SCIENTIFIC AND TECHNICAL REPORT
FOR THE
MULTI-SENSORY TRACKING MOUNT CONTROL

THE VIEWS, OPINIONS, AND FINDINGS CONTAINED IN THIS REPORT ARE THOSE OF THE AUTHOR AND SHOULD NOT BE CONSTRUED AS AN OFFICIAL DEPARTMENT OF THE ARMY POSITION, POLICY OR DECISION, UNLESS SO DESIGNATED BY OTHER DOCUMENTATION.

SUBMITTED TO
Commander
U.S. Army White Sands Missile Range
ATTN: STEWS-ID-OA/Mr. Denman
White Sands Missile Range, NM 88002

SUBMITTED BY
SPACE DATA CORPORATION
TEMPE, ARIZONA

ORIGINAL ISSUE DATE:
8 MARCH 1988
This research develops tracking algorithms for a multi-sensor tracking mount control system. The sensors are mounted to an elevation-over-azimuth pedestal. The tracking error is detected by four independent sensors; namely, RF telemetry system, millimeter wave radar, infrared sensor, and TV video system. The quality of the signal from each sensor is available as a measure of the signal-to-noise ratio such as AGC level. It is assumed that these can be expressed as the variance of the statistical measurement noise distribution.

A major requirement of the tracking mount control system is the development of tracking filters to reduce the noise content of the multi-sensor output information. The general formulation of the Kalman filter is presented first. The precise form of the algorithm would depend on the model that is selected for the time evolution of the quantity being measured. Two such cases are considered in detail with the possibility of adapting the filter in the presence of target maneuvers. Two approaches for fusion of the multi-sensor information are presented. Recommended processing hardware and processing time estimates are presented.

A computer simulation program for the evaluation of the tracking filter performance is described next with results.

Anticipated Benefits/Potential Commercial Applications of the Research or Development

The development of advanced control algorithms, control software, and the design of a microcomputer system controller is presented which adapts a multiple sensor system to the control of a tracking mount. The design of a high-performance tracking control system using the latest state variables, adaptive and optimal control technologies is described. The control algorithms, software and parallel-processing microcomputer designs will generally be adaptable to other high-performance systems.

List a maximum of 8 Key Words that describe the Project.

Tracking System
Kalman Filters
Multi-Input State Variables
Adaptive Control
Optimal Control
Control Software
SCIENTIFIC AND TECHNICAL REPORT  
FOR THE  
MULTI-SENSORY TRACKING MOUNT CONTROL  

APPROVED BY:  
BRUCE BOLLERMANN, PRINCIPAL INVESTIGATOR  
DATE: 15 March 88
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>2. GENERAL FORMULATION OF THE DISCRETE KALMAN FILTER</td>
<td>3</td>
</tr>
<tr>
<td>3. TWO-STATE KALMAN FILTER</td>
<td>7</td>
</tr>
<tr>
<td>3.1 Filter Algorithm</td>
<td>7</td>
</tr>
<tr>
<td>3.2 Filter Initialization</td>
<td>9</td>
</tr>
<tr>
<td>4. THREE-STATE KALMAN FILTER</td>
<td>12</td>
</tr>
<tr>
<td>4.1 Filter Algorithm</td>
<td>12</td>
</tr>
<tr>
<td>4.2 Filter Initialization</td>
<td>16</td>
</tr>
<tr>
<td>5. FILTER ADAPTATION AND MANEUVER DETECTION</td>
<td>19</td>
</tr>
<tr>
<td>6. MULTI-SENSOR TRACKING ANALYSIS</td>
<td>22</td>
</tr>
<tr>
<td>6.1 General</td>
<td>22</td>
</tr>
<tr>
<td>6.2 Parallel Architecture</td>
<td>22</td>
</tr>
<tr>
<td>6.3 Centralized Architecture</td>
<td>24</td>
</tr>
<tr>
<td>6.4 Processing Hardware Design</td>
<td>25</td>
</tr>
<tr>
<td>6.4.1 General</td>
<td>25</td>
</tr>
<tr>
<td>6.4.2 Processing Time Requirements</td>
<td>29</td>
</tr>
<tr>
<td>7. COMPUTER SIMULATION</td>
<td>31</td>
</tr>
<tr>
<td>7.1 Simulation Program Description</td>
<td>31</td>
</tr>
<tr>
<td>7.2 Simulation Results</td>
<td>33</td>
</tr>
<tr>
<td>8. REFERENCES</td>
<td>38</td>
</tr>
</tbody>
</table>
LIST OF FIGURES

1. Functional Block Diagram of Multi-sensor Tracking Mount Control System 39
2. Kalman Filter Loop 40
3. Schematic Representation of Parallel Architecture 41
4. Schematic Representation of Centralized Architecture 42
5. Typical Microcontroller Block Diagram 43
6. CPU Block Diagram 44
7. Simulated Target Position Versus Time 45
8. Simulated Measurement Error Versus Time 46
9. Simulated Measurement Error Versus Time 47
10. Filtered Position Error Versus Time (2-State Filter) 48
11. Filtered Velocity Error Versus Time (2-State Filter) 49
12. Kalman Gain K_1 Versus Time (2-State Filter) 50
13. Kalman Gain K_2 Versus Time (2-State Filter) 51
14. RMS Position Estimation Error Versus Time (2-State Filter) 52
15. RMS Velocity Estimation Error Versus Time (2-State Filter) 53
16. Filtered Position Error Versus Time (2-State Filter) 54
17. Filtered Velocity Error Versus Time (3-State Filter) 55
18. Filtered Acceleration Error Versus Time (3-State Filter) 56
19. Kalman Gain K_1 Versus Time (3-State Filter) 57
20. Kalman K_2 Versus Time (3-State Filter) 58
22. RMS Position Estimation Error Versus Time (3-State Filter) 60
23. RMS Velocity Estimation Error Versus Time (3-State Filter) 61
24. RMS Acceleration Estimation Error Versus Time (3-State Filter) 62
LIST OF TABLES

6-1. FLOATING POINT COMPUTATION REQUIREMENTS 29

LIST OF APPENDICES

A. ALGORITHM SIMULATION COMPUTER PROGRAM A-1
1. INTRODUCTION

The main objective of this research is to develop the tracking algorithms for a multi-sensor tracking mount control system. The tracking mount responds to independent drive signals about the azimuth and elevation axes. The tracking error is detected by means of four independent sensors; namely, RF telemetry system, millimeter wave radar, infrared sensor, and TV video system. The quality of the signal from each of the sensor systems is available in terms of measures of the signal to noise ratio. It is assumed that these in turn can be expressed as the variance of the statistical measurement noise distribution.

A major requirement of the tracking mount control system is the development of tracking filters to reduce the noise content of the multi-sensor output information. Fixed gain tracking filter algorithms, such as the $\alpha-\beta$ and $\alpha-\beta-\gamma$ tracking filters have been used in this application in the past in conjunction with single sensor measurements. This study focuses on two aspects. First the use of Kalman filters is considered for the tracking application. Secondly, the techniques for the fusion of multi-sensory information are presented. A functional block diagram representation of the multi-sensory tracking mount control system is shown in Figure 1.

The general formulation of the Kalman filter is presented first. The precise form of the algorithm would depend on the model that is selected for the time evolution of the quantity being measured. Two such cases are considered in detail. In the first, the target acceleration is modeled as white noise and a two-state Kalman filter is developed. Next, the possibility of a correlated
target acceleration is allowed for by modeling the target acceleration as a Gauss-Markov process. This results in an algorithm involving a three-state Kalman filter. In each case, the possibility of adapting the filter in the presence of target maneuvers is discussed. This is followed by a presentation of two approaches for fusion of the multi-sensor information. Recommended processing hardware and estimated processing times are presented.

A computer simulation program for the evaluation of the tracking filter performance is described next. Results of a single run using the two-state and the three-state Kalman filters are presented.
2. GENERAL FORMULATION OF THE DISCRETE KALMAN FILTER

The general formulation of the discrete Kalman filter is briefly presented in this section.

Let the random process to be estimated be described by the linear discrete state equation

\[ \mathbf{x}(k+1) = \phi(k)\mathbf{x}(k) + \mathbf{u}(k) + \mathbf{w}(k) \quad (1) \]

The measurement of the process is assumed to occur in accordance with the linear relationship

\[ \mathbf{z}(k) = \mathbf{H}(k)\mathbf{x}(k) + \mathbf{v}(k) \quad (2) \]

Here, the various quantities are defined as:

- \( \mathbf{x}(k) \) = \( n \times 1 \) process state vector at time \( t_k \)
- \( \phi(k) \) = \( n \times n \) process state transition matrix
- \( \mathbf{u}(k) \) = \( n \times 1 \) deterministic input vector at time \( t_k \)
- \( \mathbf{w}(k) \) = \( n \times 1 \) input noise vector assumed to be a white sequence with known covariance
- \( \mathbf{z}(k) \) = \( m \times 1 \) measurement vector at time \( t_k \)
- \( \mathbf{H}(k) \) = \( m \times n \) measurement matrix describing the ideal (noiseless) connection between the measurement and state vectors at time \( t_k \)
- \( \mathbf{v}(k) \) = \( m \times 1 \) measurement noise vector assumed to be a white sequence with known covariance
The covariance matrices for \( w(k) \) and \( v(k) \) are given by,

\[
E[w(k)w^T(i)] = \begin{cases} 
Q(k), & i=k \\
0, & i \neq k 
\end{cases} \\
E[v(k)v^T(i)] = \begin{cases} 
R(k), & i=k \\
0, & i \neq k 
\end{cases}
\]

\( (3) \)

The input and measurement noise vectors are assumed to be uncorrelated,

\[
E[w(k)v^T(i)] = 0, \quad \text{for all } k \text{ and } i. \quad \quad (4)
\]

Let us assume that at some point in time \( t_k \), an initial estimate \( \hat{x}(k) \) is available based on all of our knowledge about the process prior to the arrival of the measurement \( z(k) \).

