ARMY RESEARCH LABORATORY

A Ballistic Filter for GPS and Accelerometer Measurements

by Andrew A. Thompson

ARL-TR-5144 **April 2010**

Approved for public release; distribution is unlimited.

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

Army Research Laboratory
Aberdeen Proving Ground, MD 21005-5069

ARL-TR-5144 **April 2010**

A Ballistic Filter for GPS and Accelerometer Measurements

Andrew A. Thompson
Weapons and Materials Research Directorate, ARL

Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing the burden, to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.
PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED (From - To)
April 2010	Final	

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
A Ballistic Filter for GPS and Accelerometer Measurements	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Andrew A. Thompson	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) U.S. Army Research Laboratory ATTN: RDRL-WML-A Aberdeen Proving Ground, MD 21005-5066	8. PERFORMING ORGANIZATION REPORT NUMBER ARL-TR-5144
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This report extends the work of Andrew A. Thompson described in the report titled Ballistics Filtering, ARL-TR-4735, published in March 2009, by showing the process of realizing the ideas through a simulation. By providing a concrete example it is hoped other realizations of the ideas can be pursued with a reasonable effort. Ballistic Filtering describes the dynamic equations that can be used to form Extended Kalman Filters (EKF) for the estimation of a projectile's trajectory. The steps associated with initialization and implementing an EKF are demonstrated through a specific task. The performance of an EKF processing Global Positioning System (GPS) observations is compared to the performance of an EKF processing both GPS and axial accelerometer observations.

15. SUBJECT TERMS

extended Kalman filter, ballistic trajectory

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
					Andrew A. Thompson
a. REPORT	b. ABSTRACT	c. THIS PAGE			19b. TELEPHONE NUMBER (*Include area code*)
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UU	44	(410) 278-6805

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

Contents

1. Introduction **1**

2. The Extended Filter Overview **1**

2.1 Initial Conditions 1

2.2 Time Propagation 2

2.3 The Observation Phase 3

2.4 Change in the State Due to Observation 4

3. Scenario **6**

4. Initialization **10**

4.1 Main Program: driver7 11

4.2 The Function: getudrag 12

4.3 The Function: S7calcF 12

4.4 The Function: S7dx 12

4.5 Simulation Results: GPS Only 13

5. Accelerometer Observations **13**

6. Conclusions **16**

Appendix A. Code for an EKF processing GPS **19**

Appendix B. Additional and Altered Files to Include Accelerometer Measurements **25**

Appendix C. Additional Routines of Potential Interest **29**

Distribution List **32**

List of Figures

Figure 1. Summary chart for a EKF.....5
Figure 2. Error in prediction of hit point.....13
Figure 3. Hit point error for an EKF processing GPS and an accelerometer error.....15
Figure 4. Estimate of ballistic factor.....16

1. Introduction

This report extends the work of Ballistics Filtering ARL-TR-4735[1] by showing the process of realizing the ideas through a simulation. By providing a concrete example it is hoped other realizations of the ideas can be pursued with a reasonable effort. Ballistic Filtering describes the dynamic equations that can be used to form Extended Kalman Filters (EKF) for the estimation of a projectile's trajectory. The steps associated with initialization and implementing an EKF are demonstrated through a specific task. The performance of an EKF processing Global Positioning System (GPS) observation is compared to the performance of an EKF processing both GPS and axial accelerometer observations. Hit point prediction error is used as the measure of effectiveness. Both filters use the same dynamics for state and covariance propagation.

Commented code is included as appendix A to allow the reader to observe the details of the implementation. It is assumed that the reader has examined the Ballistics Filtering report[2] and can reference the report. The best way to think of this implementation is as a process that propagates the state and state covariance that is interrupted from time to time by the task of processing observations. The implementation of an extended filter is conveyed symbolically by the following set of equations. All symbols represent vectors or matrices so the operations are those of linear algebra and are only scalar operations if the quantities represented are scalar. It is assumed the reader is familiar with linear algebra; Gilbert Strang has authored many excellent books on linear algebra. To develop an EKF it is necessary to have access to a set of linear algebra routines as an EKF is described with linear algebra operations. Linpack and Eispack are two well-known linear algebra packages.

2. The Extended Filter Overview

2.1 Initial Conditions

The symbol x is used to denote the state of the system and is assumed to have a normal distribution; the ~ denotes "is distributed as" and the N(a,b) denotes a normal distribution with a mean of a and covariance of b. The hat denotes, the estimate of, so $\hat{x}$ symbolizes the estimate of the state:

$$x(0) \sim N(\hat{x}(0), P(0)). \tag{1}$$

The above statement indicates that to start the filter both the state, (a vector), and the state covariance matrix, need to be input. The projectile state in this example contains three location

[1] Thompson, A. *Ballistics Filtering*; ARL-TR-4735; U.S. Army Research Laboratory: Aberdeen Proving Ground, MD, 2009.
[2] Ibid.

parameters, three velocity parameters, and one parameter representing the projectile drag coefficient. The estimate of the state is based on knowledge of the initial conditions; for example, the estimate of the launch site's location, the estimate of the launch velocity, and the estimate of the drag factor. The state covariance can be estimated based on knowledge of the techniques used to estimate the initial state value. The position variance should be based on the location method used to determine the launch site. Velocity information would be based on the uncertainty associated with the gun tube direction, tip off at barrel exit, and muzzle velocity uncertainty, etc. The uncertainty associated with the drag factor can be approximated via knowledge of model shortcomings or from recursively simulating a launch and then empirically setting the variance. As time progresses the importance of these values diminishes; however, it is important to get a reasonable start, especially when using an EKF.

2.2 Time Propagation

The first section of code is related to change of the state and the state covariance as a result of the dynamics or the plant. These can be repeated between observations to minimize the nonlinear effects. For the case investigated there are 10 propagation updates per GPS observation update.

System nonlinear dynamics plus plant noise q~N(0,Q) is represented by the following vector relationship.

$$\dot{\hat{x}}(t_k) = f(\hat{x}(t_{k-1}), t) + q(t) \tag{2}$$

$$t_k \;\; \hat{x}(t_{k+1}) = \hat{x}(t_k) + \dot{\hat{x}}(t_k)\Delta t \tag{3}$$

The covariance propagation for an EKF is updated through the following matrix relationship.

$$\dot{P}\,(t_k) = F(\hat{x}(t_k), t)P(t_{k-1}) + P(t_{k-1})F(\hat{x}(t_k), t) + Q(t_k) \tag{4}$$

$$P(t_k) = P(t_{k-1}) + \dot{P}(t_k)\Delta t \tag{5}$$

Both the state and the state covariance are updated in the manner of Euler integration; that is, the updated values are equal to the old value added to the time interval multiplied by the differential. If an observation is not available the above steps are repeated until a sensor presents an observation to the filter. Also, to predict the future values of the state the above equations would be propagated forward in time and provide an estimate of the state and the state covariance.

From a pragmatic perspective, the Q matrix is used to account for the shortcomings of the dynamic model used for the state. It has the effect of preventing the state covariance from becoming very small. If the state covariance becomes too small then it will effectively ignore the observations; this is referred to as the closing of the filter. It can be amusing to think of a filter as being narrow minded.

The next line defines the matrix $F(\hat{x}(t), t)$, this matrix needs to be available each time step for propagation. Computationally this is the most expensive step in the filter so it is worthy of effort

to find ways to minimize the number of times this computation is made. Ideas for this can be based on the state value or on information related to change in the $F(\hat{x}(t),t)$ matrix. Ideas like these account for some of the variation in EKF implementations.

$$F(\hat{x}(t_k),t) = \frac{\partial f(\hat{x}(t),t)}{\partial \hat{x}(t)} \Big| \hat{x}(t) = \hat{x}(t_k) \tag{6}$$

Since both $f(\hat{x}(t),t)$ and $\hat{x}(t)$ are vector quantities the resulting matrix $F(\hat{x}(t_k),t)$ will be a square matrix of the same dimension as the state. In the single element "ij" notation the previous matrix equation can be expressed as:

$$F_{ij}(\hat{x}(t_k),t) = \frac{\partial f_i(\hat{x}(t),t)}{\partial \hat{x}_j(t)} \Big| \hat{x}(t) = \hat{x}(t_k)\,. \tag{7}$$

2.3 The Observation Phase

The next set of equations represents the steps taken to update the state when an observation becomes available for processing. While the time propagation produces continuous change, the observation updates result in discontinuities in both the state and the state covariance. There will be a jump in the state value and the state covariance will be instantly diminished as the consequence of processing an observation. The observation and state are combined in a least squares fashion based on their respective covariances (see Thompson 1991 BRL-TR-3303[3]). It is worth noting that the accuracy of the estimate depends on precise values of the state and observation covariance. If the covariance values are precise then the result is an optimal estimate. The processing of an observation is considered to take place at a specific time step. There is a need to distinguish between the values of the state and the state covariance before and after an observation is processed. This distinction is only needed for two formulas.

Although observations can be scalar quantities, they are generally considered as vectors, and z is assumed to be a vector. This is the observation equation v~N(0,R)

$$z(t_k) = h(x(t_k),t_k) + v(t_k)\,. \tag{8}$$

In equation 8 the error is considered a sample from the normal distribution with zero mean and covariance R. Obviously R represents the measurement error and it is typical to include a model or subroutine to calculate R for each observation. In general the elements of the R matrix are:

$$R_{ij}(t_k) = \rho_{ij}(t_k)\sigma_i(t_k)\sigma_j(t_k)\,, \tag{9}$$

where the matrix R is square, σ_i represents the standard deviation of the ith observation, and the correlation between measurements is represented by ρ_{ij}. The dimension of R is the number of

[3] Thompson, III, A. A. *Data Fusion for Least Squares*; BRL-TR-3303; U.S. Army Ballistic Research Laboratory: Aberdeen Proving Ground, MD, 1991.

measurements made by a sensor at each time step. For GPS location measurements R can be treated as a constant diagonal matrix of dimension 3.

