count + 1
if pplect = 1 then 2590
ekk[1] = covmat(s,s)
ekk[2] = covmat(t,t)
ekk[3] = covmat(s,t)
goto 2600
mat ekk = perr
lx = ekk[1]
ly = ekk[2]
max = max(lx,ly)
min = min(lx,ly)
if min<0 then 3290
if max/\min<1000 then 2670
if max/\lx then llarge = 1 else llarge = 2
if ekk(3)<0 then 2700
theta = 0
goto 2770
if ekk(1)< ekk(2) then 2730
theta = 0
goto 2770
theta = .5*atan(2*ekk(3)/ekk(1)-ekk(2))
lx = (ekk(1) + ekk(2))/2 + ekk(3)/sin(2*theta)
ly = (ekk(1) + ekk(2))/2 - ekk(3)/sin(2*theta)
if lx<0 or ly<0 then 3370
a = sqrt(lx)
b = sqrt(ly)
area = .1416*a*b
if pltarea = 1 then 3360
if k = first then 2840
if count = 1 and scale = 1 then 2840
goto 2880
smax = max(a,b)
if smax<.000001 then 3290
pscale=2*smax
nsm = -(2*smax)
if pl = 1 then plotter is 1 else plotter is 805
if cl <> c then 2930
pt = chg - .5
gclear
if pl = 0 then limit 28,241,35,184
if pl = 1 then locate 25,145,16,90 else locate 30,108,18,96
scale nsm,pscale,nsm,pscale
```vbnet
2950 if cl <> 0 then 3020
2960 mcva .8*pscale*.1*pscale
2970 label \"x-position (ft)\"
2980 move .2*rscale,1.05*pscale
2990 label \"y-position (ft)\"
3000 move .9*rscale,1.05*pscale
3010 label \"time\"; \"area (sq ft)\"; \"legend\"
3020 if sscale <> \$ and k <> first then 3050
3030 if cl = 0 then laxes pascal/2,pascal/2,0,0
3040 gto 3060
3050 if pl <> 1 and cl = 0 then laxes pascal/2,pascal/2,0,0
3060 move .9*rscale,1*chg*pscale
3070 if llarge = 0 then 3120
3080 if llarge=1 then label using 3090; \";k;\" \";area;\" \"x-axis\"
else label using 3090; \";k;\" \";area;\" \"y-axis\"
3090 image 2a,dd,2a,4de,7a
3100 llarge = 0
3110 gto 3170
3120 if count = 1 then label using 3160; \";k;\" \":area;\" -----
3130 if count = 2 then label using 3160; \";k;\" \":area;\" -----
3140 if count = 3 then label using 3160; \";k;\" \":area;\" -----
3150 if count = 4 then label using 3160; \";k;\" \":area;\" -----
3160 image 2a,dd,2a,4de,6a
3170 move 0,0
3180 if count = 2 then line type 3
3190 if count = 3 then line type 5
3200 if count = 4 then line type 1
3210 for ang = 0 to 360 step 50
3220 xx = ccs(ang)*t
3230 yy = sin(ang)*t
3240 x1 = xx*cos(theta) - yy*sin(theta)
3250 y1 = yy*cos(theta) + xx*sin(theta)
3260 xplot x1,y1
3270 next ang
3280 gto 3310
3290 disp \"no ellipse plotted, out of scale for k=\";k
3300 gto 3370
3310 cl = cl + 1
3320 pt = pt \$ 4
3330 if cl = chg then cl = 0
3340 if count = chg then count = 0
3350 gto 3370
```
LIST OF REFERENCES


BIBLIOGRAPHY


<table>
<thead>
<tr>
<th>No.</th>
<th>Copies</th>
<th>Distribution List</th>
</tr>
</thead>
</table>
| 1.  | 2      | Defense Technical Information Center  
Cameron Station  
Alexandria, Virginia 22314 |
| 2.  | 2      | Library, Code 0142  
Naval Postgraduate School  
Monterey, California 93943 |
| 3.  | 1      | Chairman, Code 62  
Department of Electrical and Computer Engineering  
Naval Postgraduate School  
Monterey, California 93943 |
| 4.  | 1      | Professor A. Gerba, Jr., Code 62Gz  
Department of Electrical and Computer Engineering  
Naval Postgraduate School  
Monterey, California 93943 |
| 5.  | 1      | Professor H. Titus, Code 62Ts  
Department of Electrical and Computer Engineering  
Naval Postgraduate School  
Monterey, California 93943 |
| 6.  | 1      | Commanding Officer  
Naval Underwater Weapons Engineering Station  
Keyport, Washington 98345 |
| 7.  | 2      | Commander Joseph Jaros, USN  
Command and Control Engineering Center  
Code G520  
Reston, Virginia 22090 |