This is known as the a priori estimate. The "hat" denotes the estimate and the super minus indicates it's a priori nature. The estimation error then becomes

\[
\hat{e}(k) = x(k) - \hat{x}(k) \quad \quad (5)
\]

We further assume that the associated error covariance matrix

\[
P(k) = E[\hat{e}(k)\hat{e}^T(k)] = E[(x(k)-\hat{x}(k))(x(k)-\hat{x}(k))^T] \quad \quad (6)
\]

is also known.

With the arrival of the measurement \( z(k) \) at time \( t_k \), let the estimate be updated according to the relation

\[
\hat{x}(k) = \hat{x}(k) + K(k)(z(k) - H(k)\hat{x}(k)) \quad \quad (7)
\]

where \( \hat{x}(k) \) represents the updated (or a posteriori) estimate and \( K(k) \) denotes a nxm gain matrix.
The updated estimation error and its covariance matrix are:

\[ e(k) = \hat{x}(k) - \hat{x}(k) \]  

\[ P(k) = E[e(k)e^T(k)] = E[(\hat{x}(k) - \hat{x}(k))(\hat{x}(k) - \hat{x}(k))^T] \] \hspace{1cm} (8, 9)

Substituting Eqn. (2) into (7), and the resulting expression for \( \hat{x}(k) \) into Eqn. (9) leads to:

\[ P(k) = (I - K(k)H(k))P^-(k)(I - K(k)H(k))^T + K(k)R(k)K^T(k) \] \hspace{1cm} (10)

The individual elements along the major diagonal of \( P(k) \) represent the estimation error variance of the elements of the state vector which are being estimated. The Kalman filter selects the gain matrix \( K(k) \) such that each element on the major diagonal of \( P(k) \) is minimized. The optimum gain \( K(k) \), known as the Kalman gain, is given by:

\[ K(k) = P^-(k)H^T(k)(H(k)P^-(k)H^T(k) + R(k))^{-1} \] \hspace{1cm} (11)

The covariance matrix \( P(k) \), using the Kalman gain, can be shown to be given by

\[ P(k) = (I - K(k)H(k))P^-(k) \] \hspace{1cm} (12)

For the next time point \( t_{k+1} \), the a priori estimate is obtained from Eqn. (1) as,

\[ \hat{x}(k+1) = \phi(k)\hat{x}(k) + u(k) \] \hspace{1cm} (13)

in view of the fact that \( w(k) \) is a white sequence (with zero mean and time-wise uncorrelated).

The a priori error and associated covariance at time \( t_{k+1} \) are then obtained as,

\[ e^-(k+1) = x(k+1) - \hat{x}(k+1) = \phi(k)e(k) + w(k) \] \hspace{1cm} (14)

\[ P^-(k+1) = E[e^-(k+1)e^T(k+1)] = \phi(k)P(k)\phi^T(k) + Q(k) \] \hspace{1cm} (15)

The Kalman filter algorithm may be summarized as follows:
1. Obtain a priori estimate \( \hat{x}(k) \) and its error covariance \( P^{-}(k) \)

2. Compute Kalman gain

\[
K(k) = P^{-}(k)H(k)(H(k)P^{-}(k)H(k) + R(k))^{-1}
\]  

3. Update estimate with measurement \( z(k) \)

\[
\hat{x}(k) = \hat{x}^{-}(k) + K(k)(z(k) - H(k)\hat{x}^{-}(k))
\]  

4. Compute error covariance for updated estimate

\[
P(k) = (I - K(k)H(k))P^{-}(k)
\]  

5. Project ahead

\[
\hat{x}^{-}(k+1) = \Phi(k)x(k) + u(k)
\]
\[
P^{-}(k+1) = \Phi(k)P(k)\Phi^{T}(k) + Q(k)
\]  

6. Set \( k = k+1 \) and go to step 2.

A schematic diagram of the Kalman filter loop is shown in Figure 2.
3. TWO-STATE KALMAN FILTER

3.1 Filter Algorithm. The explicit form of the Kalman filter for the tracking mount azimuth or elevation estimation using a two-state system model is obtained in this section. To this end, define the state variables:

\[ x_1 = \theta_t - \theta_m, \text{ tracking error} \]

\[ x_2 = \dot{\theta}_t, \text{ target velocity} \]

The continuous state equations then are

\[ \begin{align*}
\dot{x}_1 &= x_2 - \omega_m(t) \\
\dot{x}_2 &= n(t) \quad \text{(21)}
\end{align*} \]

where \( \omega_m(t) \) denotes the tracking mount rate. The target acceleration \( x_2 \) is modeled as a zero mean white process \( n(t) \) with auto correlation function \( \sigma_a^2 \delta(t) \).

The discrete model corresponding to the continuous state equations may be derived as

\[ \begin{align*}
x_1(k+1) &= x_1(k) + T x_2(k) + \Delta \theta_m(k) + w_1(k) \\
x_2(k+1) &= x_2(k) + w_2(k) \quad \text{(22)}
\end{align*} \]

where \( \Delta \theta_m(k) = \theta_m(k+1) - \theta_m(k) \) represents the change in the mount encoder reading during the sample period \( T \), and is treated as a deterministic input in this application. The covariance of the random sequence \( w(k) \) is obtained as,

\[ Q(k) = \begin{bmatrix} q_{11} & q_{12} \\ q_{12} & q_{22} \end{bmatrix} = \sigma_a^2 \begin{bmatrix} T^3/3 & T^2/2 \\ T^2/2 & T \end{bmatrix} \quad \text{(23)} \]

The measurement \( z \) is given by

\[ z(k) = x_1(k) + v(k) \quad \text{(24)} \]

with \( R(k) = \sigma_v^2 \), the variance of the sensor error.
The matrices characterizing the model are thus:

\[
\phi(k) = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}, \quad H(k) = [1 \ 0]
\] (25)

Letting

\[
P^{-}(k) = \begin{bmatrix} P_{11} & P_{12} \\ P_{12} & P_{22} \end{bmatrix}, \quad P(k) = \begin{bmatrix} P_{11} & P_{12} \\ P_{12} & P_{22} \end{bmatrix}
\] (26)

and substituting the \( \phi(k) \), \( H(k) \) expressions into the general Kalman filter equations of chapter 2, the algorithm for the two state model is found to be,

1. Obtain a priori estimates \( x_1^{-}(k), x_2^{-}(k) \) and associated covariance elements \( P_{11}, P_{12}, P_{22} \)

2. Compute Kalman gains

\[
K_1 = \frac{P_{11}}{P_{11} + \sigma_v^2}
\]

\[
K_2 = \frac{P_{12}}{P_{12} + \sigma_v^2}
\] (27)

3. Update estimate with measurement \( z(k) \)

\[
x_1(k) = x_1^{-}(k) + K_1(z(k) - x_1^{-}(k))
\]

\[
x_2(k) = x_2^{-}(k) + K_2(z(k) - x_1^{-}(k))
\] (28)

4. Compute error covariance for updated estimate

\[
p_{11} = (1-K_1)p_{11}
\]

\[
p_{12} = (1-K_1)p_{12}
\]

\[
p_{22} = -K_2p_{12} + p_{22}
\] (29)

5. Project ahead

\[
x_1^{-}(k+1) = x_1(k) + T x_2(k) + \Delta \theta_m(k)
\]
\[ x_{2}^{\prime}(k+1) = x_{2}(k) \]  \hspace{1cm} (30)

\[ p_{11} = p_{11} + 2T p_{12} + T^{2} p_{22} + q_{11} \]
\[ p_{12} = p_{12} + T p_{22} + q_{12} \]
\[ p_{22} = p_{22} + q_{22} \]  \hspace{1cm} (31)

6. Set \( k = k+1 \) and go to step 2.

3.2 Filter Initialization. The filter may be initialized using the best estimates of the system state available from prior knowledge of the process and the uncertainty associated with it. Thus one might select

\[ x^{-}(0) = \begin{pmatrix} x_{10} \\ x_{20} \end{pmatrix} \]

along with

\[ p^{-}(0) = \begin{bmatrix} \sigma_{x_{10}}^{2} & 0 \\ 0 & \sigma_{x_{20}}^{2} \end{bmatrix} \]

Here, the numbers \( \sigma_{x_{10}}, \sigma_{x_{20}} \) would be small when \( x_{10}, x_{20} \) are known with a high degree of accuracy. On the other hand, the variances would tend to infinity when one has very little confidence in the choice of the initial values \( x_{10}, x_{20} \).

Alternatively, the first two measurements of the process, \( z(0) \) and \( z(1) \), may be used to initialize the filter. In this approach, the filter would start at time \( t=T \) with the initial estimates

\[ x_{1}^{-}(1) = z(1) \] \hspace{1cm} (33)
\[ x_{2}^{-}(1) = \frac{z(1) - z(0)}{T} \] \hspace{1cm} (34)

The error covariance associated with this initialization may be derived using the measurement equations

\[ z(0) = x_{1}(0) + v(0) \] \hspace{1cm} (35)
\[ z(1) = x_1(1) + v(1) \]  
\[ \text{and the state equations (22) which yield} \]
\[ x_1(1) = x_1(0) + Tx_2(0) + \Delta \theta_m(0) + w_1(0) \]  
\[ x_2(1) = x_2(0) + w_2(0) \]

From (36) and (33),
\[ x_1(1) - x_1(1) = -v(1) \]
and hence,
\[ E[(x_1(1) - x_1(1))^2] = E[v(1)^2] = \sigma_v^2 \]  
Substitute for \( x_2(0) \) from (37) into (38) to get
\[ x_2(1) = \frac{x_1(1) - x_1(0) - \Delta \theta_m(0) + w_1(0)}{T} + w_2(0) \]
Subtracting (34) from (40),
\[ x_2(1) - x_2(1) = -\frac{(x_1(1) - z(1)) - (x_1(0) - z(0)) - \Delta \theta_m(0) + w_1(0)}{T} + w_2(0) \]
Using Eqns. (35) and (36), one obtains,
\[ x_2(1) - x_2(1) = -\frac{v(1) - v(0) + \Delta \theta_m(0) + w_1(0)}{T} + w_2(0) \]
and hence,
\[ E[(x_2(1) - x_2(1))^2] = \frac{2\sigma_v^2}{T^2} + \frac{1}{T^2} q_{11} + \frac{q_{22}}{T^2} - \frac{q_{12}}{T} \]
\[ = \frac{2\sigma_v^2}{T^2} + \frac{1}{3} \sigma_a \]  
Also, we find,
\[ E[(x_1(1) - x_1(1))(x_2(1) - x_2(1))] \]
\[ = E[\frac{v(1)}{T} - \frac{v(1)}{T} + \Delta \theta_m(0) + w_1(0) - Tw_2(0)) \]
Hence, the filter initialization at the time $t=T$ becomes,

$$
\mathbf{x}^- (1) = \begin{pmatrix} 
  z(1) \\
  \frac{z(1) - z(0)}{T} 
\end{pmatrix} 
$$

$$
\mathbf{P}^- (1) = \begin{bmatrix} 
  \frac{2}{\sigma_v^2} & \frac{2}{\sigma_v^2} & \frac{\sigma_a^2}{T} \\
  \frac{2}{\sigma_v^2} & \frac{2\sigma_v^2}{T^2} & \frac{\sigma_a^2 T}{3} \\
  \frac{1}{T} & \frac{T^2}{T^2} & \frac{1}{T^3} 
\end{bmatrix} 
$$