Relinearize the observation:

$$H(\hat{x}(t_k)) = \frac{\partial h(x(t))}{\partial x(t)} \Big| x(t) = \hat{x}(t_k) \,. \tag{10}$$

The dimension of this matrix will be the dimension of the observation, z, by the dimension of the state, $x(t)$. The gain due to an observation is:

$$K(t_k) = P(t_k) H^{'}(\hat{x}(t_k)) \left[H(\hat{x}(t_k)) P(t_k) H^{'}(\hat{x}(t_k)) + R(t_k) \right]^{-1}, \tag{11}$$

This matrix formula can be shown to be a least squares formulation for recursively processing observations. Note that exponent –1 represents matrix inversion and the operation is the matrix transpose operator.

2.4 Change in the State Due to Observation

The preobservation value of the state needs to be distinguished from the value of the state after the observation is processed. In this notation the minus sign (–) indicates previous to the observation being processed, the subscript k indicates that $t = t_k$, and the plus sign (+) indicates after the observation has been processed:

$$\hat{x}(t_k^-) = \hat{x}(t_k) \tag{12}$$

$$\hat{x}(t_k^+) = \hat{x}(t_k^-) + K(t_k)(z(t_k) - h(\hat{x}(t_k^-), t_k)) \tag{13}$$

$$\hat{x}(t_k) = \hat{x}_k(t_k^+) \,. \tag{14}$$

Update the state covariance via observation, the symbol I is used to represent the identity matrix for this example it would be seven dimensional or I_7. The portion of the formula, $z(t_k) - h(\hat{x}(t_k^-), t_k)$, is sometimes referred to as the innovation as it represents the new information being incorporated into the state. For the state covariance the following matrix operations take place:

$$P(t_k^-) = P(t_k) \tag{15}$$

$$P(t_k^+) = \left[I - K(t_k) H(\hat{x}(t_k), t_k) \right] P(t_k^-) \tag{16}$$

$$P(t_k) = P(t_k^+) \,. \tag{17}$$

The observation phase of the code can be repeated for different sensors or measurements; the observation matrix and the measurement covariance would differ for different sensors. The observation cycle of the individual sensors must be considered in the design of an EKF. It is

easy to envision an EKF moving along with state propagation but using an interrupt system to process specific observations. The previous perception makes the design of an EKF for variable time steps straightforward. Figure 1 presents a summary of an EKF in flowchart form.

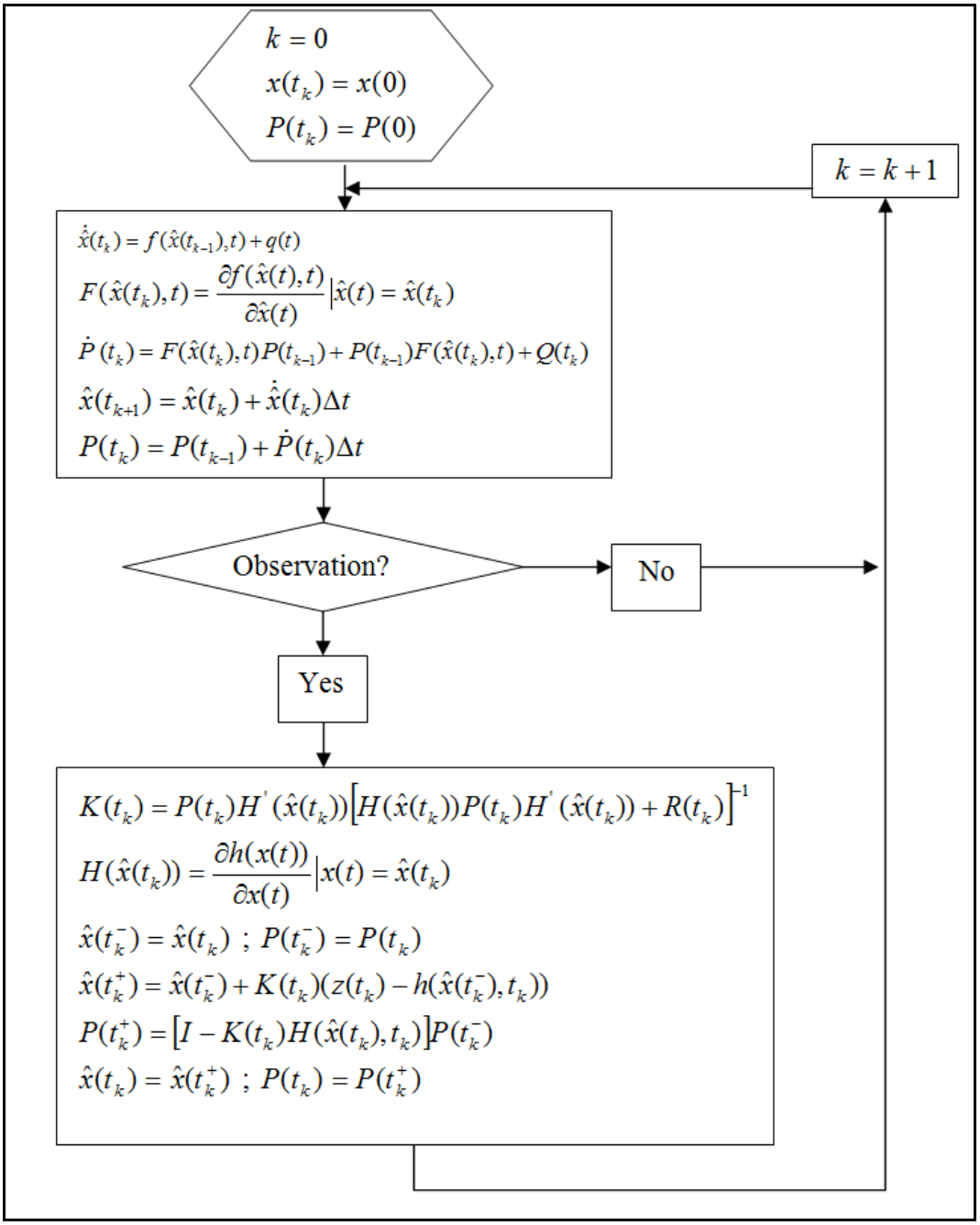

Figure 1. Summary chart for a EKF.

3. Scenario

The first scenario is to investigate the performance of an EKF processing GPS measurements to predict the hit point of a mortar trajectory. The second scenario adds an axial accelerometer to the sensor suite. The actual trajectory was generated from a 6dof model. The points on this trajectory will be used to generate observations for the EKF. By using a 6dof model, there will be nonlinearities in the trajectory that are not modeled by the EKF. In this scenario a simple state model will be used. Most of the variation in EKF implementation is due to the dynamics used. The selection of dynamics is a tradeoff between model fidelity, functionality, and speed of computation.

For this scenario the state will consist of seven dimensions location, velocity, and ballistic coefficient. The down range direction is given by x_1. The vertical direction is given by x_2. The cross range direction is given by x_3. The only effect captured will be the drag; this is done through the ballistic factor using universal drag curves. Other drag models could be used, but this presents a default model that works well with artillery and mortar rounds. The uncertainty due to unmodeled dynamics is captured by the matrix q(t). The following equations are used to model the dynamics, sometimes this is referred to as the plant. The dynamic models used are taken directly from STANAG 4355[4] or are simplified versions of the dynamics presented therein. A detailed description of the units for a more complete set of models can also be found in STANAG 4355.

$$f_1 = x_4 \tag{18}$$

$$f_2 = x_5 \tag{19}$$

$$f_3 = x_6 \tag{20}$$

$$f_4 = -x_7 A k_d V x_4 - \frac{g x_1}{R_e} - 2\Omega_y x_6 \tag{21}$$

$$f_5 = -x_7 A k_d V x_5 - g(1 - \frac{(x_1^2 + x_3^2)^{.5}}{2R_e}) + 2\Omega_x x_6 \tag{22}$$

$$f_6 = -x_7 A k_d V x_6 - \frac{g x_3}{R_e} - 2\Omega_x x_5 + 2\Omega_y x_4 \tag{23}$$

$$f_7 = 0 \tag{24}$$

[4] NATO STANAG 4355, *Modified Point Mass Trajectory Model*; North Atlantic Treaty Organization, January 20, 1997.

These equations, represented above as $f(\hat{x}(t),t)$, incorporate drag, gravity, and Coriolis Effect; other elements of the dynamics such as lift and Magnus effect are ignored.

This scenario will model a mortar round fired north from a latitude of 45 degrees north. The shot elevation is 45 degrees with a speed of 220 meters per second (m/s). The launch conditions are used to initialize projectile state vector $\hat{x}(t_0)$. The state covariance matrix $P(t_0)$ can be estimated based on knowledge of the techniques used to estimate the initial state value. The position variance should be based on the location method used to determine the launch site. Velocity information would be based on the uncertainty associated with the gun tube direction, tip off at barrel exit, and muzzle velocity uncertainty, etc. The uncertainty associated with the drag factor can be approximated via knowledge of model shortcomings or from recursively simulating a launch and then empirically setting the variance. As time progresses the importance of these values diminishes; however, it is important to get a reasonable start, especially when using an EKF.

Although a GPS sensor typically makes one observation a second, they can easily be programmed to present 5 observations a second and with some effort they can make 10 or more. It is assumed that the round contains a GPS sensor that can make 10 independent observations per second. The ability of the filter to predict impact point will be observed. If the filter performance is not adequate then it can be concluded that a GPS receiver is not adequate given the chosen dynamics.