Note that $\sigma_a = 0$ should be used if $x_1$ is known to be initially non-accelerating.
4. THREE-STATE KALMAN FILTER

4.1 Filter Algorithm. The two-state Kalman filter assumes the target acceleration to be a white noise process. In other words, the acceleration represents a timewise uncorrelated random function. In reality, however, the target acceleration would generally behave closer to a correlated process. A three-state target model following the Singer approach is developed in this section. The target acceleration is modeled as a Gauss-Markov random process and the explicit equations forming the Kalman filter are derived.

Let us define the system state variables as:

\[ \begin{align*}
    x_1 &= \theta_t - \theta_m, \text{ tracking error} \\
    x_2 &= \dot{\theta}_t, \text{ target velocity} \\
    x_3 &= \ddot{\theta}_t, \text{ target acceleration}
\end{align*} \]

The continuous state model may be written as,

\[ \begin{align*}
    \dot{x}_1 &= x_2 - \omega_m(t) \\
    \dot{x}_2 &= x_3 \\
    \dot{x}_3 &= -\alpha x_3 + \sqrt{2\alpha \sigma_a} n(t)
\end{align*} \]  \( (47) \)

where \( \omega_m(t) \) denotes the tracking mount rate. The last equation indicates that the target acceleration \( x_3(t) \) evolves as a Gauss-Markov process with correlation time \( 1/\alpha \). The process \( x_3(t) \) has the autocorrelation function \( \sigma_a e^{-2 \alpha |\tau|} \) and \( n(t) \) represents unity white noise process.

The corresponding discrete time state equations are obtained as

\[ x(k+1) = \phi(k)x(k) + \nu(k) + w(k) \]  \( (48) \)
The vector $u(k)$ is simply the deterministic input

$$u(k) = \begin{pmatrix} \Delta \theta_m(k) \\ 0 \\ 0 \end{pmatrix}$$

and is available from the encoder outputs

$$\Delta \theta_m(k) = \theta_m(k+1) - \theta_m(k)$$

The random input vector $w(k)$ denotes a zero mean white sequence with the covariance

$$Q(k) = \begin{bmatrix} q_{11} & q_{12} & q_{13} \\ q_{12} & q_{22} & q_{23} \\ q_{13} & q_{23} & q_{33} \end{bmatrix}$$

where

$$q_{11} = \frac{\sigma_a^2}{\alpha^4} \left[ 1 - e^{-2\alpha T} + 2\alpha T + \frac{3}{2} \frac{\alpha^3}{T} - 2\alpha^2 \frac{\alpha^3}{T^2} - 4\alpha T e^{-\alpha T} \right]$$

$$q_{12} = \frac{\sigma_a^2}{\alpha^3} \left[ e^{-2\alpha T} + 1 - 2e^{-\alpha T} + 2\alpha T e^{-\alpha T} - 2\alpha T + \frac{2}{2} \frac{\alpha^2}{T} \right]$$

$$q_{13} = \frac{\sigma_a^2}{\alpha^2} \left[ 1 - e^{-2\alpha T} - 2\alpha T e^{-\alpha T} \right]$$

$$q_{21} = q_{12}$$

$$q_{22} = \frac{\sigma_a^2}{\alpha^4} \left[ 1 - 2e^{-\alpha T} + 2\alpha T e^{-\alpha T} - 2\alpha T + \frac{2}{2} \frac{\alpha^2}{T} \right]$$

$$q_{23} = \frac{\sigma_a^2}{\alpha^3} \left[ e^{-2\alpha T} - 2\alpha T e^{-\alpha T} \right]$$

$$q_{31} = q_{13}$$

$$q_{32} = q_{23}$$

$$q_{33} = \frac{\sigma_a^2}{\alpha^2} \left[ 1 - e^{-\alpha T} - 2\alpha T e^{-2\alpha T} \right]$$
\[
q_{22} = \frac{\sigma_a^2}{2} \left[ 4 e^{-\alpha T} - 3 - e^{-2\alpha T} + 2\alpha T \right]
\]
\[
q_{23} = \frac{\sigma_a^2}{\alpha} \left[ e^{-2\alpha T} + 1 - e^{-\alpha T} \right]
\]
\[
q_{33} = \sigma_a^2 \left[ 1 - e^{-2\alpha T} \right]
\]

The measurement is given by
\[
z(k) = x_1(k) + v(k)
\]

with \( R(k) = \sigma_v^2 \), the variance of the sensor error. The \( H(k) \) matrix simply equals \((1 \ 0 \ 0)\) since only measurements of \( x_1(k) \) are available. Let the a priori and a posteriori estimation error covariances be denoted as
\[
P^- = \begin{bmatrix} p_{11} & p_{12} & p_{13} \\ p_{12} & p_{22} & p_{23} \\ p_{13} & p_{23} & p_{33} \end{bmatrix},
\]

\[
P(k) = \begin{bmatrix} P_{11} & P_{12} & P_{13} \\ P_{12} & P_{22} & P_{23} \\ P_{13} & P_{23} & P_{33} \end{bmatrix}
\]

The various quantities describing the discrete three-state model may now be substituted in the general Kalman filter equations of chapter 2. The explicit filter algorithm for the three-state Kalman filter then becomes:

1. Obtain initial state estimates \( \hat{x}_1(-k), \hat{x}_2(-k), \hat{x}_3(-k) \) and the associated estimation error covariance elements \( p_{11}, p_{12}, p_{13}, p_{22}, p_{23}, p_{33} \).

2. Compute Kalman gains
\[
K_1 = \frac{p_{11}}{p_{11} + \sigma_v^2}
\]
\[ K_2 = \frac{\bar{p}_{12}}{\bar{p}_{11} + \sigma_v^2} \]

\[ K_3 = \frac{\bar{p}_{13}}{\bar{p}_{11} + \sigma_v^2} \]  

(57)

3. Update estimate with measurement \( z(k) \)

\[ x_1(k) = \hat{x}_1(k) + K_1(z(k) - \hat{x}_1(k)) \]

\[ x_2(k) = \hat{x}_2(k) + K_2(z(k) - \hat{x}_1(k)) \]

\[ x_3(k) = \hat{x}_3(k) + K_3(z(k) - \hat{x}_1(k)) \]  

(58)

4. Compute error covariance for updated estimate

\[ P_{11} = (1 - K_1)\bar{p}_{11} \]

\[ P_{12} = (1 - K_1)\bar{p}_{12} \]

\[ P_{13} = (1 - K_1)\bar{p}_{13} \]

\[ P_{22} = -K_2\bar{p}_{12} + \bar{p}_{22} \]

\[ P_{23} = -K_2\bar{p}_{13} + \bar{p}_{23} \]

\[ P_{33} = -K_3\bar{p}_{13} + \bar{p}_{33} \]  

(59)
5. Project ahead

\[ x_1^{-(k+1)} = x_1(k) + T x_2(k) + \phi_{13} x_3(k) + \Delta \theta_m(k) \]
\[ x_2^{-(k+1)} = x_2(k) + \phi_{23} x_3(k) \]
\[ x_3^{-(k+1)} = \phi_{33} x_3(k) \]  

\[ P_{11} = P_{11} + 2 T P_{12} + 2 \phi_{13} P_{13} + T^2 P_{22} + 2 T \phi_{13} P_{33} + q_{11} \]
\[ P_{12} = P_{12} + \phi_{23} P_{13} + T P_{22} + (\phi_{13} + T \phi_{23}) P_{23} + \phi_{13} \phi_{23} P_{33} + q_{12} \]
\[ P_{13} = \phi_{33} (P_{13} + T P_{23} + \phi_{13} P_{33}) + q_{13} \]
\[ P_{22} = P_{22} + 2 \phi_{23} P_{23} + \phi_{23} P_{33} + q_{22} \]
\[ P_{23} = \phi_{33} (P_{23} + \phi_{23} P_{33}) + q_{23} \]
\[ P_{33} = \phi_{33} P_{33} + q_{33} \]  

\[ \text{(60)} \]

6. Set \( k = k + 1 \) and go to step 2.

4.2 Filter Initialization. The three-state Kalman filter may be initialized using the best available initial position, velocity and acceleration states, and the associated error covariance. One might use the initialization

\[ \bar{x}^{-}(0) = \begin{pmatrix} x_{10} \\ x_{20} \\ 0 \end{pmatrix} \]

along with

\[ P^{-}(0) = \begin{bmatrix} \sigma_{x_{10}}^2 & 0 & 0 \\ 0 & \sigma_{x_{20}}^2 & 0 \\ 0 & 0 & \sigma_a^2 \end{bmatrix} \]

\[ \text{(62)} \]

The zero initialization for the acceleration is appropriate because it is modeled as a Gauss-Markov process which is a zero mean process in the ensemble sense.
Alternatively, the first two measurements, \( z(0) \) and \( z(1) \), may be utilized to initiate the filter at time \( t=T \). Using the measurement equations

\[
\begin{align*}
  z(0) &= x_1(0) + v(0) \\
  z(1) &= x_1(1) + v(1)
\end{align*}
\]

and the state equations (48) to yield

\[
\begin{align*}
  x_1(1) &= x_1(0) + T x_2(0) + p_{13} x_3(0) + w_1(0) \\
  x_2(1) &= x_2(0) + p_{23} x_3(0) + w_2(0) \\
  x_3(1) &= p_{33} x_3(0) + w_3(0)
\end{align*}
\]

the initial error covariance may be derived following the approach detailed in the two-state filter case. The initial state estimate is thus

\[
\begin{pmatrix}
  z(1) \\
  z(1) - z(0) / T \\
  0
\end{pmatrix}
\]

and the elements of the symmetric \( P^{-1}(1) \) matrix are given by

\[
\begin{align*}
  p_{11}^{-1}(1) &= \sigma_v^2 \\
  p_{12}^{-1}(1) &= \frac{\sigma_v^2}{T} \\
  p_{13}^{-1}(1) &= 0 \\
  p_{22}^{-1}(1) &= \frac{2\sigma_v^2}{T} + \frac{\sigma_a^2}{\alpha T} \left[ 2 - \alpha T + \left( \alpha T - 2e^{-\alpha T} - 2\alpha Te^{-\alpha T} \right) \right] \\
  p_{23}^{-1}(1) &= \frac{\sigma_a^2}{\alpha^2 T} \left[ e^{-\alpha T} + \alpha T - 1 \right] \\
  p_{33}^{-1}(1) &= \sigma_a^2
\end{align*}
\]
Note that, $\sigma_a = \dot{\omega}$ should be used if the system is known to be non-accelerating initially.
5. FILTER ADAPTATION AND MANEUVER DETECTION

The Kalman filter development presented in the previous chapters assumes a knowledge of the target dynamic characteristics as well as the measurement uncertainty. However, in practice, the information concerning the target acceleration variance $Q_a$ may be limited. This is particularly true in case of targets that are known to exhibit maneuvering dynamics.

The target dynamics uncertainty enters the filter algorithm through the matrix $Q(k)$. Lacking a precise value of $Q(k)$, it may be viewed as a parameter influencing the filter design. Increasing the elements of $Q(k)$ results in larger values of the a priori estimation error covariance $P^-(k)$ as per Eqn (20). When the elements of $Q(k)$ (and hence those of $P^-(k)$) are large compared to those of $R(k)$, the Kalman gains $K(k)$ assume larger values as indicated by Eqn (16). The filter thus puts a relatively greater weight on the new measurement than on the a priori estimate. Relatively more measurement noise is then present in the filtered estimate $\hat{x}(k)$. The algorithm therefore represents a wideband filter.

Conversely, when the elements of $Q(k)$ are relatively small compared to those of $R(k)$, the Kalman gains $K(k)$ become small. The algorithm then puts more weight on the a priori estimate $\hat{x}^-(k)$ compared to the new measurement in forming the filtered estimate $\hat{x}(k)$. In other words, it relies more on the past history and the state projection via the state transition matrix $\Phi(k)$. Therefore, less noise is present in the filtered estimate and the algorithm represents a narrowband filter.
In view of the above a narrowband filter design would appear more desirable. However, if the true uncertainty in the target dynamics exceeds the $Q(k)$ value used in the filter design, the narrowband characteristic (small $Q(k)$ to $R(k)$ ratio and hence small $K(k)$) would cause too much of the signal contained in the observations $z(k)$ to be rejected. The filtered estimate may then fail to follow the true motion and may even diverge from it. Consequently, a judicious selection of the filter bandwidth is clearly indicated.

The effect of a mismatch in modeling has a similar effect. When the state transition matrix $\phi(k)$ assumed in the filter design deviates from the true model of the system, the projected estimate $\tilde{x}^-(k)$ would be in greater error than indicated by its covariance matrix $P^-(k)$. Again, the possibility of divergence due to this reason may be reduced by incorporating an increased $Q(k)$ leading to a wideband filter.

The ability to control the filter bandwidth by varying the $Q(k)$ to $R(k)$ ratio may be used to track maneuvering targets. Suppose a good narrowband filter has been designed for a known type of target motion. If the target makes a sudden maneuver, the narrowband filter may be too slow to react to it because of the less weightage on the new observation and the track may be lost. One way of correcting the situation would be to increase the filter bandwidth as soon as the maneuver occurs by increasing $Q(k)$. When the maneuver stops, $Q(k)$ may be reduced to realize a narrowband filter again and achieve better filtering. This would represent an adaptive technique for tracking maneuvering targets. However, a maneuver detection method is required.

A maneuver may be detected by monitoring the innovations process
\[ \mathbf{v}(k) = \mathbf{z}(k) - \mathbf{H}(k)\mathbf{x}(k) \]  

(67)

The innovations process represents a zero mean process with the covariance

\[ S(k) = \mathbf{H}(k)\mathbf{P}^{-}(k)\mathbf{H}^T(k) + \mathbf{R}(k) \]  

(68)

Assuming a Gaussian distribution for the components of \( \mathbf{v}(k) \), the probability that each component \( v_i \) lies in the interval \( \pm c \sqrt{S(i,i)} \) where \( c \) is a positive constant is well known. A simple maneuver detection scheme may thus be formed by checking each \( v_i(k) \) against this interval. Whenever any of the \( v_i(k) \) falls outside the range a maneuver may be declared.

An alternative approach is to consider the normalized random variable \( \rho(k) \) defined as

\[ \rho(k) = \mathbf{v}^T(k)\mathbf{S}^{-1}(k)\mathbf{v}(k) \]  

(69)

It can be shown that \( \rho(k) \) has a chi-squared distribution with \( m \) degrees of freedom where \( m \) is the dimension of the \( \mathbf{v}(k) \) vector. Hence,

\[ E[\rho(k)] = m \]  

(70)

The criterion \( \rho(k) > cm \), where \( c > 1 \) is a constant, may then be used to detect a maneuver. Or, a fading memory average computed from

\[ g(k) = \gamma g(k-1) + \rho(k), 0 \leq \gamma < 1 \]  

(71)

may be used. Note that

\[ \lim_{k \to \infty} E[g(k)] = \frac{m}{1-\gamma} \]  

(72)

Thus, whenever \( g(k) \) exceeds \( c \frac{m}{1-\gamma} \), a maneuver may be declared.

Once a maneuver is detected using either of the above techniques, the \( Q(k) \) matrix may be increased suitably. This would widen the filter bandwidth enabling it to track the maneuvering target.
6. MULTI-SENSOR TRACKING ANALYSIS

6.1 General. The earlier sections addressed the development of Kalman filter algorithms assuming a single sensor providing noisy measurement of the position misalignment between the target and the tracking mount. When a single target is to be tracked using a set of multiple sensors providing noisy measurements of the tracking error, the sensor outputs may be processed using either a parallel or a centralized architecture.

6.2 Parallel Architecture. The parallel processing architecture is schematically shown in Figure 3. Each sensor output $z^i$ is processed by a Kalman filter which provides the optimal state estimate $\hat{x}^i$ and the associated estimation error covariance matrix $P^i$, $i = 1,4$. The individual Kalman filter estimates are then optimally combined to form the single state estimate $\hat{x}$ and its error covariance matrix $P$.

The track fusion relationships may be developed readily using the minimum expected Mean Square Error (MSE) criterion. To simplify the approach, let us consider the scalar problem of optimally combining the four Kalman filter estimates of the tracking error. Further, we introduce the simplified notion:

\[ y = \text{true value of } x_1 \]
\[ y_i = x_1 \quad , \quad i = 1,4 \]
\[ \sigma_i = p_{ii} \]
\[ y_c = x_1 \quad , \quad \text{the combined estimate of } x_1 \quad (73) \]
Let the combined estimate be expressed as,

\[ y_c = c_1 y_1 + c_2 y_2 + c_3 y_3 + c_4 y_4 \]  \hspace{1cm} (74)

The estimation error then becomes,

\[ e_c = y - y_c = y - c_1 y_1 - c_2 y_2 - c_3 y_3 - c_4 y_4 \]
\[ = c_1(y - y_1) + c_2(y - y_2) + c_3(y - y_3) + c_4(y - y_4) \]
\[ + (1 - c_1 - c_2 - c_3 - c_4)y \]
\[ = c_1 e_1 + c_2 e_2 + c_3 e_3 + c_4 e_4 \]
\[ + (1 - c_1 - c_2 - c_3 - c_4)y \]  \hspace{1cm} (75)

For an unbiased estimate, one requires

\[ E[e_c] = 0 \]  \hspace{1cm} (76)

Hence,

\[ c_1 + c_2 + c_3 + c_4 - 1 = 0 \]  \hspace{1cm} (77)

which leads to

\[ e_c = c_1 e_1 + c_2 e_2 + c_3 e_3 + c_4 e_4 \]  \hspace{1cm} (78)

\[ E[e_c^2] = c_1 \sigma_1^2 + c_2 \sigma_2^2 + c_3 \sigma_3^2 + c_4 \sigma_4^2 \]  \hspace{1cm} (79)

Minimization of \( E[e_c^2] \) subject to the constraint (77) results in

\[ c_i = \frac{1}{\sigma_i^2} \sum_{i=1}^{2} \frac{1}{\sigma_i^2} \]  \hspace{1cm} (80)

The optimal combined tracking error estimate becomes

\[ y_c = \frac{1}{\sum_{i=1}^{2} \frac{1}{\sigma_i^2}} \left( \sum_{i=1}^{2} \frac{1}{\sigma_i^2} y_i \right) \]  \hspace{1cm} (81)
or

\[ x_1(c) = \frac{1}{\sum P_{11}(i)} \left( \sum \frac{1}{P_{11}(i)} x_1(i) \right) \quad (82) \]

Substituting Eqn (74) into Eqn (73), the variance of the combined tracking position estimation error becomes

\[ \sigma_c^2 = \frac{1}{\sum \frac{1}{\sigma_i^2}} \quad (83) \]

6.3 Centralized Architecture. In the centralized architecture, the measurements from all the sensors are first combined optimally to produce a single equivalent measure of the track misalignment. A single Kalman filter then processes the combined measurement data to produce an optimal estimate of the system state. Figure 4 shows a schematic diagram illustrating the centralized architecture.