In order to propagate the state covariance the partials of the dynamic equations with respect to the state are needed the needed partials are:

$$F_{14} = 1, \tag{25}$$

$$F_{25} = 1, \tag{26}$$

$$F_{36} = 1, \tag{27}$$

$$F_{41} = \frac{-g}{R_e}, \tag{28}$$

$$F_{42} = -x_7 k_d V x_4 \frac{\partial A}{\partial x_2}, \tag{29}$$

$$F_{44} = -x_7 A\left(V x_4 \frac{\partial k_d}{\partial x_4} + k_d x_4 \frac{\partial V}{\partial x_4} + k_d V \right), \tag{30}$$

$$F_{45} = -x_7 A x_4 \left(V \frac{\partial k_d}{\partial x_5} + k_d \frac{\partial V}{\partial x_5} \right), \tag{31}$$

$$F_{46} = -x_7 A x_4 \left(V \frac{\partial k_d}{\partial x_6} + k_d \frac{\partial V}{\partial x_6} \right) - 2\Omega_y \,, \tag{32}$$

$$F_{47} = -A k_d V x_4 \,, \tag{33}$$

$$F_{51} = \frac{g}{2R_e} \frac{x_1}{(x_x^2 + x_3^2)^{.5}} \,, \tag{34}$$

$$F_{52} = -x_7 k_d V x_5 \frac{\partial A}{\partial x_2} \,, \tag{35}$$

$$F_{53} = \frac{g}{2R_e} \frac{x_3}{(x_x^2 + x_3^2)^{.5}} \,, \tag{36}$$

$$F_{54} = -x_7 A x_5 \left(V \frac{\partial k_d}{\partial x_4} + k_d \frac{\partial V}{\partial x_4} \right), \tag{37}$$

$$F_{55} = -x_7 A \left(V x_5 \frac{\partial k_d}{\partial x_5} + k_d x_5 \frac{\partial V}{\partial x_5} + k_d V \right), \tag{38}$$

$$F_{56} = -x_7 A x_5 \left(V \frac{\partial k_d}{\partial x_6} + k_d \frac{\partial V}{\partial x_6} \right) + 2\Omega_x \,, \tag{39}$$

$$F_{57} = -A k_d V x_5 \,, \tag{40}$$

$$F_{62} = -x_7 k_d V x_6 \frac{\partial A}{\partial x_2} \,, \tag{41}$$

$$F_{63} = \frac{-g}{R_e} \,, \tag{42}$$

$$F_{64} = -x_7 A x_6 \left(V \frac{\partial k_d}{\partial x_4} + k_d \frac{\partial V}{\partial x_4} \right) + 2\Omega_y \,, \tag{43}$$

$$F_{65} = -x_7 A x_6 \left(V \frac{\partial k_d}{\partial x_5} + k_d \frac{\partial V}{\partial x_5} \right) - 2\Omega_x \,, \tag{44}$$

$$F_{66} = -x_7 A \left(V x_6 \frac{\partial k_d}{\partial x_6} + k_d x_6 \frac{\partial V}{\partial x_6} + k_d V \right), \tag{45}$$

$$F_{67} = -A k_d V x_6 \,. \tag{46}$$

The following information is also needed to obtain numerical values for the F-matrix:

$$V = (x_4^2 + x_5^2 + x_6^2)^{.5}$$
$$\frac{\partial V}{\partial x_i} = \frac{x_i}{(x_4^2 + x_5^2 + x_6^2)^{.5}} \qquad i = \{4,5,6\}, \tag{47}$$

$$\begin{aligned} a_0 &= 1.223 \\ a_1 &= 1.071e-4 \\ A &= a_0 e^{-a_1 x_2} \qquad , \\ \frac{\partial A}{\partial x_2} &= -a_0 a_1 e^{-a_1 x_2} \end{aligned} \tag{48}$$

$$g = 9.80665(1 - .0026\cos(2Lat)), \tag{49}$$

$$R_e = 6356766. \tag{50}$$

A is the air density as a function of altitude x_2, g is the acceleration of gravity as a function of altitude, and R_e is the radius of the Earth. The drag coefficient is calculated using a fourth degree polynomial of Mach number m. Mach number is the speed divided by the speed of sound. Notice that in this formulation the partial of the speed of sound with respect to height is not included in the F-matrix.

$$\begin{aligned} s_0 &= 340.3 \qquad temperture = 59^o F \\ x_{sl} &= hieght(launchabovesealevel) \\ c_v &= 2.26e-5 \\ s &= s_0(1 - c_v(x_{sl} + x_2))^{.5} \\ k_d &= \sum_{i=0}^{4} c_i m^i \\ \frac{\partial k_d}{\partial x_i} &= \left(\sum_{i=1}^{4} i c_i m^{i-1} \right) \frac{x_i}{sV} \end{aligned} \tag{51}$$

$$\begin{aligned} \Omega &= 7.2921e-5 \\ \Omega_x &= \Omega\cos(lat) \\ \Omega_y &= \Omega\sin(lat) \end{aligned} \tag{52}$$

s_0 is the speed of sound a sea level in m/s and s is the speed of sound at altitude x_{sl}+x_2. k_d is the drag coefficient is calculate using a 4th order polynomial fit. Ω_1 and Ω_2 are the Coriolis factor in the I=1 and I=2 directions.

In the example, the Q matrix is assumed to be constant in time. There is assumed to be no correlation between errors so all of the off diagonal terms are zero. All of the main diagonal terms are set to 0.05:

$$Q = \begin{bmatrix} 0.05 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.05 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.05 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.05 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.05 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.05 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.05 \end{bmatrix}. \tag{53}$$

In this example the measurements to be used are assumed to be estimates of position from a GPS receiver. It has been assumed that observations are independent of one another and constant in time. Neither of these assumptions is true in a real GPS but have been made here to simplify the calculation. The R matrix is constant over time:

$$R(t_k) = \begin{bmatrix} 4 & 0 & 0 \\ 0 & 9 & 0 \\ 0 & 0 & 4 \end{bmatrix}, \tag{54}$$

The observation matrix H is expressed as follows:

$$H = \begin{bmatrix} \frac{\partial z_1}{\partial x_1} & \frac{\partial z_1}{\partial x_2} & \frac{\partial z_1}{\partial x_3} & \frac{\partial z_1}{\partial x_4} & \frac{\partial z_1}{\partial x_5} & \frac{\partial z_1}{\partial x_6} & \frac{\partial z_1}{\partial x_7} \\ \frac{\partial z_2}{\partial x_1} & \frac{\partial z_2}{\partial x_2} & \frac{\partial z_2}{\partial x_3} & \frac{\partial z_2}{\partial x_4} & \frac{\partial z_2}{\partial x_5} & \frac{\partial z_2}{\partial x_6} & \frac{\partial z_2}{\partial x_7} \\ \frac{\partial z_3}{\partial x_1} & \frac{\partial z_3}{\partial x_2} & \frac{\partial z_3}{\partial x_3} & \frac{\partial z_3}{\partial x_4} & \frac{\partial z_3}{\partial x_5} & \frac{\partial z_3}{\partial x_6} & \frac{\partial z_3}{\partial x_7} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{55}$$

In this case the simplicity is the result of the measurements being the same as the position portion of the state variable. Throughout the remainder of this report a task will be mentioned and then the code associated with that task will be discussed.

4. Initialization

The initialization problem can be subdivided into two major sections: those tasks associated with the dynamics being used, and those associated with the operation of the filter. The routine INIT8 sets up the universal or default model for drag. In other implementations models for lift

and spin may be required and this would be a logical spot to include them. The drag model is a polynomial fit between breakpoints; thus, the shape of the curve is generated through these models. The data gives the breakpoints and the coefficients of the fitted polynomials between each set of breakpoints

The task of the routine INIT is to set up variables for the filter; most of these are related to the particular scenario. Line 7 perturbs the assumed starting point from the actual starting point. Lines 8–12 set up the assumed original velocity. The actual 6dof was fired at 45 degrees so there is 4 degrees of error included. Line 13 initializes the value of the ballistic factor. Line 15 sets up the default speed of sound. Lines 18–21 set up the q matrix. The q matrix represents the short comings of the chosen dynamics; part of designing an EKF is the process of adjusting the q matrix to a satisfactory value. There is rarely a way to predetermine the q matrix; typically simulations or multiple runs are used to tune these values. Lines 25–27 are the model for the observation error. These models can be quite complex and may need to consider the situational geometry. For this scenario GPS observations are assumed to be for position and have the same error structure from observation to observation; thus, the h matrix will not change from observation to observation. Lines 29–30 define the h matrix; this is the observation in terms of the state variables. Since GPS gives position this matrix just selects the three position variables and ignores the other state variables. Line 32 is just to define an identity matrix of the same order as the state. The last task of the INIT routine is to define the state uncertainty. The p matrix is used to represent state covariance. The use of ancillary knowledge and/or simulations can indicate good values for the initial state covariance. In this case each position error and each velocity error is assumed to have a variance of 1. The variance of the ballistic coefficient is assumed to be .0001. This value was chosen by observing the variation in this parameter over several simulations and selecting a value that reflected the observed variation.