To obtain the optimal sensor data fusion relation, express the desired combined measurement \( z(c) \) as a linear combination of the individual sensor measurements of \( x_1 \)

\[ z(c) = c_1z(1) + c_2z(2) + c_3z(3) + c_4z(4) \quad (84) \]

The equivalent measurement equation used by the Kalman filter may be written as

\[ z(c) = x_1 + v(c) \quad (85) \]
The error in the equivalent measurement is given by

\[ v(c) = z(c) - x_1 \]

\[ = c_1(z(1) - x_1) + c_2(z(2) - x_1) + c_3(z(3) - x_1) \]

\[ + c_4(z(4) - x_1) + (c_1 + c_2 + c_3 + c_4 - 1)x_1 \]

\[ = c_1v(1) + c_2v(2) + c_3v(3) + c_4v(4) \]

\[ + (c_1 + c_2 + c_3 + c_4 - 1)x_1 \]

\[ \text{(86)} \]

As done earlier in the parallel architecture case, the requirement of an unbiased estimate and the minimization of the error variance leads to

\[ c_i = \frac{1}{\sigma_{v_i}^2} \cdot \frac{1}{\sum \frac{1}{\sigma_{v_i}^2}} \]

\[ \text{which gives the optimum data fusion relations} \]

\[ z(c) = \frac{1}{\sum \frac{1}{\sigma_{v_i}^2}} \left( \sum \frac{1}{\sigma_{v_i}^2} z(i) \right) \]

\[ \sigma_{v_c}^2 = \frac{1}{\sum \frac{1}{\sigma_{v_i}^2}} \]

\[ \text{The equivalent measurement and its error variance thus obtained are processed by} \]

\[ \text{the single Kalman filter to produce the optimal state estimate.} \]

6.4 Processing Hardware Design

6.4.1 General. The microcontroller accepts multiple sensors input information and processes the data based on the above control algorithms. The results of the
processing are output in the form of control command signals capable of driving a tracking and pointing system.

The microcontroller is composed of several subsystems. The subsystem functions provide for user interface, shared RAM, sensor/signal input and processing, and output control. Figure 5 shows a block diagram of a typical microcontroller and the various subsystems and functions.

The user interface consists of a keyboard/terminal for user interaction. The user interface provides the means by which the operator can interact with the microcontroller using a high level programming language to vary the control parameters of the pointing system. The level of interaction will allow the user to develop and implement the necessary algorithms.

The Central processing Unit (CPU) is a general purpose subsystem used in the controller to implement the Kalman filter and optimal combining algorithms. A block diagram of the CPU, Motorola Part Number MVME133-1, is presented in Figure 6. The main features of the CPU system are microprocessors, cache memory, memory management, timing, interrupt handling, and serial I/O. The CPU subsystem combines the processing power of the Motorola 32-bit MC68020 Microprocessor with the speed-enhancement properties of a 16Kb Cache Memory Accelerator. The Cache Accelerator, a small, fast memory system, compensates for the typical speed mismatch between a very fast CPU and its relatively slow associated dynamic RAM main memory system. It does so by concurrently storing the data most recently stored in main-memory locations. These data can subsequently be obtained by fast
accesses of "cached" locations rather than by much slower accesses of main memory.

To accommodate for the fact that the cache memory normally has much less capacity than the main memory system, little-used accesses stored in the cache memory are routinely replaced with more active ones, based on the bus traffic between the Microprocessor Module and the bus and memory interfaces. This assures maximum utilization of the cache. The microprocessor operates at a fixed frequency of 16 MHz. To further improve the processing speed and capabilities of the CPU system, the microprocessor is supported by a Motorola MC68881 floating point coprocessor. Commercial grade CPU systems are available which operate with clock frequencies up to 24 MHz. The MC68020 can interface and control as many as four 68881 coprocessors.

The memory management function implements demand-paged virtual memory operations. It provides the required logical to physical address translation by performing searching of translation tables in main memory.

The onboard programmable timer provides three independent cascadable 16-bit counters with interrupt capability.

The interrupt handler allows interrupts to the onboard CPU from up to 20 sources. The interrupt handler preprocesses interrupt sources into three groups of seven interrupts corresponding to the seven possible MC68020 interrupt levels. The groups are labeled Group 1, Group 2, and Group 3. The interrupt service priority is determined by the interrupt level and the group number. Interrupts with
Different interrupt levels are processed according to the standard interrupt processing discipline. Interrupts within an interrupt level are processed according to the group number.

Group 1 is reserved for system bus interrupts. The interrupt handler processes Group 2 and 3 interrupts differently from Group 1 interrupts. If the interrupt being acknowledged is a Group 2 or 3 interrupt, the interrupt handler fetches the appropriate exception vector number from PROM and sends it to the CPU via the local bus. If the interrupt being acknowledged is a Group 1 interrupt, the exception vector number is fetched from the system bus where it was placed by the interrupting device.

The serial I/O ports are accessible via a 50-pin flat ribbon cable connector at the top of the board. Signal levels at this connector correspond to TTL specifications, but can be transformed to RS-232-C levels by means of a separate distribution board and cable assembly. The ports have full- and half-duplex compatibility with programmable baud rates. They are suitable for synchronous or asynchronous operation, with 5 to 8 bits per character, plus parity.

The sensor/signal input and processing consists of parallel digital input ports, analog-to-digital converters, and television frame signal processors. It is expected that the existing hardware will be usable in the new controller. These inputs are made available for processing in the new CPU assembly.
The output control subsystem provides the interface between the processed data of the signal input/processing subsystem or the user command data from the user interface subsystem to drive the control command signals for pointing control.

6.4.2 Processing Time Requirements. The estimated processing time to compute the state-variable filter and combining algorithms are presented. Worst case analysis is presented which ignores the speed increases available through the inherent pipelining capabilities of the suggested CPU with cache RAM and math coprocessors. Also, a 16 MHz clock is assumed.

Table 6-1 presents the amount of double precision floating point calculations required for filtering and combining one set of data. The two-state Kalman filter computation requirements are based on the algorithm presented in Section 3.1. Section 4.1 describes the three-state Kalman filter. The optimal combining algorithms are presented in Sections 6.2 and 6.3.

<table>
<thead>
<tr>
<th>ALGORITHM</th>
<th>ADD</th>
<th>SUBTRACT</th>
<th>MULTIPLY</th>
<th>DIVIDE</th>
<th>PROCESSING TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-STATE FILTER</td>
<td>13</td>
<td>5</td>
<td>10</td>
<td>2</td>
<td>274.0 μSEC</td>
</tr>
<tr>
<td>3-STATE FILTER</td>
<td>34</td>
<td>7</td>
<td>37</td>
<td>3</td>
<td>744.5 μSEC</td>
</tr>
<tr>
<td>PARALLEL COMBINING</td>
<td>6</td>
<td>0</td>
<td>5</td>
<td>5</td>
<td>158.5 μSEC</td>
</tr>
<tr>
<td>CENTRAL COMBINING</td>
<td>6</td>
<td>0</td>
<td>5</td>
<td>5</td>
<td>158.5 μSEC</td>
</tr>
</tbody>
</table>

The estimated time for performing each math computation listed in Table 6-1 is given below. This assumes double-precision, floating point calculations using a 68020 CPU, 68881 math coprocessor, a 16 MHz clock, and a 32-bit wide data bus.

- FADD: 136 CLOCK CYCLES, 8.5 μSEC
- FSUB: 136 CLOCK CYCLES, 8.5 μSEC
- FMUL: 156 CLOCK CYCLES, 9.75 μSEC
- FDIV: 188 CLOCK CYCLES, 11.75 μSEC
Methods available to easily improve the required processing time includes using faster system clocks which would provide up to a 33% increase and using multiple math coprocessors. Although the 68020 can physically interface with eight math coprocessors, the factory states that four coprocessors per CPU is a practical limit. The four coprocessors would improve the processing time by about 50%.
7. COMPUTER SIMULATION

7.1 Simulation Program Description. A computer program has been developed to evaluate the filtering techniques presented in the study. The program simulates the target motion, measurement noise and the filtering process. Either the two-state or the three-state Kalman filter algorithm may be simulated.

The target motion is assumed to be governed by the state equations:

\[
\begin{align*}
0 & : x_1 = x_2 \\ 0 & : x_2 = x_3 + a(t) \\ 0 & : x_3 = -\beta x_3 + \sqrt{2 \beta \sigma_{a_t}^2} n(t)
\end{align*}
\]  \hspace{1cm} (90)

where \( x_1 \) = target position, \( x_2 \) = target velocity. The target acceleration is composed of a deterministic part \( a(t) \) and a random component \( x_3 \). The random component is modeled as a Gauss-Markov process with the autocorrelation function \( \sigma_{a_\tau}^2 - \beta |\tau| \). The function \( n(t) \) represents unity amplitude white noise process.

The discrete equivalent of these continuous target state equations used by the program is

\[
\chi(k+1) = \phi(k)\chi(k) + u(k) + w(k)
\]  \hspace{1cm} (91)

Here \( u(k) \) denotes the deterministic input vector

\[
u(k) = \begin{pmatrix} 1 \\ -T^2 \\ 2 \\ T \end{pmatrix} a(k)
\]  \hspace{1cm} (92)
The state transition matrix $\phi(k)$ and the zero-mean white sequence $w(k)$ are described by equations of section 4.1 with $\alpha$ replaced by $\beta$, and $\sigma_a$ replaced by $\sigma_at$.

The random numbers forming the sequence $w(k)$ are generated using a random number subroutine. The subroutine generates random numbers $x$ uniformly distributed between 0 and 1. Zero mean random numbers $y$ uniformly distributed in the range $-A$ to $+A$ are then obtained through the transformation $y = A(2x - 1)$. The numbers $y$ have the variance $\sigma_y^2 = \frac{A^2}{3}$. In the present program, the $w(k)$ inputs are zero-mean sequences with specified covariance $Q(i,j)$. The program thus uses $A_i = \sqrt{3Q_{ii}}$ to simulate the appropriate random numbers representing the elements of the $w(k)$ vector.