4.1 Main Program: driver7

After initialization the main program starts. This is basically a loop that processes the observations. Typically in an EKF each observation is associated with a cycle of the filter; however, in this example there are 10 sub cycles associated with each observation. These sub cycles can be thought of as a way to minimize the effects of linearization. Rather than one large step being taken between observations the interval is subdivided to minimize the error associated with the assumption of linearity. The basic time step is 1/100 of a second and the observations come at the rate of 10 per second. Line 4 calls the previously discussed initialization routines. Line 5 sets the time step. Line 9 loads the observations. At this point in the program this is the true trajectory that was generated from a 6dof model. Noise will be added to these values before they are fed to the EKF as observations. Lines 11–12 set the earth's latitude and radius. In lines 13–14 variables related to wind speed are set to zero. Line 16 sets constants that are used for the standard atmosphere model so air density can be calculated. Lines 18–30 set variables used for Mach number, the gravity model, and Coriolis. Line 31 dimensions the f matrix and initializes it to be all zeros. Lines 33–35 initialize a set of record keeping variables. These are for the state,

the predictions, and the trace of the covariance matrix. Line 40 initializes a counter for between observation updates of the state and the state covariance. There are 10 propagation updates for each observation update. Line 45 initializes a counter for observations.

The main loop is controlled by the height of the projectile, once the height returns to ground level the simulation ends. Lines 50–54 get the air pressure, velocity, and Mach number. Lines 55–56 the drag is calculated. Line 57 calls a routine to calculate the f matrix. This is needed for the covariance propagation. Line 58 calls a routine to get the change in the state. The state and state covariance are propagated in lines 59 and 60. Line 61 increments the observation subinterval counter.

The steps to account for an observation are in lines 63–79. Lines 66 and 67 create the observation by adding noise to the actual location. After this the observation subinterval counter is reset in line 69. The gain of information due to the observation is represented by the matrix K; this is calculated in line 72. The new state based on a least squares combination of the state and the observation is calculated on line 73. The new state covariance that includes the reduction due to processing the observation is calculated on line74. Line 77 calls the prediction routine. The remaining lines update variables associated with record keeping and closing loops.

4.2 The Function: getudrag

This function is used to find the drag based on a universal drag curve. The basic information was read in the routine init8. This routine is basically a binary search to select the correct set of drag coefficients. Lines 7–31 select the proper row of the drag matrix. Once this is completed the interpolation polynomials are set up for the drag and the derivative of the drag and the values are calculated. Lines 33–35 complete the calculation.

4.3 The Function: S7calcF

The rational for the calculations in this routine are discussed in the report Ballistic Filtering. Basically the rows are the partials of the dynamic equations with respect to the state variables. Partials that are small across the entire trajectory can be dropped with small effect; of course doing this depends on the application. The savings in computation time can be considerable.

First partials of the velocity are found and then partials of the Mach number. These are done in lines 8 to 16. Line 18 states in terms of the state that the change in position is due to the velocity. The partials of equation 46 from Ballistic Filtering[5] are calculated in lines 22–27. The rest of the function performs similar calculations for the other components of velocity.

4.4 The Function: S7dx

This function calculates the change in the state based upon the dynamics.

[5] Thompson, A. *Ballistics Filtering*; ARL-TR-4735; U.S. Army Research Laboratory: Aberdeen Proving Ground, MD, 2009.

4.5 Simulation Results: GPS Only

The simulation described above was used to investigate the performance of the seven state ballistic EKF. Most GPS sensors give yield one sample per second; it is a minor modification to get five observations per second, and possible to obtain 10 per second. A sample rate of 10 was chosen as this will be the best possible situation. The results are summarized in figure 2. After an initial increase the error decreases exponentially.

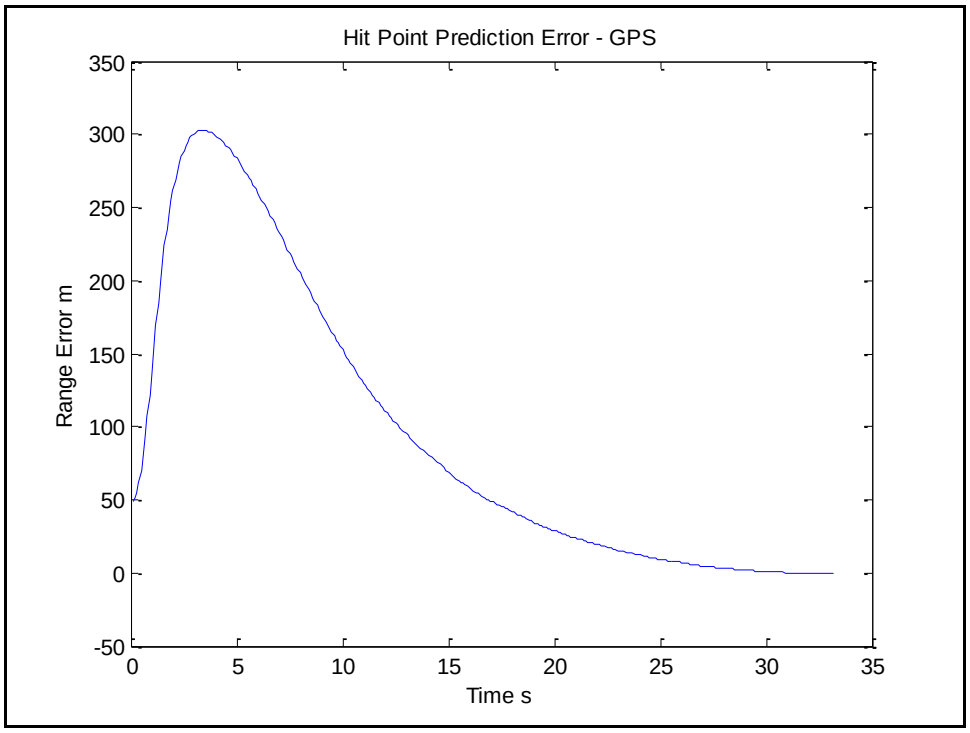

Figure 2. Error in prediction of hit point.

The range error grows for about 5 seconds (s) peaking at about 300 meters (m) before it starts to diminish to an error of –.35 m at the end of flight. After 22 s the error is approximately 20 m and after 28 s the error is less than 3 m. The performance of an EKF typically provides estimates that other subsystems utilize; and thus directly influences system performance.

5. Accelerometer Observations

Next the effects of adding an accelerometer to the sensing unit are investigated. Appendix B includes the additional and altered routines for this situation. An accelerometer aligned with the spin axis is assumed to be aligned with the velocity. This will never be true as precession and nutation result in the spin axis oscillating around the direction of motion. The angle between the spin axis and the velocity vector is typically small—on the order of a few degrees. This angle is also related to the spin of the projectile with lower spin being associated with larger angles. In

this simulation the attainable improvement in accuracy due to including an axial accelerometer on a projectile that already contains GPS is investigated.

The force seen by the accelerometer is the inner product of the force vector and the normalized orientation of the sensing direction of the accelerometer. Assuming the accelerometer is on the spin axis and the spin axis is pointing in the direction of velocity will present the best possible case. In this situation the force seen by the accelerometer is the inner product of the force and the normalized velocity. The previous equations f_4, f_5, f_6 give the force vector in terms of state variables, and the direction of the velocity can be represented as the vector $\left(\frac{x_4 \quad x_5 \quad x_6}{v}\right)'$ where v is the magnitude of the velocity. The inner product of these two quantities results in the following lengthy expression.

$$\begin{aligned} &-x_7 A k_d v x_4 \frac{x_4}{v} - \frac{g x_1}{R_e}\frac{x_4}{v} - 2\Omega_y \frac{x_4}{v} - x_7 A k_d v x_5 \frac{x_5}{v} - g\left(1 - \frac{\left(x_1^2 + x_3^2\right)^{.5}}{2R_e}\right)\frac{x_5}{v} + 2\Omega_x x_6 \frac{x_5}{v} \\ &-x_7 A k_d v \frac{x_6}{v} - \frac{g x_3}{R_e}\frac{x_6}{v} - 2\Omega_x x_5 \frac{x_6}{v} + 2\Omega_y x_4 \frac{x_6}{v} \end{aligned} \tag{56}$$

Taking the partial of the above expression with respect to each of the state variables gives the elements of the observation matrix for axial accelerometer measurements. This will be a row rather than a column. Note that the values of h have been simplified.

$$H_1 = \frac{-g x_4}{R_e v} + \frac{g x_5}{2R_e v}\left(x_1^2 + x_3^2\right)^{-.5} x_1 \tag{57}$$

$$H_2 = 0 \tag{58}$$

$$H_3 = \frac{-g x_6}{R_e v} + \frac{g x_5}{2R_e v}\left(x_1^2 + x_3^2\right)^{-.5} x_3 \tag{59}$$

$$H_4 = -x_7 A k_d 2x_4 - \frac{g x_1}{R_e v} \tag{60}$$

$$H_5 = -x_7 A k_d 2x_5 - \frac{g}{v}\left(1 - \frac{\left(x_1^2 + x_3^2\right)^{.5}}{2R_e}\right) \tag{61}$$

$$H_6 = -x_7 A k_d 2x_6 - \frac{g x_3}{R_e v} \tag{62}$$

$$H_7 = -A k_d v^2 \tag{63}$$

An examination of the values of H indicates that H_7 will have the largest magnitude; thus axial accelerometer measurements will have a relatively large influence on the estimate of the ballistic factor, or terms directly related to drag. Since drag and an axial accelerometer both work along the direction of the velocity vector, this is not too surprising.

The modification to the previous ECF to accommodate the new axial accelerometer observations is straightforward. A new block is added to process the observations at the proper time intervals. In this situation the previously used blocks for state propagation and GPS observation processing remain unchanged. The GPS observations are processed before the accelerometer readings when both observations are available at the same instance of time. An examination of the routine driver7gpsacc demonstrates how to extend the previous EKF to process axial accelerometer observations. The routine accel is used to generate the force seen by the accelerometer. The routine hmatrix generates the H-matrix for each accelerometer observation.

A comparison of figure 2 and figure 3 illustrates the change in prediction error due to including an axial accelerometer. The new sensor suite has a lower maximum.

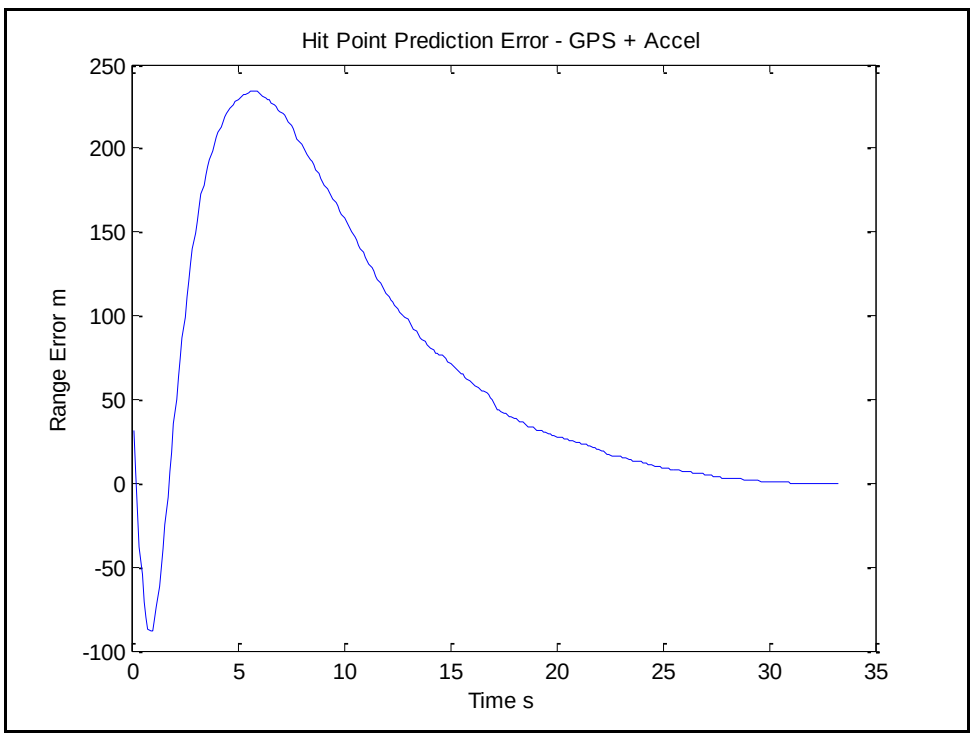

Figure 3. Hit point error for an EKF processing GPS and an accelerometer error.

The hit point prediction error is similar to the GPS only EKF after 15 s. The advantages associated with adding an accelerometer can only be assessed with respect to a specific system.

Figure 4 shows the estimate of ballistic factor as a function of time. This figure is of interest because it shows the filter adjusting the overall drag based on the observations. Given a perfect model of the drag this value would be constant. The filter diminishes the ballistic factor for

5 s and then increases it for the remainder of the flight. This gives an indication of some of the shortcomings of the seven dimensional point mass dynamics attempting to mimic the dynamics of a 6dof model.

Appendix C includes additional routines that may be of interest. Included are simplified models for drift and spin. These routines provide a default capability.

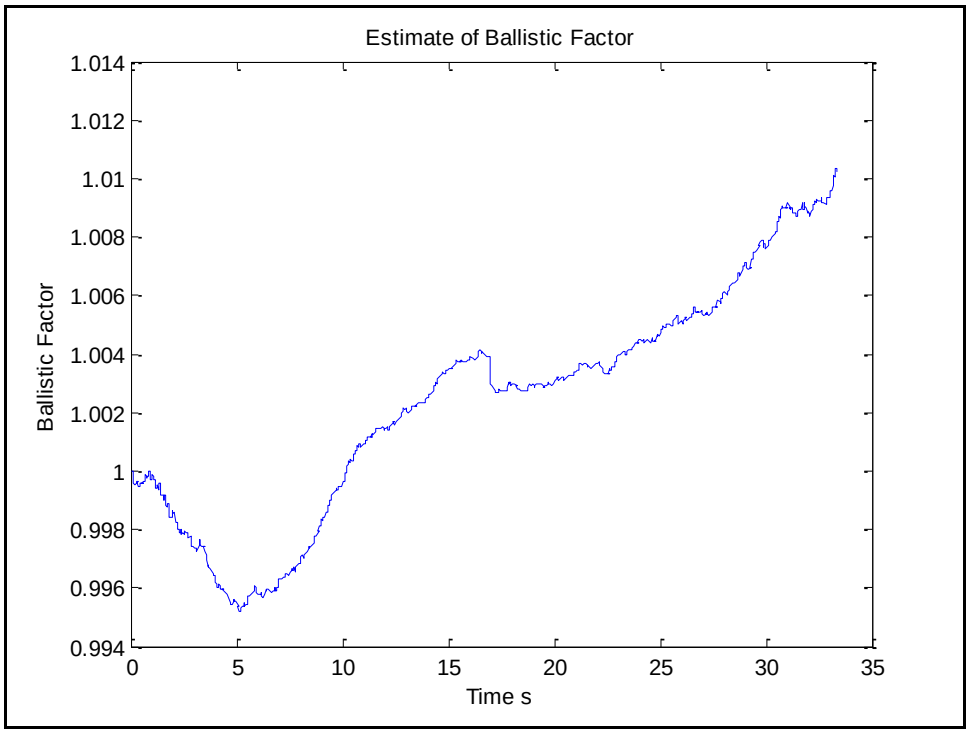

Figure 4. Estimate of ballistic factor.

6. Conclusions

There is no precise step-by-step way to design a Kalman filter (KF) or an EKF. This paper has demonstrated one path through the set of possible decisions to design an EKF. It is evident that there are many possible settings that can and should be investigated in setting up a filter. A central question is the quality of the dynamics used. Usually this means the simplest set of dynamics that will allow the system to get the job done. The sensor suite has a huge effect on filter performance and it is always desirable to have high quality measurements. Also, the timeliness of the measurements influences filter performance in that many observations can offset the shortcomings of a given dynamics model. The possibilities to investigate seem endless even for well defined problems; thus the design of a KF or an EKF is typically based on meeting system requirements. Conceptually it is beneficial to conceive of the filter as a set of dynamic equations that get interrupted to receive corrections based on observations.

In this investigation an EKF processing GPS observations was compared to an EKF processing GPS and axial accelerometer observations. The results showed little difference for the predictions at times exceeding 15 s. After an EKF has achieved accurate values of the state additional observations will not add much value; however, additional observations will keep the state from drifting away from the true values. The value of additional information depends on how it related to the state through the observation matrix, the covariance of the observation, and the accuracy of the state.

INTENTIONALLY LEFT BLANK.

Appendix A. Code for an EKF processing GPS

Initx

```
%function y=initx ()
%this next routine sets up the drag, lift, and spin curves
inits8
%the state will contain location velocity and drag
%set up state variable
x=zeros(7,1);
x(1:3)=[.01;.01;.01];
speed=220;
el=49/180*pi;
x(4)=speed*cos(el);
x(5)=speed*sin(el);
x(6)=0;
x(7)=1;
%speed of sound for sea level about 53 degrees
v_s=340.3;
%terms for model mismatch
q=zeros(7);
q(1,1)=1;q(2,2)=1;q(3,3)=1;
q(4,4)=1;q(5,5)=1;q(6,6)=1;
q=q*.05;
%error associated with the observations
%this is used to simulate GPS cband
%r is the observation variance
r=zeros(3);
r(1,1)=4;r(2,2)=9;r(3,3)=4;
r_sd=[2;3;2];
%make observation matrix in terms of state
h=zeros(3,7);
h(1,1)=1;h(2,2)=1;h(3,3)=1;
%the dimension of the state for I used in covariance propagation
I=eye(7);
%set up initial state uncertainty
p=eye(7);
p=p*.001;
p(1,1)=1;p(2,2)=1;p(3,3)=1;
p(4,4)=1;p(5,5)=1;p(6,6)=1;
p(7,7)=.0001;
```

Published with MATLAB® 7.5

Inits8

```
%drag coefs for polynomial to estimate drag
dr1=[.27754e-4 0 0 0 0];
dr2=[.29407446e-3 -.16856609e-2 .39541129e-2 -.40706187e-2 .15497302e-2];
dr3=[-.56492074 .24140455e1 -.38636028e1 .27445913e1 -.73004070];
dr4=[.16449122e1 -.6472231e1 .95427249e1 -.62486232e1 .15332897e1];
dr5=[-.38991679e-2 .12251269e-1 -.14008227e-1 .70697832e-2 -.13323438e-2];
```

```
dr6=[.17693159e-3 -.14042065e-3 .74643008e-4 -.21621397e-4 .26788426e-5];
%drag breakpoints for sets of coef
udrag_bp=[.6 .9 .99 1.06 1.5 5];
udrag=[dr1;dr2;dr3;dr4;dr5;dr6];
```