The measurement is modeled by adding a random noise sequence $v(k)$ to the true target position $x_1(k)$ obtained as the solution of the target state equations. The measurement noise is assumed to be a uniformly distributed zero-mean sequence with variance $\sigma_v^2$. This is achieved by letting $A = \sqrt{3\sigma_v}$ in the random number generator computation. The program permits the input of piecewise constant values of the measurement variance $\sigma_v^2$.

Either the two-state or the three-state Kalman filter algorithm may be simulated. The filter may be initialized either using a priori state estimates and their associated error covariances, or using the first two measurements to generate these initial quantities.
The simulation also provides the option of evaluating the adaptive filtering approach using the innovations process. This enables the filter to achieve a wider bandwidth whenever a maneuver is detected.

All quantities of interest concerning the target motion, measurement and filtering errors, error covariance, Kalman gains etc. can be printed out. The mean and mean square values of the measurement errors simulated during a run are printed out to provide a check on the random number generation. The mean and mean square values of the filtered position and velocity errors during a run are printed out to provide a measure of the effectiveness of the filter.

The program is presently limited to the input of a pulse-type acceleration profile for the deterministic component \( a(t) \) of the target acceleration. The measurement error variance \( \sigma_v^2 \) also is restricted to five constant values during one run. These restrictions, however, are minor in nature and may be removed quite readily. The program currently performs a single run. On the other hand, Monte Carlo methods should be used to fully establish the performance of the filter design under consideration. It would therefore be desirable to enhance the capability of the program to perform multiple runs and output the ensemble statistics.

7.2 Simulation Results. The results of a few typical computer simulation runs are presented in this section. A comment concerning the units of the various quantities of interest would be appropriate. Any time unit, such as second, minute, may be associated with the time scale values. Similarly, any position
unit such as degrees, radians, feet etc. may be associated with the position scale. The units of the various quantities of interest then become:

- measurement: [position unit]
- position: [position unit]
- velocity: [position unit/(time unit)]
- acceleration: [position unit/(time unit)^2]
- Kalman gain K1: [dimensionless]
- Kalman gain K2: [1/(time unit)]
- Kalman gain K3: [1/(time unit)^2]

The selection of the filter parameters requires some knowledge of the process, i.e., the target dynamics. Suppose it is known that the target may have any acceleration with an equal probability in the range of -60 to +60. This yields the acceleration variance \( \sigma_a^2 = 1200 \). Let the best estimate of the acceleration correlation time be about 100 time units, i.e., \( \alpha = 0.01 \). These parameters are used to define the two- and three-state Kalman filters for the simulation.

The filters may be initialized using either of the two approaches discussed earlier. Here we assume an initial position value of 3.0 with an associated variance of 25.0 representing the accuracy of the sensor system. Assuming an a priori knowledge that the initial velocity may be any value in the range -60 to +60 with uniform probability yields a variance of 1200. Using the mean values as the best initial estimates, the three-state filter is thus initialized as

\[
\mathbf{x}^-(0) = \begin{pmatrix} 3.0 \\ 0.0 \\ 0.0 \end{pmatrix}
\]
The initialization used for the two-state filter is

\[ x(0) = \begin{bmatrix} 3.0 \\ 0.0 \end{bmatrix} \]

\[ P^{-}(0) = \begin{bmatrix} 25 & 0 \\ 0 & 1200 \end{bmatrix} \]

Unknown to the Kalman filter, we consider a target motion consisting of a constant acceleration of 40. Superposed on this is a random acceleration uniformly distributed between -8.66 to 8.66 which corresponds to a variance of 25. The random acceleration is assumed correlated with \( R = 1.0 \).

The measurements are assumed to be of varying accuracy over the time period of interest. Uniformly distributed measurement errors with variances of

\[ \sigma_v = \begin{cases} 25 & (0 \leq t < 2) \\ 1 & (2 \leq t < 4) \\ 49 & (4 \leq t < 6) \\ 4 & (6 \leq t < 8) \\ 16 & (8 \leq t < 10) \end{cases} \]

were arbitrarily selected for the simulation. Note that this corresponds to the measurement errors being uniformly distributed in the range \(-\sqrt{3} \sigma_v\) to \(+\sqrt{3} \sigma_v\).

The sampling rate was assumed to be 100 samples per unit time, i.e., \( T = 0.01 \) time units.
Figure 7 shows a plot of the true target position $x_1$ as a function of time. The measurement error at each sampling point is shown in Figure 8. Joining the successive points, the plot in Figure 9 shows a better perspective of the measurement errors as the data arrive sequentially. The errors generated by the program confirm to the specified variation of the measurement accuracy.

The output of the two-state filter is presented first. Figure 10 shows the variation of the filtered position error as a function of time. A comparison with the measurement error time history clearly shows the effectiveness of the Kalman filter in reducing the effect of measurement noise. Over the time period of 10 units, the rms position error is found to be 1.4210 in contrast to the rms measurement error of 4.3589. The filtered velocity output is shown in Figure 11 and the rms velocity error over 10 time units is 11.4643. Figure 12 shows the variation of the Kalman position gain $K_1$. It is apparent that the gain automatically adjusts in accordance with the quality of the measurements. For example, at $t=2$, the measurement quality changes from a variance of $\sigma_y^2 = 25$ to 1. The filter assumes a much bigger gain $K_1$ and thus puts a substantially higher weight on the new data in comparison to the prediction from the state model. This follows a transient period during which the model predictions successively improve and the weightage on the new observations reduces. Finally, a steady state is arrived when $K_1$ assumes a constant value. The variation of the velocity gain $K_2$ showing similar behavior is presented in Figure 13.

A measure of the quality of the filter design may be obtained from the diagonal elements of the estimation error covariance matrix $P(t)$. Plots of the rms
position and velocity estimation error, $\sqrt{p_{11}}$ and $\sqrt{p_{22}}$, respectively, are shown in Figures 14 and 15. These represent the theoretical values of the filtering error statistics and are meaningful in an ensemble sense. Furthermore, these correspond to the ideal situation when the system state model used in the filter and the truth model are identical. Again, the effect of the measurement quality variations on these statistics is apparent.

The results obtained with the three-state filter are presented next. Figure 16 shows the position filtering error. It shows an rms value of 0.8320 over the 10 time units of simulation. The velocity error, with an rms value of 7.6585 over the simulation period, is shown in Figure 17. The three-state filter estimates the acceleration with the error presented in Figure 18. The Kalman gains $K_1$, $K_2$ and $K_3$ are shown in Figure 19, 20, and 21, respectively. The theoretical position, velocity and acceleration error statistics associated with this filter design are presented in Figures 22, 23, and 24, respectively.

From the above, it would appear that the three-state filter performs better than the two-state filter. However, due to the statistical nature of the problem, a single simulation suggests nothing of value regarding a comparison between the two situations. In fact, the performance of each filter is highly dependent on the nature of the target motion and the tuning of the filter via the selection of $\sigma_a^2$, the correlation characteristic $\alpha$ and the sampling interval $T$. As a general feature, however, reducing the sampling interval would improve the filtering performance in both cases.
8. REFERENCES


Figure 1. Functional Block Diagram Of Multi-sensor Tracking Mount Control System

\[ \theta_i = \text{TGAR \ POSITION (AZIMUTH OR ELEVATION) } \]

\[ \theta_m = \text{MOUNT POSITION (AZIMUTH OR ELEVATION) } \]

\[ \theta_e = \text{TRACKING ERROR (AZIMUTH OR ELEVATION) } \]
Figure 2. Kalman Filter Loop

1. **Initialize**
   - $\hat{x}(0)$
   - $P(0)$

2. **Compute Kalman Gain**
   - Equation (16)

3. **Update Estimate with Measurement**
   - Equation (17)

4. **Compute Error Covariance for Updated Estimate**
   - Equation (18)

5. **Project Ahead**
   - Equation (19) & (20)

6. **Measurements**
   - $z(k)$

7. **Measurement**
   - $z(k)$, $R(k)$

8. **Update with Measurement**
   - $K(k)$

9. **Error Covariance Update**
   - $P(k)$

10. **Error Covariance Update**
    - $P(k+1)$

11. **Final State Estimate**
    - $\hat{x}(k+1)$
Figure 3. Schematic Representation Of Parallel Architecture
Figure 4: Schematic Representation of Centralized Architecture
Figure 5. Typical Microcontroller Block Diagram
Figure 6. CPU Block Diagram
Figure 8. Simulated Measurement Error Versus Time
Figure 10. Filtered Position Error Versus Time (2-State Filter)
Figure 11. Filter Velocity Error Versus Time (2-State Filter)

FILTERED VELOCITY ERROR
Figure 12. Kalman Gain $K_1$ Versus Time (2-State Filter)
Figure 13. Kalman Gain K₂ Versus Time (2-State Filter)
Figure 16. Filtered Position Error Versus Time (3-State Filter)
Figure 17. Filtered velocity error versus time (3-State Filter)
Figure 18. Filtered Acceleration Error Versus Time (3-State Filter)
Figure 21. Kalman Gain K3 Versus Time (3-State Filter)
Figure 22. RMS Position Estimation Error Versus Time (3-State Filter)
Figure 23. RMS Velocity Estimation Error Versus Time (3-State Filter)
Figure 24. RMS Acceleration Estimation Error Versus Time (3-State Filter)
APPENDIX A

ALGORITHM SIMULATION COMPUTER PROGRAM
The FORTRAN listing of the computer simulation program are enclosed in this appendix. The program input and a sample output file are also given. The various input quantities are:

\[
T = \text{Sampling period}
\]

\[
\text{ALPHA} = \alpha
\]

\[
\text{SIA} = \sigma_a
\]

\[
\text{SIM} = \sigma_v
\]

\[
\text{TMAX} = \text{Value of time for program termination}
\]

\[
\text{SIAM} = \sigma_{at}
\]

\[
\text{BETA} = \beta
\]

\[
\text{XINL} = \text{Initial target position}
\]

\[
\text{VINL} = \text{Initial target velocity}
\]

\[
\text{AINRAND} = \text{Initial value of random target acceleration}
\]

\[
\text{AMEAN1} = \text{Constant target acceleration outside the range } T_1 < t < T_2
\]

\[
\text{AMEAN2} = \text{Constant target acceleration in the range } T_1 < t < T_2
\]

\[
\text{XSTART} = x_1^{-}(0)
\]

\[
\text{VSTART} = x_2^{-}(0)
\]

\[
\text{ASTART} = x_3^{-}(0)
\]

\[
\text{ISEED} = \text{Random number generator seed (odd integer)}
\]

\[
\text{INITIAL} = 0 \text{ for filter initialization using first two measurements}
\]

\[
= \text{nonzero for alternative initialization}
\]

\[
\text{ISTATE} = 2 \text{ for simulating two-state filter}
\]

\[
= 3 \text{ for simulating three-state filter}
\]

\[
\text{GAMMA} = \gamma
\]

\[
\text{GMAX} = \text{max value of } g(\kappa) \text{ for maneuver detection}
\]
QFACTOR = factor by which Q(k) matrix is increased upon detection of maneuver