Published with MATLAB® 7.5

Driver7

```
%function y=driver7()
%do the first intialization phase
initx
dt=.01;
%these are from the 6dof trajectory
%obviously in real time they would not be known in advance
%these are used as the observation feed
load obsdat
lat=45/180*pi;  %adefault value
r_e=6378000; h0=0;
x_w=0;%0 indicates no wind
y_w=0;
%air pressure
ap_c0=1.223; ap_c2=1.071e-4;
%mach number
v_s0=340.3; %59F
v_c=2.26e-5;
%gravity
g0=9.80665;
g1=.0026;
g=g0*(1-g1*cos(2*lat));
%corriolis
omega=7.2921e-5;
om_v=[omega*cos(lat);omega*sin(lat);0];
%allocate f-matrix
f=zeros(7);
state=[x];
pre=[];
ptrace=trace(p);
%this is the counter for between observation updates
%this helps minimize the effects of nonlinearity
%many times these substeps can be ignored and the propagation can
%be calculated in one step between observations
obs=0;
%the next variable just counts the observations
%the observations start with count+1
%note that this variable can be used to start the observation stream
%at any desired point of the trajectory
count=1;
%main loop
while x(2)>0
    %air preasure etc is triggered by altitude
    %anyhow update variables that change as the state changes
    air_p=ap_c0*exp(-ap_c2*x(2));  %current air pressure
    %velocity and mach number
```

```
    v_s=v_s0*(1-v_c*(x(2)+h0))^.5;
    v=sqrt((x(4)-x_w)^2+x(5)^2+(x(6)-y_w)^2);
    m=v/v_s;  %mach number
    k=getudrag(m,udrag,udrag_bp); %udrag&udrag_bp need 2b initialized
    kd=k(1);kd_m=k(2);
    f=S7calcF(x,air_p,kd,kd_m,v,m,r_e,om_v,ap_c2,g,f);
    dx=S7dx(x,air_p,kd,v,r_e,om_v,g);
    x=x+dx*dt;
    p=propP(dt,f,p,q);
    obs=obs+1;
    %this inside loop is triggered by a GPS observation being available
    if obs==10
        count=count+1;
        %add some error to the observation
        z_er=diag(randn(3,1))*r_sd*.1;
        z=obsdat(:,count)+z_er;
        %reset counter
        obs=0;
        %if r is variable the routine getR should be designed to model
        %the observation error and included here
        K=getK(p,h,r);
        x=x+K*(z-h*x);
        p=(I-K*h)*p;
        %predict routine this will change somewhat based on specific
        %application
        %pos=predictX(x,r_e,om_v,h0,x_w,y_w,udrag,udrag_bp,g,dt,5);
        pos=predictH(x,r_e,om_v,h0,x_w,y_w,udrag,udrag_bp,g,dt);
        pre=[pre pos];
    end
    state=[state x];
    ptrace=[ptrace trace(p)];
end
y=state([1 3 2],:);
```

Published with MATLAB® 7.5

Getudrag

```
function y=getudrag(m,drag,bp)
%this finds the drag based on a universal drag curve
%the operations are set up under the assumptin that most of the time is
%spent near m=1 can be further improved by inserting the actual values
%rather than passing the breakpoints
row=1;
if m<bp(3)
        if m>bp(2)
            row=3;
        else
            if m>bp(1)
                row=2;
            else
                row=1;
            end
        end
else
```

```
    if m<bp(4)
        row=4;
    else
        if m<bp(5)
            row=5;
        else
            if m<bp(6)
                row=6;
            else
                row=0;
            end
        end
    end
end

x=[1; m; m*m; m*m*m; m*m*m*m];
dx=[1;2*m;3*m*m;4*m*m*m];
y=[drag(row,:)*x drag(row,2:5)*dx];
```

Published with MATLAB® 7.5

S7calcF

```
function f=S7calcF(x,air_p,kd,kd_m,v,m,r_e,om_v,ap_c2,g,f)
ap_x2=-ap_c2*air_p;
%v=sqrt(x(4)^2+x(5)^2+x(6)^2);
%m=v/v_s;
%partials of velocity follow
v_x4=x(4)/v;
v_x5=x(5)/v; %remember x5 is velocity in height or altitude
v_x6=x(6)/v;
%m_x2=-v/v_s^2*vs_x2;
mv_x4=m*v_x4;
mv_x5=m*v_x5;
mv_x6=m*v_x6;
f(1,4)=1;f(2,5)=1;f(3,6)=1;
%start calculation of the F matrix Iguess f will b 14d
%only the 4-6 rows change each step
%row 4
f(4,1)=-g/r_e;
f(4,2)=-x(7)*v*x(4)*ap_x2*kd;
f(4,4)=-x(7)*air_p*(v*kd_m*mv_x4*x(4)+v_x4*kd*x(4)+kd*v);
f(4,5)=-x(7)*air_p*x(4)*(v*kd_m*mv_x5+v_x5*kd);
f(4,6)=-x(7)*air_p*x(4)*(v*kd_m*mv_x6+v_x6*kd)-2*om_v(2);
f(4,7)=-air_p*kd*v*x(4);
%row 5
f(5,1)=g/(2*r_e)*x(1)/sqrt(x(1)*x(1)+x(3)*x(3));
f(5,2)=-x(7)*v*x(5)*ap_x2*kd;
f(5,3)=g/(2*r_e)*x(3)/sqrt(x(1)*x(1)+x(3)*x(3));
f(5,4)=-x(7)*air_p*x(5)*(v*kd_m*mv_x4+v_x4*kd);
f(5,5)=-x(7)*air_p*(v*kd_m*mv_x5*x(5)+v_x5*kd*x(5)+kd*v);
f(5,6)=-x(7)*air_p*x(5)*(v*kd_m*mv_x6+v_x6*kd)+2*om_v(1);
f(5,7)=-air_p*kd*v*x(5);
%row 6
f(6,2)=-x(7)*v*x(6)*ap_x2*kd;
```

```
f(6,3)=-g/r_e;
f(6,4)=-x(7)*air_p*x(6)*(v*kd_m*mv_x4+v_x4*kd)+2*om_v(2);
f(6,5)=-x(7)*air_p*x(6)*(v*kd_m*mv_x5*+v_x5*kd)-2*om_v(1);
f(6,6)=-x(7)*air_p*(v*kd_m*mv_x6*x(6)+v_x6*kd*x(6)+kd*v);
f(6,7)=-air_p*kd*v*x(6);
```

Published with MATLAB® 7.5

S7dx

```
function dx=S7dx(x,air_p,kd,v,r_e,om_v,g)
dx=zeros(7,1);
dx(1)=x(4);
dx(2)=x(5);
dx(3)=x(6);
dx(4)=-x(7)*air_p*kd*v*x(4)-g*x(1)/r_e-om_v(2)*x(6);
dx(5)=-x(7)*air_p*kd*v*x(5)-g*(1-
sqrt(x(1)*x(1)+x(3)*x(3))/(2*r_e))+om_v(1)*x(6);
dx(6)=-x(7)*air_p*kd*v*x(6)-g*x(3)/r_e+om_v(2)*x(4)-om_v(1)*x(5);
```

Published with MATLAB® 7.5

propP

```
function pnew=propP(dt,f,p,q)
pdot=f*p+p*f';
pnew=p+(pdot+q)*dt;
```

Published with MATLAB® 7.5

getK

```
function k=getK(p,h,r)
%calculate gain matrix for a Kalman filter
%p is th ecovariance of the state
%h is the observation matrix
%r is the covariance of the observations
v=p*h';
vi=inv(h*v+r);
k=v*vi;
```

Published with MATLAB® 7.5

predictH

```
function hitloc=predictH(x,r_e,om_v,h0,x_w,y_w,udrag,udrag_bp,g,dt)
%t is the prediction time
%the other arguments could be bundled in a structure and passed
%that way but that calls for packing and unpacking
h=0;
%air pressure
ap_c0=1.223; ap_c2=1.071e-4;
```

```
%mach number
v_s0=340.3; %59F
v_c=2.26e-5;
%tend=t;
t=dt;
pos=[x];
hitloc=[0; 0];
while (x(2)>0)
    air_p=ap_c0*exp(-ap_c2*x(2));  %current air pressure
    %velocity and mach number
    v_s=v_s0*(1-v_c*(x(2)+h0))^.5;
    v=sqrt((x(4)-x_w)^2+x(5)^2+(x(6)-y_w)^2);
    m=v/v_s;
    k=getudrag(m,udrag,udrag_bp); %udrag&udrag_bp need 2b initialized
    kd=k(1);
    dx=S7dx(x,air_p,kd,v,r_e,om_v,g);
    x=x+dx*dt;
    t=t+dt;
    pos=[pos x ];
end
[r,c]=size(pos);
d=pos(1:3,c)-pos(1:3,c-1);
pr=-pos(2,c-1)/d(2);
hitloc(1)=pos(1,c-1)+pr*d(1);
hitloc(2)=pos(3,c-1)+pr*d(3);
```