AA = Value of σ_y for 2 ≤ t < 4
BB = Value of σ_y for 4 ≤ t < 6
CC = Value of σ_y for 6 ≤ t < 8
DD = Value of σ_y for 8 ≤ t
C KALMAN FILTER SIMULATION PROGRAM
C
IMPLICIT REAL*8 (A-H, O-Z)
REAL *4 YFL
DIMENSION XM(3), X(3), P(3, 3), ZM(2)
DIMENSION PM(3,3), PHI(3,3), Q(3,3)
DIMENSION S(3), SS(3), STM(3,3), QQ(3,3)
C
CHARACTER * 80 IBUFF
COMMON /CARDCOMMON / IBUFF, ICARDNUM, IOUTCHAN, INCHAN
INCHAN = 2
IOUTCHAN = 3
ICARDNUM = 0
OPEN (2, FILE='K(AL. IN', STATUS='OLD')
OPEN (3, FILE='KAL. OUT', STATUS='NEW')
C
CALL RDLINE
CALL GETDFP(1, T, 0)
CALL GETDFP(2, ALPHA, 0)
CALL GETDFP(3, SIA, 0)
CALL GETDFP(4, SIM, 0)
CALL GETDFP(5, TMAX, 0)
C
CALL RDLINE
CALL GETDFP(1, SIAM, 0)
CALL GETDFP(2, BETA, 0)
CALL GETDFP(3, XINL, 0)
CALL GETDFP(4, VINL, 0)
CALL GETDFP(5, AINRAND, 0)
C
CALL RDLINE
CALL GETDFP(1, AMEANI, 0)
CALL GETDFP(2, T1, 0)
CALL GETDFP(3, T2, 0)
CALL GETDFP(4, AMEAN2, 0)
C
CALL RDLINE
CALL GETDFP(1, XSTART, 0)
CALL GETDFP(2, VSTART, 0)
CALL GETDFP(3, ASTART, 0)
C
CALL RDLINE
CALL GETINT(1, ISEED, 0)
CALL GETINT(2, INITIAL, 0)
CALL GETINT(3, ISTATE, 0)
C
CALL RDLINE
CALL GETDFP(1, GAMMA, 0)
CALL GETDFP(2, GMAX, 0)
CALL GETDFP(3, GFACTOR, 0)
C
CALL RDLINE
CALL GETDFP(1, AA, 0)
CALL GETDFP(2, BB, 0)
CALL GETDFP(3, CC, 0)
CALL GETDFP(4, DD, 0)
C
AT = ALPHA * T
ET = DEXP(-AT)
EET = DEXP(-2.* AT)
SASO = SIA * SIA
R = SIM * SIM
A-4
C
C INITIALIZE PM MATRIX (ERROR COVARIANCE MATRIX)
IF \( \text{INITIAL\_EQ} \ 0 \) THEN

\[
\text{PM}(1,1) = R
\]
\[
\text{PM}(1,2) = R / T
\]
\[
\text{PM}(1,3) = 0
\]

\[
\text{PP} = \text{SASQ} / (\alpha^4 \cdot T^T)
\]
\[
\text{PPP} = 2 \cdot \text{AT} \cdot \text{AT} + 2 \cdot \text{AT}^3 / 3 - 2 \cdot \text{ET} - 2 \cdot \text{AT} \cdot \text{ET}
\]
\[
\text{PM}(2,2) = 2 \cdot R / (T \cdot T) + \text{PP} \cdot \text{PPP}
\]

IF (ISTATE, EQ. 2) \( \text{PM}(2,2) = 2 \cdot R / (T \cdot T) + \text{SASQ} \cdot T / 3 \).

\[
\text{PM}(2,3) = \text{SASQ} \cdot (\text{ET} + \text{AT} - 1) / (\alpha \cdot \alpha \cdot T)
\]
\[
\text{PM}(3,3) = \text{SASQ}
\]

ELSE

DO 201 J = 1, 3

CALL RDLINE

CALL GETDFP(1, PM(J, 1), 0)

CALL GETDFP(2, PM(J, 2), 0)

201 CALL GETDFP(3, PM(J, 3), 0)

END IF

C

\[
\text{PHI}(1,1) = 1
\]
\[
\text{PHI}(1,2) = T
\]
\[
\text{PHI}(1,3) = (-1 + \text{AT} + \text{ET}) / (\alpha \cdot \alpha)
\]
\[
\text{PHI}(2,1) = 0
\]
\[
\text{PHI}(2,2) = 1
\]
\[
\text{PHI}(2,3) = (1 - \text{ET}) / \alpha
\]
\[
\text{PHI}(3,1) = 0
\]
\[
\text{PHI}(3,2) = 0
\]
\[
\text{PHI}(3,3) = \text{ET}
\]

C

\[
\text{QG}1 = 1 - \text{ET} + 2 \cdot \text{AT} + 2 \cdot \text{AT}^3 / 3 - 2 \cdot \text{AT} \cdot \text{AT} - 2 \cdot \text{AT} \cdot \text{ET}
\]
\[
\text{QG}2 = \text{ET} + 1 - 2 \cdot \text{ET} + 2 \cdot \text{AT} \cdot \text{ET} - 2 \cdot \text{AT} \cdot \text{AT}
\]

C

\[
\text{QG}(1,1) = \text{QG}1 \cdot \text{SASQ} / \alpha^4
\]
\[
\text{QG}(1,2) = \text{QG}2 \cdot \text{SASQ} / \alpha^3
\]
\[
\text{QG}(1,3) = (1 - \text{ET} - 2 \cdot \text{AT} \cdot \text{ET}) \cdot \text{SASQ} / \alpha^2
\]
\[
\text{QG}(2,1) = \text{QG}(1,2)
\]
\[
\text{QG}(2,2) = (4 \cdot \text{ET} - 3 - \text{ET} + 2 \cdot \text{AT} \cdot \text{ET}) \cdot \text{SASQ} / \alpha^2
\]
\[
\text{QG}(2,3) = (\text{ET} + 1 - 2 \cdot \text{ET}) \cdot \text{SASQ} / \alpha
\]
\[
\text{QG}(3,1) = \text{QG}(1,3)
\]
\[
\text{QG}(3,2) = \text{QG}(2,3)
\]
\[
\text{QG}(3,3) = (1 - \text{ET}) \cdot \text{SASQ}
\]

C

IF (ISTATE, EQ. 2) THEN

\[
\text{QG}(1,1) = \text{SASQ} \cdot T^T / T^3
\]
\[
\text{QG}(1,2) = \text{SASQ} \cdot T^T / 2
\]
\[
\text{QG}(2,1) = \text{QG}(1,2)
\]
\[
\text{QG}(2,2) = \text{SASQ} \cdot T
\]

END IF

C

GENERATE FICTIOUS MEASURED POSITION DATA ASSUMING A STOCHASTIC MODEL WITH ACCLN. CORRELATION BETA AND ACCLN VARIANCE SIAM**2.

GENERATE STM = STATE TRANSITION MATRIX TO BE USED TO CREATE THE FICTIOUS TARGET MEASUREMENT DATA.

C

\[
\text{AT} = \text{BETA} \cdot T
\]
\[
\text{ET} = \text{DEXP}(-\text{AT})
\]
\[
\text{EET} = \text{DEXP}(-2 \cdot \text{AT})
\]
\[
\text{SASQ} = \text{SIAM} \cdot \text{SIAM}
\]

C

\[
\text{STM}(1,1) = 1
\]
\[
\text{STM}(1,2) = T
\]
\[
\text{STM}(1,3) = (-1 + \text{AT} + \text{ET}) / (\alpha \cdot \alpha)
\]
\[
\text{STM}(2,1) = 0
\]
\[
\text{STM}(2,2) = 1
\]
\[
\text{STM}(2,3) = (1 - \text{ET}) / \alpha
\]
\[
\text{STM}(3,1) = 0.
\]

GFICT = (1. - EET) * SASQ

IX = ISEED
CALL GRAND(IX, IY, YFL)
IX = IY
YFLERR = 2. * ( YFL - 0.5 ) * DSQRT(3. * R )

S(1) = 0.
S(2) = 0.
S(3) = AINRAND
XOLD = XINL
VOLD = VINL

XNEW = XOLD + T*VOLD + 0.5*T*T*AMEAN1
VNEW = VOLD + T*AMEAN1
XTRUE = XNEW + S(1)
VTRUE = VNEW + S(2)
ATRUE = S(3) + AMEAN1
ZM(1) = YFLERR + XTRUE

CALL GRAND(IX, IY, YFL)
IX = IY
YFL3 = 2. * ( YFL - 0.5 ) * DSQRT(3. * GFICT )
CALL GRAND(IX, IY, YFL)
IX = IY
YFLERR = 2. * ( YFL - 0.5 ) * DSQRT(3. * R )

SS(1) = STM(1, 1)*S(1) + STM(1, 2)*S(2) + STM(1, 3)*S(3)
SS(2) = STM(2, 1)*S(1) + STM(2, 2)*S(2) + STM(2, 3)*S(3)
SS(3) = STM(3, 1)*S(1) + STM(3, 2)*S(2) + STM(3, 3)*S(3) + YFL3

DO 33 I=1,3
33 S(I) = SS(I)