Published with MATLAB® 7.5

Appendix B. Additional and Altered Files to Include Accelerometer Measurements

Driver7gpsacc

```
%function y=driver7()
%do the first intialization phase
initx
dt=.01;
%these are from the 6dof trajectory
%obviously in real time they would not be known in advance
%these are used as the observation feed
load obsdat
lat=45/180*pi;  %adefault value
r_e=6378000; h0=0;
x_w=0;%0 indicates no wind
y_w=0;
%air pressure
ap_c0=1.223; ap_c2=1.071e-4;
%mach number
v_s0=340.3; %59F
v_c=2.26e-5;
%gravity
g0=9.80665;
g1=.0026;
g=g0*(1-g1*cos(2*lat));
%corriolis
omega=7.2921e-5;
om_v=[omega*cos(lat);omega*sin(lat);0];
f=zeros(7);
state=[x];
pre=[];
ptrace=trace(p);
%this is the counter for between observation updates
%this helps minimize the effects of nonlinearity
%many times these substeps can be ignored and the propagation can
%be calculated in one step between observations
obs=0;
ac_obs=0;
%the next variable just counts the observations
%the observations start with count+1
%note that this variable can be used to start the observation stream
%at any desired point of the trajectory
count=0;
acount=0;
%main loop
while x(2)>0
    %air preasure etc is triggered by altitude
    %anyhow update variables that change as the state changes
    air_p=ap_c0*exp(-ap_c2*x(2));  %current air pressure
    %velocity and mach number
    v_s=v_s0*(1-v_c*(x(2)+h0))^.5;
    v=sqrt((x(4)-x_w)^2+x(5)^2+(x(6)-y_w)^2);
```

```
    m=v/v_s;  %mach number
    k=getudrag(m,udrag,udrag_bp); %udrag&udrag_bp need 2b initialized
    kd=k(1);kd_m=k(2);
    f=S7calcF(x,air_p,kd,kd_m,v,m,r_e,om_v,ap_c2,g,f);
    dx=S7dx(x,air_p,kd,v,r_e,om_v,g);
    x=x+dx*dt;
    p=propP(dt,f,p,q);
    obs=obs+1;
    ac_obs=ac_obs+1;
    %this inside loop is triggered by a GPS observation being available
    if obs==10
        count=count+1;
        %add some error to the observation
        z_er=diag(randn(3,1))*r_sd*.1;
        z=obsdat(:,count)+z_er;
        %reset counter
        obs=0;
        %if r is variable the routine getR should be designed to model
        %the observation error and included here
        K=getK(p,h,r);
        x=x+K*(z-h*x);
        p=(I-K*h)*p;
        %predict routine this will change somewhat based on specific
        %application
        %pos=predictX(x,r_e,om_v,h0,x_w,y_w,udrag,udrag_bp,g,dt,5);
        pos=predictH(x,r_e,om_v,h0,x_w,y_w,udrag,udrag_bp,g,dt);
        pre=[pre pos];
    end
    %note that GPS observations are processed first
    if ac_obs==5
        acount=acount+1;
        %first find the observation
        a_axis=accel(x,air_p,kd,v,r_e,om_v,g);
        %add some error to the observation
        r_a=abs(10*a_axis);
        za=a_axis+randn(1)*r_a;
        %reset counter
        ac_obs=0;
        % the H matrix needs to be calculated for each observation a call
        % to the routine hmatrix does this
        h_a=hmatrix(x,air_p,kd,v,r_e,om_v,g);
        %if r is variable the routine getR should be designed to model
        %the observation error and included here
        K=getK(p,h_a,r_a);
        x=x+K*(za-h_a*x);
        p=(I-K*h_a)*p;
        %predict routine this will change somewhat based on specific
        %application
        %pos=predictX(x,r_e,om_v,h0,x_w,y_w,udrag,udrag_bp,g,dt,5);
       % pos=predictH(x,r_e,om_v,h0,x_w,y_w,udrag,udrag_bp,g,dt);
       % pre=[pre pos];
    end
    state=[state x];
    ptrace=[ptrace trace(p)];
end
y=state([1 3 2],:);
```

Accel

```
function a_axis=accel(x,air_p,kd,v,r_e,om_v,g)
%calculates the force measured by an axial acclerometer
%uses the inner product of the force and the normalized velocity
%assumes the velocity is aligned with the spin axis
%7 dimensional dynamics
dx=zeros(3,1);
dx(1)=-x(7)*air_p*kd*v*x(4)-g*x(1)/r_e-om_v(2)*x(6);
dx(2)=-x(7)*air_p*kd*v*x(5)-g*(1-
sqrt(x(1)*x(1)+x(3)*x(3))/(2*r_e))+om_v(1)*x(6);
dx(3)=-x(7)*air_p*kd*v*x(6)-g*x(3)/r_e+om_v(2)*x(4)-om_v(1)*x(5);
a_axis=dx'*x(4:6)/v;
```

Published with MATLAB® 7.5

Hmatrix

```
function hmatrix=hmatrix(x,air_p,kd,v,r_e,om_v,g)
x13=(x(1)^2+x(3)^2)^.5;
dx=zeros(7,1);
dx(1)=-g*x(4)/r_e/v+g*x(5)/(2*r_e)/x13*x(1)/v;
dx(3)=-g*x(6)/r_e/v+g*x(5)*x(3)/x13/r_e/v;
dx(4)=-x(7)*air_p*kd*x(4)*2-g*x(1)/r_e;
dx(5)=-x(7)*2*air_p*kd*x(5)-g/v*(1-x13/(2*r_e))+om_v(1)*x(6);
dx(6)=-x(7)*2*air_p*kd*v*x(6)-g*x(3)/r_e/v;
dx(7)=-air_p*kd*v*v;
hmatrix=dx';
```

Published with MATLAB® 7.5

INTENTIONALLY LEFT BLANK.

Appendix C. Additional Routines of Potential Interest

Inits8 with drag, drift, and spin

```
f=zeros(8,8);
f(1,4)=1;
f(2,5)=1;
f(3,6)=1;

dr1=[.27754e-4 0 0 0 0];
dr2=[.29407446e-3 -.16856609e-2 .39541129e-2 -.40706187e-2 .15497302e-2];
dr3=[-.56492074 .24140455e1 -.38636028e1 .27445913e1 -.73004070];
dr4=[.16449122e1 -.6472231e1 .95427249e1 -.62486232e1 .15332897e1];
dr5=[-.38991679e-2 .12251269e-1 -.14008227e-1 .70697832e-2 -.13323438e-2];
dr6=[.17693159e-3 -.14042065e-3 .74643008e-4 -.21621397e-4 .26788426e-5];

drft1=[.94178026e-2 -.26634522e-2 .61690538e-2 -.32734184e-2 -.33347962e-2];
drft2=[.10546990e2 -.48064499e2 .82211585e2 -.62499311e2 .17816323e2];
drft3=[.26615371e2 -.10493487e3 .15495878e3 -.10156664e3 .24936711e2];
drft4=[-.48373662 .14761935e1 -.16536581e1 .82453192 -.15394547];
drft5=[.15625846e-1 -.16050604e-1 .17887456e-1 -.69302356e-2 .96186882e-3];

dspin1=[.7e-2 -.26504608e-2 -.90103102e-3 .2528689e-2 -.11479416e-2];
dspin2=[.6724987e-2 -.24994776e-2 .71838136e-3 -.12021482e-3 .80635505e-5];

udrag_bp=[.6 .9 .99 1.06 1.5 5];
udrag=[dr1;dr2;dr3;dr4;dr5;dr6];

udrft_bp=[.84 .965 1.07 1.5 4];
udrft=[drft1;drft2;drft3;drft4;drft5];

uspin=[dspin1;dspin2];
uspin_bp=[.9 2.5];
```

Published with MATLAB® 7.5

Getudrft gets the drift

```
function y=getudrft(m,drft,bp)



row=1;
if m<bp(3)
        if m>bp(2)
            row=3;
```

```
        else
            if m>bp(1)
                row=2;
            else
                row=1;
            end
        end
else
    if m<bp(4)
        row=4;
    else
        if m<bp(5)
            row=5;
        else
            row=0;
        end
    end
end

x=[1; m; m*m; m*m*m; m*m*m*m];
y=drft(row,:)*x;
```

Published with MATLAB® 7.5

Getuspin

```
function y=getuspin(m,spin,bp)
if m<bp(1)
    row=1;
else
    row=2;
end
x=[1; m; m*m; m*m*m; m*m*m*m];
y=spin(row,:)*x;
```

Published with MATLAB® 7.5

predictX predicted location in t time units

```
function pos=predictX(x,r_e,om_v,h0,x_w,y_w,udrag,udrag_bp,g,dt,t)



h=0;

ap_c0=1.223; ap_c2=1.071e-4;

v_s0=340.3;
v_c=2.26e-5;
tend=t;
t=dt;
pos=[0;x];
while (t<tend)
    air_p=ap_c0*exp(-ap_c2*x(2));
```

```

    v_s=v_s0*(1-v_c*(x(2)+h0))^.5;
    v=sqrt((x(4)-x_w)^2+x(5)^2+(x(6)-y_w)^2);
    m=v/v_s;
    k=getudrag(m,udrag,udrag_bp);
    kd=k(1);kd_m=k(2);
    dx=S7dx(x,air_p,kd,v,r_e,om_v,g);
    x=x+dx*dt;
    t=t+dt;
    pos=[pos [t;x] ];
end
pos=x;
```

Published with MATLAB® 7.5

end

NO. OF COPIES	ORGANIZATION
1 (PDF only)	DEFENSE TECHNICAL INFORMATION CTR DTIC OCA 8725 JOHN J KINGMAN RD STE 0944 FORT BELVOIR VA 22060-6218
1	DIRECTOR US ARMY RESEARCH LAB IMNE ALC HRR 2800 POWDER MILL RD ADELPHI MD 20783-1197
1	DIRECTOR US ARMY RESEARCH LAB RDRL CIM L 2800 POWDER MILL RD ADELPHI MD 20783-1197
1	DIRECTOR US ARMY RESEARCH LAB RDRL CIM P 2800 POWDER MILL RD ADELPHI MD 20783-1197
	ABERDEEN PROVING GROUND
1	DIR USARL RDRL CIM G (BLDG 4600)

NO. OF COPIES	ORGANIZATION
2	DIRECTOR US ARMY RESEARCH LAB RDRL SER L M DUBEY B PIEKARSKI 2800 POWDER MILL RD ADELPHI MD 20783-1197
1	DIRECTOR US ARMY RESEARCH LAB RDRL SES J EICKE 2800 POWDER MILL RD ADELPHI MD 20783-1197
2	DIRECTOR US ARMY RESEARCH LAB RDRL SES A D FLIPPEN J PRICE 2800 POWDER MILL RD ADELPHI MD 20783-1197
3	DIRECTOR US ARMY RESEARCH LAB RDRL SES S A LADAS M D'ONOFRIO D WARD 2800 POWDER MILL RD ADELPHI MD 20783-1197
1	DIRECTOR US ARMY RESEARCH LAB RDRL SES P A EDELSTEIN 2800 POWDER MILL RD ADELPHI MD 20783-1197
2	COMMANDER US ARMY TACOM ARDEC RDRL MEF FA W KONICK C ROBINSON 2800 POWDER MILL RD ADELPHI MD 20783-1197
1	DIRECTOR US ARMY RESEARCH LAB RDRL WMP F A FRYDMAN 2800 POWDER MILL RD ADELPHI MD 20783-1197

NO. OF COPIES	ORGANIZATION
1	DIRECTOR US ARMY CECOM RDEC AMSEL RD C2 CS J VIG FORT MONMOUTH NJ 07703-5601
5	COMMANDER US ARMY TACOM ARDEC AMSRD AAR QEM E M BOMUS BLDG 65S AMSRD AAR AEM L D VO BLDG 65S A MOLINA BLDG 65S AMSRD AAR AEM M LUCIANO BLDG 65S AMSRD AAR AEM L M PALATHINGAL BLDG 65S PICATINNY ARSENAL NJ 07806-5000
4	COMMANDER US ARMY TACOM ARDEC AMSRD AAR AEM A S CHUNG BLDG 95 W KOENIG BLDG 95 W TOLEDO BLDG 95 T RECCHIA BLDG 95 PICATINNY ARSENAL NJ 07806-5000
1	COMMANDER US ARMY TACOM ARDEC F BROWN BLDG 151 PICATINNY ARSENAL NJ 07806-5000
1	COMMANDER US ARMY TACOM ARDEC AMSRD AAR AEM C A MOCK BLDG 171A PICATINNY ARSENAL NJ 07806-5000
1	COMMANDER US ARMY TACOM ARDEC J POTUCEK BLDG 61S PICATINNY ARSENAL NJ 07806-5000
1	COMMANDER US ARMY TACOM ARDEC AMSRD AAR AEP M CILLI BLDG 382 PICATINNY ARSENAL NJ 07806-5000

NO. OF COPIES	ORGANIZATION
11	COMMANDER US ARMY TACOM ARDEC AMSRD AAR AEP E J VEGA BLDG 94 D CARLUCCI BLDG 94 M HOLLIS BLDG 94 J KALINOWSKI BLDG 94 S PEARCY BLDG 94 AMSRD AR CCF D D PASCUA BLDG 94 AMSRD AAR AEP S R WERKO BLDG 94 M MARSH BLDG 94 Q HUYNH BLDG 94 T ZAPATA BLDG 94 N GRAY BLDG 94 PICATINNY ARSENAL NJ 07806-5000
1	COMMANDER US ARMY TACOM ARDEC D TROAST BLDG 171 PICATINNY ARSENAL NJ 07806-5000
1	COMMANDER US ARMY ARDEC AMSTA AR CCF D H RAND BLDG 61S PICATINNY ARSENAL NJ 07806-5000
2	COMMANDER US ARMY TACOM ARDEC AMSRD AR AEP I S LONGO BLDG 95N C HALKIAS BLDG 95N PICATINNY ARSENAL NJ 07806-5000
1	COMMANDER US ARMY TACOM ARDEC AMSRD AAR AEP S C PEREIRA BLDG 400 PICATINNY ARSENAL NJ 07806-5000
1	COMMANDER US ARMY TACOM ARDEC J STRUCK BLDG 407 PICATINNY ARSENAL NJ 07806-5000
1	COMMANDER US ARMY TACOM ARDEC AMSRD AAR EI R CARR BLDG 65N PICATINNY ARSENAL NJ 07806-5000

NO. OF COPIES	ORGANIZATION
1	COMMANDER US ARMY TACOM ARDEC SFAE SDR SW IW B D AHMAD BLDG 151 PICATINNY ARSENAL NJ 07806-5000
1	COMMANDER US ARMY TACOM ARDEC SFAE AMO MAS C GRASSANO BLDG 354 PICATINNY ARSENAL NJ 07806-5000
1	COMMANDER US ARMY TACOM ARDEC SFAE AMO MAS SMC R KOWALSKI BLDG 354 PICATINNY ARSENAL NJ 07806-5000
3	PRODUCT MANAGER FOR MORTARS SFAE AMO CAS MS G BISHER P BURKE D SUPER BLDG 162 S PICATINNY ARSENAL NJ 07806-5000
1	PRODUCT MANAGER FOR MORTARS SFAE AMO CAS MS J TERHUNE BLDG 354 PICATINNY ARSENAL NJ 07806-5000
1	COMMANDER US AMRY TACOM ARDEC SFAE AMO CAS A HERRERA BLDG 171A PICATINNY ARSENAL NJ 07806-5000
1	COMMANDER OFC OF NAVAL RSCH CODE 333 P MORRISSON 800 N QUINCY ST RM 507 ARLINGTON VA 22217-5660

NO. OF COPIES	ORGANIZATION
1	DIRECTOR NAVAL AIR SYSTEMS CMD TEST ARTICLE PREP DEP CODE 5 4 R FAULSTICH BLDG 1492 UNIT 1 47758 RANCH RD PATUXENT RIVER MD 20670-1456
1	COMMANDER NAWC WPNS DIV CODE 543200E G BORGEN BLDG 311 POINT MUGU CA 93042-5000
2	PROGRAM MANAGER ITTS PEO-STRI AMSTI EL D SCHNEIDER C GOODWIN 12350 RESEARCH PKWY ORLANDO FL 32826-3276
1	COMMANDER US ARMY RDEC AMSRD AMR SG SD P JENKINS BLDG 5400 REDSTONE ARSENAL AL 35898-5247
1	COMMANDER US ARMY AVN & MIS CMND AMSRD AMR SG P RUFFIN REDSTONE ARSENAL AL 35898-5247
1	DIRECTOR US ARMY RTTC ATTN STERT TE F TD R EPPS REDSTONE ARSENAL AL 35898-8052
1	ARROW TECH ASSOC W HATHAWAY 1233 SHELBURNE RD STE 8 SOUTH BURLINGTON VT 05403

NO. OF COPIES	ORGANIZATION
5	ALLIANT TECHSYSTEMS A GAUZENS J MILLS B LINDBLOOM E KOSCO D JACKSON PO BOX 4648 CLEARWATER FL 33758-4648
3	ALLIANT TECHSYSTEMS G PICKUS F HARRISON M WILSON 4700 NATHAN LANE N PLYMOUTH MN 55442-2512
8	ALLIANT TECHSYSTEMS ALLEGANY BALLISTICS LAB S OWENS C FRITZ J CONDON B NYGA J PARRILL M WHITE S MCCLINTOCK K NYGA MS WV01-08 BLDG 300 RM 180 210 STATE RTE 956 ROCKET CTR WV 26726-3548
3	SAIC J GLISH J NORTHRUP G WILLENBRING 8500 NORMANDALE LAKE BLVD STE 1610 BLOOMINGTON MN 55437-3828
1	SAIC D HALL 1150 FIRST AVE STE 400 KING OF PRUSSIA PA 19406
1	AAI CORPORATION MS 113 141 C BEVARD 124 INDUSTRY LNE HUNT VALLEY MD 21030

NO. OF COPIES	ORGANIZATION
1	DREXEL UNIV DEPT OF MECHANICAL ENGRG B C CHANG 3141 CHESTNUT ST PHILADELPHIA PA 19104
1	JOHNS HOPKINS UNIV APPLIED PHYSICS LAB W D'AMICO 1110 JOHNS HOPKINS RD LAUREL MD 20723-6099
4	CHLS STARK DRAPER LAB J CONNELLY J SITOMER T EASTERLY A KOUREPENIS 555 TECHNOLOGY SQ CAMBRIDGE MA 02139-3563
2	ECIII LLC R GIVEN J SWAIN BLDG 2023E YUMA PROVING GROUND AZ 85365
1	GD OTS E KASSHEIMER PO BOX 127 RED LION PA 17356
1	ALION SCIENCE P KISATSKY 12 PEACE RD RANDOLPH NJ 07861
1	GEORGIA TECH RESEARCH INST GTRI ATAS A LOVAS SMYRNA GA 30080
	ABERDEEN PROVING GROUND
2	COMMANDER US ARMY TACOM ARDEC R LIESKE BLDG 305 J MATTS BLDG 305 APG MD 21005-5051
1	COMMANDER CSTE DTC AT TD B K MCMULLEN BLDG 359 APG MD 21005-5059

NO. OF COPIES	ORGANIZATION
1	COMMANDER CSTE DTC AT SL B D DAWSON BLDG 359 APG MD 21005-5059
2	COMMANDER CSTE DTC AT FC L R SCHNELL BLDG 526 J DAMIANO BLDG 381 APG MD 21005-5059
1	COMMANDER CSTE DTC AT TD S WALTON BLDG 359 APG MD 21005-5059
2	COMMANDER USAATC TEDT AT ADR S CLARK B GILLICH BLDG 400 COLLERAN RD TRAILER TI APG MD 21005-5059
41	DIR USARL RDRL CII C B BODT RDRL CIN T R PRESSLEY RDRL SL R COATES RDRL SLB D J COLLINS L MOSS RDRL WMS T ROSENBERGER RDRL WML A M ARTHUR B FLANDERS T KOGLER W OBERLE R PEARSON A THOMPSON (4 CPS) D WEBB P WYANT R YAGER RDRL WML D J COLBURN M NUSCA

NO. OF COPIES	ORGANIZATION
	RDRL WML E
	F FRESCONI
	B GUIDOS
	P WEINACHT
	G COOPER
	RDRL WML F
	T BROWN
	E BUKOWSKI
	J CONDON
	B DAVIS
	R HALL
	T HARKINS
	D HEPNER
	M ILG
	G KATULKA
	D LYON
	D MCGEE
	P MULLER
	B PATTON
	P PEREGINO
	RDRL WML G
	J BENDER
	W DRYSDALE
	RDRL WMP
	B BURNS

INTENTIONALLY LEFT BLANK.