XOLD = XNEW
VOLD = VNEW
XNEW = XOLD + T*VOLD + 0.5*T*T*AMEAN1
VNEW = VOLD + T*AMEAN1
XTRUE = XNEW + S(1)
VTRUE = VNEW + S(2)
ATRUE = S(3) + AMEAN1
ZM(2) = YFLERR + XTRUE

END OF FIRST TWO MEASUREMENTS AT TIME = 0. AND TIME = T

XM(1) = ZM(2)
XM(2) = ( ZM(2) - ZM(1) ) / T
XM(3) = ASTART

ALTERNATIVELY INITIALIZED FILTER (INITIAL .NE. 0 )

IF ( INITIAL .NE. 0 ) THEN
XM(1) = XSTART
XM(2) = VSTART
END IF

TIME = 2. * T

ICOUNT = 0
SUM1 = 0.
SUM2 = 0.
SUM3 = 0.
SUM4 = 0.
SUM5 = 0.
GOLD = 0.
Z = ZM(2)

C IF(ISTATE. EQ. 2) THEN
WRITE(3, 301)
ELSE
WRITE(3, 302)
END IF

C BEGIN KALMAN FILTER LOOP
C
10 CONTINUE
C
DEL = PM(1,1) + R
GK1 = PM(1,1) / DEL
GK2 = PM(1,2) / DEL
GK3 = PM(1,3) / DEL
C
PINV = Z - XM(1)
X(1) = XM(1) + GK1 * PINV
X(2) = XM(2) + GK2 * PINV
X(3) = XM(3) + GK3 * PINV
C
P(1,1) = (1. - GK1) * PM(1,1)
P(1,2) = (1. - GK1) * PM(1,2)
P(1,3) = (1. - GK1) * PM(1,3)
P(2,2) = - GK2 * PM(1,2) + PM(2,2)
P(2,3) = - GK2 * PM(1,3) + PM(2,3)
P(3,3) = - GK3 * PM(1,3) + PM(3,3)
C
COMPUTE FADING MEMORY AVERAGE OF INNOVATION PROCESS
C
RHO = PINV * PINV / ( PM(1,1) + R )
GNEW = GAMMA * GOLD + RHO
GOLD = GNEW
C
C WRITE QUANTITIES OF INTEREST
C
PERR = -X(1) + XTRUE
VERR = -X(2) + VTRUE
AERR = -X(3) + ATRUE
SUM1 = SUM1 + PERR * PERR
SUM2 = SUM2 + YFLERR * YFLERR
SUM3 = SUM3 + PERR
SUM4 = SUM4 + YFLERR
SUM5 = SUM5 + VERR * VERR
SPE = DSQRT(PM(1,1))
SPF = DSQRT(P(1,1))
SVE = DSQRT(P(2,2))
SVF = DSQRT(P(2,2))
SAE = DSQRT(PM(3,3))
SAF = DSQRT(P(3,3))
ICOUNT = ICOUNT + 1
C
IF( ISTATE. EQ. 2 ) THEN
WRITE(3, 100) TIME, YFLERR, PERR, VERR, GK1, GK2, SPF, SVF
100 FORMAT(F7.2, 3F11.3, 2X, 4(E11.4, 2X))
FORMAT(F7.2,3F11.3,2X,7(E11.4,2X))
END IF
C
C COMPUTE PM AND XM FOR THE NEXT TIME STEP
C
IF(GNEW. GT. GMAX) THEN
C
ADAPTIVE FILTER 
BEGIN ADAPTATION
C
DO 44 I=1,3
DO 44 J=1,3
44 PM(I,J) = GFACTOR * G(I,J)
C
IF( ISTATE. EQ. 2 ) THEN
PM(1,1) = P(1,1)+ 2.*T*P(1,2) + T*T*P(2,2) + GG(1,1)
PM(1,2) = P(1,2)+ T*P(2,2) + GG(1,2)
PM(2,2) = P(2,2)+ GG(2,2)
ELSE
PM(1,1)= P(1,1)+ 2.*T*P(1,2) + 2.*PHI(1,3)*P(1,3) + T*T*P(2,2) + 2.*PHI(1,3)*P(2,3) + PHI(1,3)*PHI(1,3)*P(3,3) + GG(1,1)
PM(1,2)= P(1,2)+ PHI(2,3)*P(1,3) + PHI(2,3)*P(3,3) + PHI(2,3)*PHI(2,3)*P(3,3) + GG(1,2)
PM(1,3)= PHI(3,3)*P(1,3) + T*P(2,3) + PHI(1,3)*P(3,3) + GG(1,3)
PM(2,2)= P(2,2)+ 2.*PHI(2,3)*P(2,3) + PHI(2,3)*PHI(2,3)*P(3,3) + GG(2,2)
PM(2,3)= PHI(3,3)*P(2,3) + PHI(2,3)*P(3,3) + GG(2,3)
PM(3,3)= PHI(3,3)*PHI(3,3)*P(3,3) + GG(3,3)
END IF
ELSE
C
IF( ISTATE. EQ. 2 ) THEN
PM(1,1) = P(1,1)+ 2.*T*P(1,2) + T*T*P(2,2) + G(1,1)
PM(1,2) = P(1,2)+ T*P(2,2) + G(1,2)
PM(2,2) = P(2,2)+ G(2,2)
ELSE
PM(1,1)= P(1,1)+ 2.*T*P(1,2) + 2.*PHI(1,3)*P(1,3) + T*T*P(2,2) + 2.*PHI(1,3)*P(2,3) + PHI(1,3)*PHI(1,3)*P(3,3) + G(1,1)
PM(1,2)= P(1,2)+ PHI(2,3)*P(1,3) + PHI(2,3)*P(3,3) + PHI(2,3)*PHI(2,3)*P(3,3) + G(1,2)
PM(1,3)= PHI(3,3)*P(1,3) + T*P(2,3) + PHI(1,3)*P(3,3) + G(1,3)
PM(2,2)= P(2,2)+ 2.*PHI(2,3)*P(2,3) + PHI(2,3)*PHI(2,3)*P(3,3) + G(2,2)
PM(2,3)= PHI(3,3)*P(2,3) + PHI(2,3)*P(3,3) + G(2,3)
PM(3,3)= PHI(3,3)*PHI(3,3)*P(3,3) + G(3,3)
END IF
END IF
C
END OF ADAPTATION
C
C
END
C
C END
C
C IF( ISTATE. EQ. 2 ) THEN
XM(1) = X(1) + T*X(2)
XM(2) = X(2)
ELSE
XM(1) = X(1) + T*X(2) + PHI(1,3)*X(3)
XM(2) = X(2) + PHI(2,3)*X(3)
XM(3) = PHI(3,3)*X(3)
END IF
C
TIME = TIME + T
IF(TIME.GT. 2.) R = AA*AA
IF(TIME.GT. 4.) R = BB*BB
IF(TIME.GT. 6.) R = CC*CC
IF(TIME.GT. 8.) R = DD*DD
A=8
C
GENERATE. FICTIOUS. POSITION. MEASUREMENT. FOR. THE. NEXT. ITERATION.
CALL GRAND(IX, IY, YFL)
IX = IY
YFL3 = 2.* ( YFL - 0.5 ) * DSQRT(3. * GFICT )
CALL GRAND(IX, IY, YFL)
IX = IY
YFLERR = 2.* ( YFL - 0.5 ) * DSQRT(3. * R )

C
SS(1) = STM(1,1)*S(1) + STM(1,2)*S(2) + STM(1,3)*S(3)
SS(2) = STM(2,1)*S(1) + STM(2,2)*S(2) + STM(2,3)*S(3)
SS(3) = STM(3,1)*S(1) + STM(3,2)*S(2) + STM(3,3)*S(3) + YFL3

C
DO 34 I=1,3
34 S(I) = SS(I)

C
IF(TIME. LT. T1. OR. TIME. GT. T2) THEN
   AMEAN = AMEAN1
ELSE
   AMEAN = AMEAN2
END IF

C
XOLD = XNEW
VOLD = VNEW
XNEW = XOLD + T*VOLD + 0.5*T*T*AMEAN
VNEW = VOLD + T*AMEAN
XTRUE = XNEW + S(1)
VTRUE = VNEW + S(2)
ATRUE = S(3) + AMEAN
Z = YFLERR + XTRUE

C
IF(TIME. GT. TMAX) GO TO 11
GO TO 10

C
SMEAS = DSQRT(SUM2/ICOUNT)
SFPOS = DSQRT(SUM1/ICOUNT)
SUM3 = SUM3 / ICOUNT
SUM4 = SUM4 / ICOUNT
SUM5 = DSQRT(SUM5/ICOUNT)
WRITE(3,103)SUM4, SMEAS
103 FORMAT(/, 'MEAN MEAS ERR =',F10.4, ' RMS MEAS ERR =',F10.4)
WRITE(3,104)SUM3, SFPOS
104 FORMAT(/, 'MEAN PESTM ERR =',F10.4, ' RMS PESTM ERR =',F10.4)
WRITE(3,105)SUM5
105 FORMAT(/, 'RMS VESTM ERR =',F10.4)
STOP

C
RANDOM NUMBER GENERATOR SUBROUTINE

C
SUBROUTINE GRAND(IX, IY, YFL)
IY = IX * 65539
IF(IY)5,6,6
5 IY = IY + 2147483647 + 1
6 YFL = IY
YFL = YFL + 0 4656613E-9
RETURN
END
SAMPLE INPUT FILE FOR KALMAN FILTER PROGRAM

T, ALPHA, SIA, SIM, TMAX
SIAM, BETA, XINL, VINL, AINRAND
AMEAN1, T1, T2, AMEAN2
XSTART, VSTART, ASTART
ISEED, INITIAL, 1STATE
GAMMA, GMAX, GFACTOR
AA, BB, CC, DD
PM(1,1), PM(1,2), PM(1,3)
PM(2,1), PM(2,2), PM(2,3)
PM(3,1), PM(3,2), PM(3,3)

0.01 0.01 34 64 5.1
0.01 0.01 34 64 5.1
5.0 1.0 50.2
4.0 12.0 6.0 40.0
3.0 0.0 3.0
1193 1.0 3.0
0.0 4.0 1.0
1.0 7.0 2.4
25.0 0.0 0.0
0.0 1200.0 0.0
9.0 0.0 1200.0